# Prediction of the Water Level During Storm Situations using Neural Networks

M.C. van de Weg

Department of Computer Science
Leiden University
Postbus 9512
2300 RA Leiden
The Netherlands
mvdweg@wi.LeidenUniv.NL
Fax: 31-715276985
Phone: 31-71527101

# Abstract

This thesis provides an overview of the research and development of a water level prediction system for Hoek van Holland during storm situations. The prediction in this project is done with neural networks.
Example storm situations are a very important factor for the quality of the neural network, but the database used in this project did not contain many severe storm situations. In this thesis several techniques and implementations are discussed for covering the lack of severe storm situations.
In this project two neural network architectures are used, radial basis networks and feed-forward networks. This last architecture is working well for predicting storms. This network can predict surge levels with a standard deviation of 14,18 centimeter, six hours in advance. This result is better than the results obtained by current prediction methods. Our approach is data-driven and we expect that we can improve our results if more data becomes available.

# Preface

This thesis contains results of the research done on water level prediction during storm situations at Hoek van Holland using neural networks.
After the 1953 severe flooding in the Netherlands people became more aware of the damage which can be done by water. After this flooding a special service was formed in order to predict the water level as accurate as possible. Predicting the water level with neural models is initiated to enhance the accuracy of water level predictions during storm situations.
The research was done as a graduation project at the University of Leiden and at the National Institute for Coastal and Marine Management in The Hague.

I would like to thank all those who helped me during the research for and the writing of this thesis: Joost Kok, my thesis advisor, who gave me great support in finding the right way to describe the results of this project and for assisting me with problems that occurred during this project; Peter Heinen, my advisor at the National Institute for Coastal and Marine Management, who helped me with his ideas about a Storm Warning System and gave me support during the six months practical period; My secondary thesis advisor, for reading and judging my this thesis; Maarten Lamers, a PhD student who gave me hints and the access to the *Heskes* routines; Hans Wüst for providing me data and information on the storm situations in the period 1990 till 1996. I would also like to thank Koos Doekes, Eric Marsman, Jan Blaauw, Ardjoen Poeran and all colleagues at the department ITS who gave me so generously of their time and knowledge, and created a environment in which ideas developed rapidly. I also would like to thank Mira Gleisberg for her encouragement and tolerance during the months I spent in the company of my computer.

Marco van de Weg — Leiden, July 1997

# Contents

# Chapter 1

# Introduction

In this chapter we introduce the reader to the project. In section 1.1 we give information concerning the background of this project. In section 1.2 we give a short introduction to the working environment and in section 1.3 we give more details about the Storm Warning Service of the National Institute for Coastal and Marine Management. In section 1.4 we discuss the currently available prediction methods for forecasting the water level. Finally in section 1.5 we give a short overview of the rest of this project.

## 1.1 Background of this project

In 1989 the Directorate North Sea of the Dutch department Public Works and Water Management (*Dutch: Rijkswaterstaat*) had the idea to construct a water level prediction system which uses neural network techniques. A few years later this has been implemented. This system continuously predicts the water level for various locations in the Netherlands. When the expected water level is just around the average the system performs well. But when the expected water level is above the average, like in storm-situations, the system shows a systematic error between the predicted water level and the real water level.

So the next step is to predict the water height at storm-situations using neural networks. The National Institute for Coastal and Marine Management (Dutch: *RijksInstituut voor Kust en Zee*, see section 1.2) initiated this project in order to react as accurate as possible on coming storms.
The primary aim of this project is to develop a neural network which can predict the highest water level during storm situations at *Hoek van Holland*. If the neural computation approach gives good results, then further research will be done.

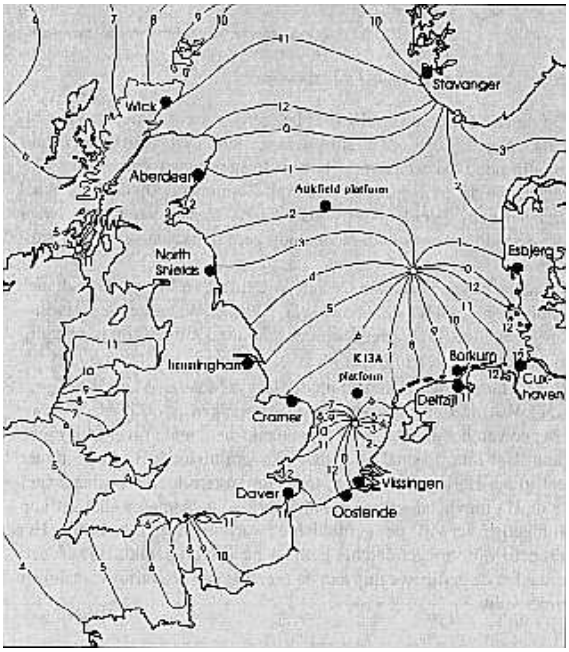


Figure 1.1: Overview of measuring locations

## 1.2 Working environment

The project is carried out at the research department of the department Information, Technology & Systems (ITS) at the National Institute for Coastal and Marine Management, in The Hague. As part of the Ministry of Transport, Public Works and Water Management, the National Institute for Coastal and Marine Management provides advice and information on:

- coastal flood protection;
- the sustainable use of estuaries, coasts and seas.

The knowledge and information function of the National Institute for Coastal and Marine Management is managed through account plans for the regions: North Sea , Wadden Sea and Delta Area.

## 1.3 Storm Warning Service

In the night of 1 February 1953 1835 people where killed and 135.000 hectares in the southern part of the Netherlands where flooded. This was the biggest flooding catastrophe in recent Dutch history. After this disaster people realized that the Netherlands lacked adequate protection against flooding. After 1953

many dams and dykes where build and the Oosterschelde is guarded by a storm surge-barrier. The catastrophe in 1953 shows that accurate water level prediction is very important for the national safety of most of the Netherlands. Prediction of the water level on coming storms is not only valuable for the Netherlands but also for many other countries in the world. In Table 1.1 we show some regions where severe floodings occurred.

| Date | Region | Maximum surge level | Lives lost |
|---|---|---|---|
| 1170, 1219, 1287 | Zuider Zee | unknown | 100.000 |
| 1864, 1876 | Bangladesh | unknown | 250.000 |
| September 1900 | Galveston, Texas | 4.5 m | 6.000 |
| January/February 1953 | Southern North Sea | 3.0 m | 2.000 |
| March 1962 | Atlantic Coast, USA | 2.0 m | 32 |
| November 1970 | Bangladesh | 9.0 m | 500.000 |

Table 1.1: Estimated results of some historical storm surge events

If a high water level is expected then the Storm Warning Service (SVSD) watches and forecasts the coming water level. The purpose of the SVSD is to inform the dyke and dam authorities and other persons in the Dutch tidal region about the occurrence of high water levels whenever a storm surge is expected. It is the SVS-D's task to issue warnings and give information concerning expected and actual tidal rises and storm surges and to recommend dyke watches. This information is intended responsible for the bodies in the coastal area and the estuary on lower river areas that are responsible for the protection of the land against high water levels. Only in exceptional cases the SVSD supplies major port authorities with information.

The SVSD-warning levels, different types of surges and some typical situations are shown in Table 1.2.

## 1.4 Available prediction methods

To predict the water level we need information about the tide, water level and the weather forecast. The Royal Meteorological Institute (KNMI) takes care of predicting the weather. In order to create an appropriate weather forecast the KNMI runs an atmospheric model four times a day, which generates a thirty hour forecast. With the forecast of wind- and atmospheric pressure area's the Directorate-General of Public Works and Water Management (RWS) can predict the water movement of the North Sea. This is done with the Continental Shelf Model (CSM). This model comprises the North Sea and a part of the Atlantic

## 1.4. AVAILABLE PREDICTION METHODS

| *Sector*<br>Measuring Location | *Schelde*<br>Vlissingen | *West of Holland*<br>Hoek van Holland | *Den Helder*<br>Den Helder | *Harlingen*<br>Harlingen | *Delfzijl*<br>Delfzijl | Exceedance Frequency<br>In One Year |
|---|---|---|---|---|---|---|
| Prewarning level | 310 | 200 | - | - | 260 | +/- 5 |
| Warning level | 330 | 220 | 190 | 270 | 300 | +/- 2 |
| Border level | 350 | 260 | 225 | 305 | 350 | 0.5 |
| Alarm level (dyke guarding) | 370 | 280 | 260 | 330 | 380 | +/- 0.2 |
| High tides | 305–350 | 210–260 | 165–225 | 225–305 | 260–350 | 5–0.5 |
| Low storm surges | 350–385 | 260–300 | 225–275 | 305–350 | 350–410 | 0.5–0.1 |
| Medium storm surges | 385–440 | 300–360 | 275–340 | 350–415 | 410–495 | $10^-1 \cdots 10^-2$ |
| High storm surges | 440–495 | 360–430 | 340–395 | 415–465 | 495–560 | $10^-2 \cdots 10^-3$ |
| Uncommon high storm surges | 495–550 | 430–505 | 395–445 | 465–505 | 560–620 | $10^-3 \cdots 10^-4$ |
| Extreme storm surges | $\geq 550$ | $\geq 505$ | $\geq 445$ | $\geq 505$ | $\geq 620$ | $\leq 10^-4$ |
| 1 February 1953 | 455 | 385 | 325 | 334 | 307 | |
| 3/4 January 1976 | 394 | 298 | 297 | 369 | 435 | |
| 29 en 30 August 1996 | 361 | 197 | 122 | 165 | 218 | |
| Highest level known | 455 | 385 | 325 | 369 | 460 | |

Table 1.2: Water levels of SVSD Base Locations in Centimeters Above NAP

Ocean. In order to predict the water level, the CSM divides the prediction area in areas of 8 by 8 kilometers. The following phenomena are handles within this model:

- water transport;
- tide movement;
- the friction at the bottom of the sea;
- the forces on the water due to the earth rotation;
- the drying out and the flooding of a sand bank;
- the effects of variations in the atmospherical pressure;
- the forces of the wind that effects the water.

In the nineties the whole prediction protocol was improved with a few 'new' techniques, like:

- improvement of the wind model;
- using an improved Continental Shelf Model;
- data-assimilation using the Kalman filter.

To illustrate the effects of the introduction of these improvements on the SVSD forecasts we compare the results of different periods in Table 1.3.

The Storm Warning Service (SVSD) uses various models (CSM , KNMI forecast and other) to predict the water level during storm situations, but most important and hard to model is the knowledge of the storm-experts from the SVSD. With

| Period | 1986-1995 | 1986-1990 | 1991-1995 |
|---|---|---|---|
| Standard deviation | 20.1 | 23.6 | 17.2 |
| Mean deviation | -11.1 | -11.9 | -10.4 |
| Number of observations | 30 | 14 | 16 |

Table 1.3: Improvement of SVSD forecasts 1986-1995

| Period | SVSD | | | KNMI | | | 6h CSM | | |
|---|---|---|---|---|---|---|---|---|---|
| | Std | Mean deviation | # | Std | Mean deviation | # | Std | Mean deviation | # |
| 1954 ··· 1995 | 26.0 | -7.2 | 83 | 29.2 | -4.4 | 83 | 30.4 | -19.5 | 24 |
| 1954 ··· 1983 | 29.2 | -5.1 | 51 | 33.3 | -2.2 | 51 | - | - | - |
| 1984 ··· 1995 | 19.7 | -10.5 | 32 | 21.2 | -7.8 | 32 | 30.4 | -19.5 | 24 |
| 1954 ··· 1965 | 24.5 | -13.2 | 11 | 24.5 | 36.2 | 11 | - | - | - |
| 1966 ··· 1975 | 32.9 | -9.0 | 22 | 36.2 | 8.7 | 22 | - | - | - |
| 1976 ··· 1985 | 24.5 | 4.2 | 20 | 26.5 | 13.9 | 20 | - | 49.0 | 1 |
| 1986 ··· 1995 | 20.1 | -11.1 | 30 | 21.8 | -8.5 | 30 | 27.2 | -22.5 | 23 |
| 1991 ··· 1995 | 17.2 | -10.4 | 16 | 19.8 | -8.6 | 16 | 24.6 | -27.1 | 16 |
| 1954 ··· 1994 | 26.4 | -7.3 | 79 | 29.7 | -4.4 | 79 | 30.5 | -15.6 | 20 |
| 1986 ··· 1990 | 23.6 | -11.9 | 14 | 24.6 | -8.5 | 14 | 32.0 | -12.1 | 7 |

Table 1.4: Accuracy SVSD-predictions 1954 - 1995 at Hoek van Holland

their experience in recognizing different situations they can predict the water height very well. In Table 1.4 the accuracy of the predictions at different periods are presented. These tables show that the current accuracy of the water level prediction using models is increased. This improvement results in a better prediction for SVSD. The Directorate-general of Public Works and Water Management expects to achieve the target value of a standard deviation of 15 centimeter for a six hours prediction in an operational environment in the near future.

## 1.5 Organization

Within this project we examine the possibilities to find a prediction method for severe storm situations using neural networks. The research which is done is discussed in the following chapters:

- In chapter 2 we discuss the natural effects that influence the water level, like showers and wind;
- In chapter 3 we give more detail on the data set which has been used in this project;

## 1.5. ORGANIZATION

- We discuss the methods and models used in chapter 4;
- In chapter 5 we present the results of the project;
- Finally in chapter 6 we evaluate the project and give directions to future research.

A small dictionary is included as an Appendix.

# Chapter 2

# Natural effects that influence the water level

The water level at the Dutch coast is influenced by the tidal cycles and the interaction between the atmosphere and the water surface. In this chapter we discuss these more or less unpredictable natural effects on the movement of the water. First, in section 2.1 we give more detail about the tidal cycles of the water level. After this we discuss in section 2.2 different water level terms. In section 2.3 we discuss the influence of the wind on the water level, in section 2.4 we give more detail about the influence of the atmospheric pressure, and finally in section 2.5 we give more detail about the extreme and important storm-situations.

## 2.1 Tidal cycles

First, it is important to give a interpretation of the term *tide* which will be used further on. We define tide as a periodic movement which is directly related to some periodic geographic force in amplitude and phase. So the meteorological components are non-tidal. We use the term *astronomical water level* for the sum of the periodical movement of water level at a certain location and time. To describe the non-tidal components which influence the water level (like wind force and direction of the wind) we use the term *residual*. So, the real water level at a location at a specific time is the addition of the astronomical tide and the residual.

This natural phenomenon results in a difference in water level at the Dutch coast from two till four meters. Using the so-called harmonic analysis it is possible to determine the astronomical water level. This is the expected water level when no atmospherical effects are influencing the water level. So it is possible to compute the astronomical water level for a larger period. Every year, the Directorate-General of Public Works and Water Management publishes a book with the computed tidal cycles [22].

The periodical water level difference (astronomical tide) at the Dutch coast is caused by the gravitational pull between the sun, moon and earth, and the spinning force that corresponds with their movement. The following periodical movements influence the atmospherical water level:

- earth rotation (0.997 day);
- elliptical orbit of the moon around the earth (27.32 days);
- elliptical orbit of the earth around the sun (365.26 days);
- rotation of the moon-orbit around the earth (8.85 years);
- orientation moon-orbit versus earth-orbit (18.6 years).

## 2.2 High- and low water

The tidal *low water* is the minimum tidal level reached during a cycle. The actual level of low water may be greater or less than the estimated tidal level, because of meteorological effects (see chapter 2). Similarly *high water* is the highest water level reached during a cycle.

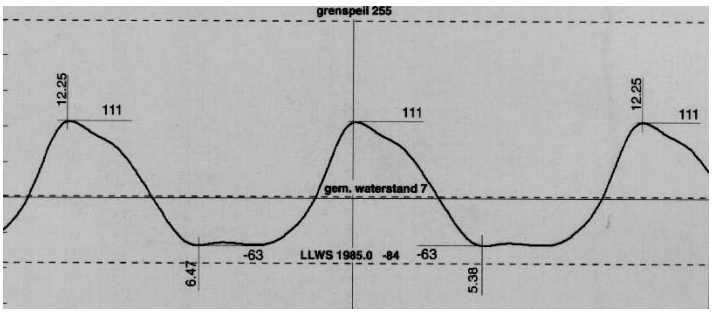


Figure 2.1: Mean tide of Hoek van Holland

The largest differences between tides in Holland occur at *Bath* in the *Westerschelde*, the mean of this difference is 4.75 meters. The smallest tide differences occur at *Den Helder* where the mean of the difference is 1.37 meters. In figure 2.2 we present the mean high water and low water at the Dutch coast, in order to illustrate the differences between tides. This figure shows that the biggest difference between the tidal high water and low water can be found at Vlissingen. This difference decreases until Den Helder, after Den Helder it increases.

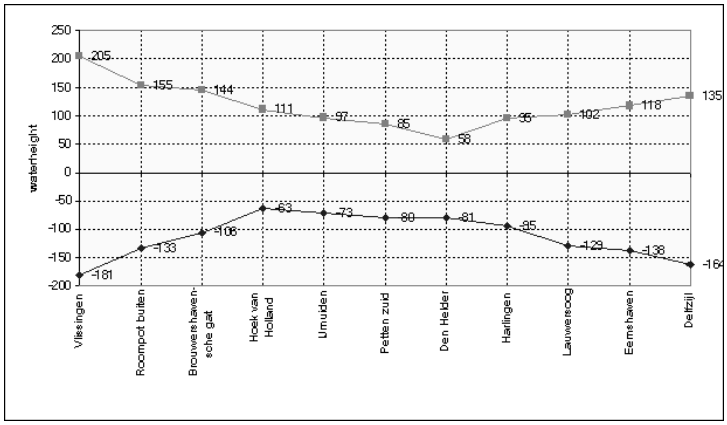


Figure 2.2: Mean High- and Low water at the Dutch coast

The height change, the low water level and the high water level vary throughout the month. The tides build up to a maximum and fall to a minimum twice a month.

The tides with the largest difference between high and low water are called springs and those with the smallest difference are called neaps. Spring tides happen just after every full and new moon, when the sun, moon and earth are in line. Because of the regular motion of the sun, moon and earth, spring tides in the North Sea occur roughly a few days after the full and new moons, and for any given location, always at roughly the same time of day.
More detailed information on tides in the Netherlands can be found in [21] and [22].

## 2.3 Wind effects

Due to the interaction between atmosphere and the seas the real water level at Hoek van Holland differs one to two meters from the computed astronomical level. At approximately 18 per cent of the total time the deviation is bigger than 30 centimeter. The direction and the power of the wind are important parameters for the real water level. In this section we give an overview of some effects that occur in the North Sea.

*Sea wind*
When, in the summer time, the land is warmed up by the sun the atmospheric pressure at land drops. The air at sea is not warmed up so the pressure at open sea remains nearly the same. The difference in atmospheric pressure results in

a wind which is blowing towards the land, which is called *sea wind*. These air-movements are only local, so the effect on the water level of the North Sea can be neglected.

*The oscillation effect*
The North Sea behaves like a big tub, the water in it oscillates from the English coast towards the Dutch coast and backwards. Oscillations occur in the following situations:

- after a strong south-west wind before a coming wind front. During this the water level is lowered. If the wind lies down, the water level stops abruptly with lowering;
- dropping of wind from the north;
- quick variations in atmospheric pressure.

The oscillation is initialized by a quick change in water level. If we assume a situation with a quick lowering of the water level, then we see that after approximately eighteen hours after the change half of the water returns. And again after eighteen hours a quarter of the original lowering goes away.

The estimated pattern of the tides is modified by irregular factors like the atmospheric pressure and the winds acting on the sea surface. Figure 2.3 shows an example situation where meteorological factors influences the actual water height.

## 2.4 Atmospheric pressure

Changes in atmospheric pressure produce changes in the forces acting vertically on the sea surface. The term *inverted barometer effect* describes the water level change during this situation.

An increase in atmospheric pressure of one millibar will produce a decrease in sea-level of one centimeter. This response of sea-level is called the *inverted barometer effect*. During a typical year tropical pressures may vary between 980 millibar and 1030 millibar. For the standard atmospheric pressure of 1013 millibar this implies a range of static sea levels between +33 centimeter and -17 centimeter.

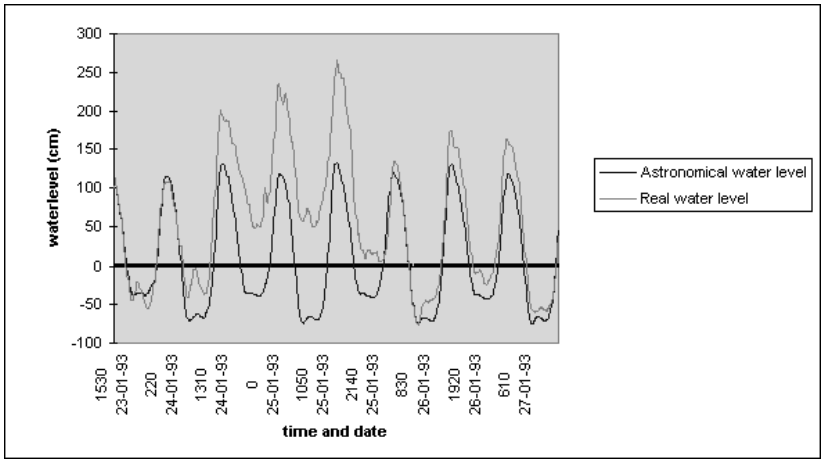


Figure 2.3: Influences on the tide during a storm in January 1993

## 2.5 Storm surge development in the North Sea

We use the term *storm surge* to describe a phenomena wherein sea level rises above the normal tide level when polar depressions move from the ocean along or across a coastal region. The North Sea has been described as a splendid sea for storm surges (see [6]) because it is open to the North Atlantic Ocean the polar depressions, which travel across this entrance from west to east, are able to set the water in motion.

Surges are generated by wind-forces acting over the sea north and northwest of Scotland and by atmospheric pressure forces travelling from the deep Atlantic to the shallow North Sea. When these water movements are heading to the North Sea they are affected by geographic forces and by the shallowing water as they approach the narrow region in the south. Because the surges travel like the tides from north to south along the coast of Scotland, England and the Netherlands, reliable warnings are possible. The most occurring situation which lead to storm surges is a depression which crosses from West to East the North and the center of the North Sea.

# Chapter 3

# Data analysis

In this chapter we describe the input data which are used to predict the water level at a certain time. In the following sections we present the data-storage and -retrieval system (section 3.1) used for collecting water level information, the time systems which are used in the database (section 3.2), and the water level data, which are described in section 3.3). In section 3.4 we discuss the meteorological data, and in section 3.6 we give more detail about the selection and classification of the complete data set. Finally in section 3.7 we discuss the gaps in the data.

## 3.1 Data storage & retrieval

The Monitoring System Water (MSW) supplies continuous information on the present water level of Dutch coastal and inland waters.
The central MSW computer is connected to the British east coast by a permanent telephone connection, so British North Sea water level information is available in the MSW-system.
The measurement network comprises more than 160 measuring stations, approximately 25 junction and sub junction stations and a central computer system. Each measuring station is connected by a permanent telephone line or by radio to a (sub)junction station. The data from these stations are sent to the central computer system in The Hague. Mean values are determined at the measuring stations every 10 seconds. After automatic monitoring, these 10 second mean values are converted in the central computer system into 10 minute mean values, which are stored in the database. MSW supplies information for:

- research and description of the hydrology of the country;
- warning services during storm surges or river floods;
- control of locks, weirs and storm surge barriers;

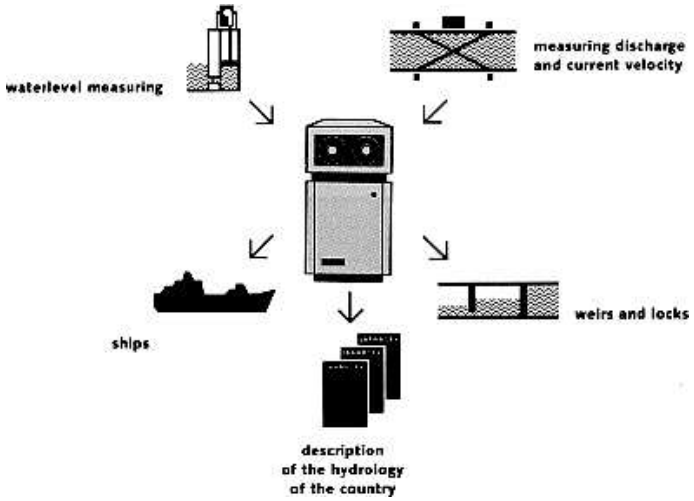


Figure 3.1: MSW Organization

- management of water distribution by weir and lock operators;
- navigation.

## 3.2 Time systems

The data from the database which we used in this project uses the Universal Time Coordinates (UTC) as a system of time. Universal Time Coordinates (UTC) is equivalent to Greenwich Mean Time (GMT), which is the system of time used in meteorology. UTC uses no summer- or winter time system and is one hour earlier than Middle European Time (MET). The parts of the databases which are based on the MET system are converted to UTC.

## 3.3 Water level data

The water level data can be split up into two types: the astronomical water level, which is discussed in section 2.1 and the real water level which is the same as the addition of the astronomical water level and the residual.

In this project the astronomical data of the following locations are available: AUK, DFZ, EPF, HVH, K13, LOW, NTS and VLI (see Table 3.1).
The astronomical data is of importance for the exactness of the prediction, because the high water at storm surges is partially caused by a high astronomical level. The astronomical level can also be used to teach the location depended sinus-like tide to the network.

| Location ID | Location name |
|---|---|
| AUK | Aukfield Platform |
| DFZ | Delfzijl |
| EPF | Euro Platform |
| HVH | Hoek van Holland |
| K13 | K13 Platform |
| LOW | Lowestoft |
| NTS | North Shield |
| VLI | Vlissingen |
| VR | Vlakte van de Raan |
| VZU | Vak Zuid |

Table 3.1: Available locations in this project

The real water level measurements are also of great importance for the estimation of the highest water level during storms. The water level data give besides information about the interaction between the surface of the sea and the meteorological influences, information about the wave-like water level heightening or lowering which crosses the estuary between Scotland and Norway and arrives at the North Sea. The measurements at the English coast give more information about the speed of traveling of the wavelike-disturbance.
In this project the real water level of the locations AUK, DFZ EPF HVH, K13, LOW, NTS, VLI and VR are available.

## 3.4 Meteorological data

The atmospheric pressure and specially the inverted barometer effect (section 2.4) are of secondary importance for the prediction of the water level at Hoek van Holland, so it suffices to just some few atmospherical pressure data. The following atmospheric pressure locations are available in this project: AUK, EPF, HVH and VZU (for explanation of the codes see Table 3.1).
The wind data is of great importance for the exactness of the neural prediction system. The wind has two components, the wind velocity and the direction of the wind, and hence we can represent it as a vector. If we represent this vector in polar coordinates there will be a discontinuous behavior in the representation of the direction of the wind. In order to avoid this problem, we use the following transformations:

- wind_x = wind velocity * cos(direction of the wind)

- wind_y = wind velocity * sin(direction of the wind)

If we use this representation and the wind changes direction from 1 degree to 360 degrees as in figure 3.2, the difference is only 1 degree and not 359 degrees. The locations AUK, EPF, HVH, K13, VR and VZU are available in the wind data files.

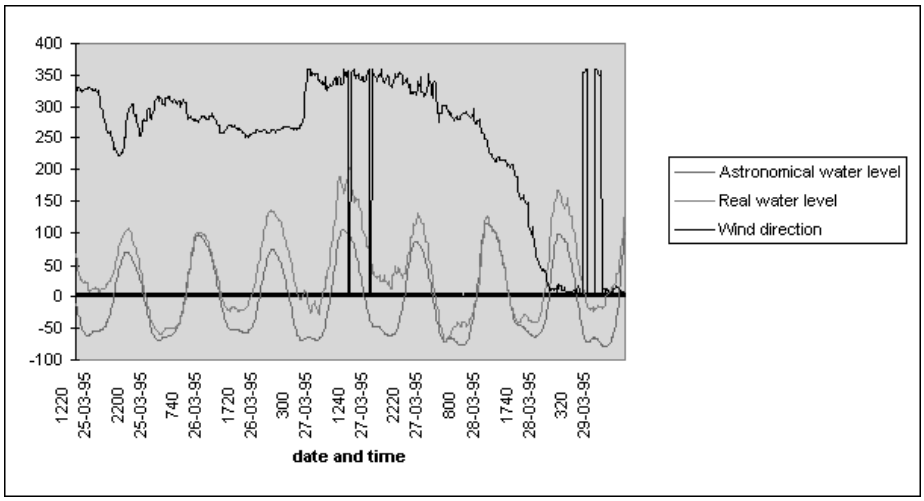


Figure 3.2: Change in the direction of the wind during a storm

## 3.5 Database description

The database with the water level and meteorological information consists of numerous files with for every data type different files. These files are named as follows:

*Water level data*
The data that describes the astronomical water level are placed in several files which filename has the following format: $wthYYMM.LOC$. The letters $YYMM$ stand for the year/month combination which the file describes. In this project we used data from the period January 1990 until March 1996. The location $LOC$ is the identification of the location which is described in this file. The values in these files are in centimeter above NAP[1]. The file in which the water level measurements are stored is named $wtoYYMM.LOC$, which has the same conventions as the astronomical water level data files.
The unity of these water level data is registered in centimeter above NAP.

[1]Normaal Amsterdams Peil

*Meteorological data*
The files in which the atmospheric pressure are stored are named: $ldoYYMM.LOC$, using the same conventions as previously described. The unity of this data is registered in millibar.
The files with wind data are named $wioYYMM.LOC$ and are transformed as described in section 3.4.
The database contains only real, non-predicted values. In a real situation we only use predicted values for the meteorological data.

## 3.6 Selection & classification

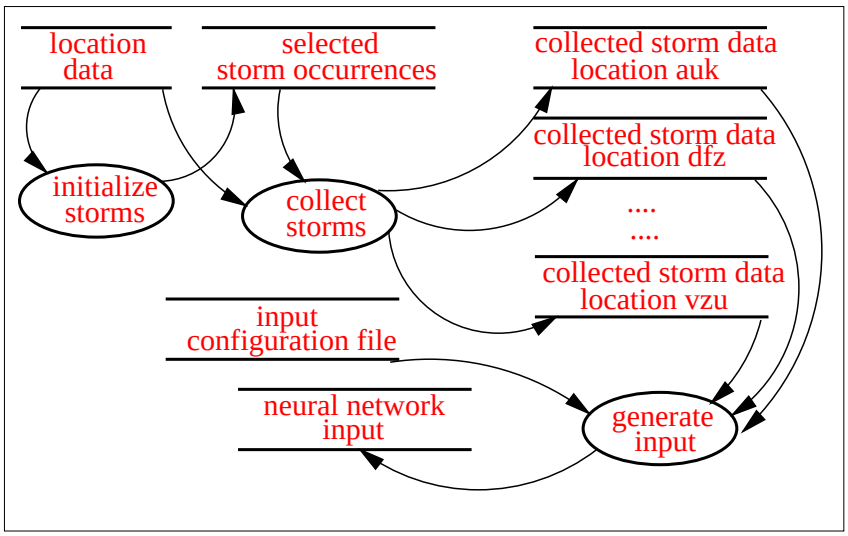


Figure 3.3: Data selection process

In figure 3.3 we present a schematic DFA-representation of the data selection process used in this project. The processes in this figure includes the following steps:

- **initialize storms** creates a file with all occurrences of storm situations. This file contains the date and time of the highest water heightening during a certain period. This process uses the database which is described in section 3.3 and 3.4. The selection of a storm situation is based on the highest water level during a period of time, which can be adjusted with parameters when executing this process;

- **collect storms** reads the previously created file with storm occurrences and creates, also using the original database, per location files containing all available data during the specified storm situations;
- **generate input** uses a configuration file to create the input for the neural network. In this configuration file the type of data and the hours are specified, in order to create a functional input set;

```
#
# This file is an example of a configuration file and contains only dummy values
#
# Location File in database Astronomical Real Pressure Wind Hours
  hvh      storm.hvh      1            0    0        0    32 34 36 38 40 42
  vli      storm.vli      1            0    0        0    34 38 42
  hvh      storm.hvh      0            1    0        0    32 34 36
  vli      storm.vli      0            1    0        0    34
  hvh      storm.hvh      0            0    1        0    48
```

Figure 3.4: Small example configuration file

The DFA of figure 3.3 also contains several storage items, these items are defined as follows:

- **location data** contains the whole database discussed in section 3.3 and 3.4.
- **selected storm occurrences** is a generated file which contains the date, time and real water level of the highest occurring water level during a variable period. This period can be adjusted with a parameter when executing the program.
- **collected storm data of various locations**, this collection of files contains all the data which is available during the storm situation (which is defined in the file **selected storm occurrences**);
- **input configuration file** is a file which is read by the *network input generation software* and contains for every location the exact moment for selecting various data. In figure 3.4 we show an example of such a configuration file. If we use this file we must keep in mind that the highest water level occurs in 48 hours, so if we select the 42nd hour we select the data at 6 hours before this water level peak;

- **collected data** is a file which contains per storm situation all collected data which will be imported into the neural network or the reduction algorithm, which will be discussed in 4.

## 3.7 Data gaps

The data contains various gaps due to the temporarily disfunctioning of some measuring locations. The quality of the predictions is suffering under these gaps. These gaps occur mostly at the stations *Aukfield Platform*, *Northshield Platform* and *Lowestoft*. Before gathering the data some preprocessing was done at the Directorate North Sea. This covers the reconstruction of water levels, wind dataand atmospheric pressure at the locations *Aukfield Platform*, *Northshield Platform*, *Lowestoft*, *K13 Platform* and *Vlissingen*. Reconstruction contains the visual and computational comparison with neighbour measuring locations. The data contains also some gaps that could not be reconstructed and could not be used for water level prediction.

# Chapter 4

# Description of models and techniques

This chapter contains an overview of the models and techniques used for the prediction of the water level. In section 4.1 we discuss the technique used for reducing the number of input nodes. In section 4.2 we give more detail about the estimation of the performance of the network. After this, in section 4.3 we give more detail on the various neural network techniques used. And finally in section 4.4 we discuss how the Matlab[1] package is used for the creation of the water height prediction system.

## 4.1 Principal component analysis

As a preprocessing technique we used Principal Component Analysis (PCA). The objective of the technique is to reduce the number of independent variables, or to ask whether we can explain all the variations observed in our data with fewer than the original number of dimensions. With this method we can select the most valuable data points.

PCA is a statistical technique which extracts the main relation in high-dimension-data. A common way to find the principal components of a data set is by calculating the eigenvectors of the correlation-coefficient-matrix. These vectors give then directions in the desired data cloud. The projections of the data on the eigenvectors are the principal components. The corresponding eigenvalues give an indication of the amount of information the corresponding principal components represent. The principal component which corresponds to a large eigenvalue represents much information in the data set and thus gives us information about the relations between the data points.

[1]Matlab is a registered trademark of The MathWorks, Inc.

PCA is also known as the Karhunen-Loève Transform (see [13]) , Hotelling transform (see [10]), and Eigenvector approach. It is also closely related to Singular Value Decomposition (see [7]).

Principal Component Analysis is applied to the data in the following way:
Consider a population of random vectors of the form of equation 4.1. Let the columns of $\mathbf{x}$ be the input patterns which will be used by the neural network. So, input vector $\mathbf{i}$ is the vector $\mathbf{x_{1i} \ldots x_{Ni}}$ in the $\mathbf{N}$ x $\mathbf{M}$ matrix of equation 4.1.

$$\mathbf{x} = \begin{pmatrix} X_{11} & X_{12} & X_{13} & \cdots & X_{1M} \\ X_{21} & X_{22} & X_{23} & \cdots & X_{2M} \\ X_{31} & X_{32} & X_{33} & \cdots & X_{3M} \\ \vdots & \vdots & \vdots & & \vdots \\ X_{N1} & X_{N2} & X_{N3} & \cdots & X_{NM} \end{pmatrix} \tag{4.1}$$

The mean vector $\mathbf{m}$ is a matrix of column averages from $\mathbf{x}$, and is given by equation 4.2:

$$\mathbf{m_x} = \frac{\sum_{\mathbf{i=1}}^{\mathbf{M}} \mathbf{X_{ij}} \mathbf{E}\{\mathbf{x}\}}{\mathbf{N}} \tag{4.2}$$

After computation of the $\mathbf{m_x}$ we compute the covariance matrix $\mathbf{C_x}$:

$$\frac{\mathbf{C_x} = \sum_{\mathbf{k=1}}^{\mathbf{M}} \mathbf{x_k} \mathbf{x_k^T}}{\mathbf{M}} - \mathbf{m_x} \mathbf{m_x}^{\mathbf{T}} \tag{4.3}$$

Where $\mathbf{T}$ indicates vector transposition.
If elements $\mathbf{x_i}$ and $\mathbf{x_j}$ are uncorrelated, their covariance is zero and therefore $\mathbf{c_{ij} = c_{ji} = 0}$ After $\mathbf{C_x S}$ has been constructed, its eigenvalues and unit eigenvectors are extracted and the principal components must be formed. Let $\xi_i$ be the eigenvectors and $\lambda_\mathbf{i}$ the corresponding eigenvalues of $\mathbf{C_x}$, sorted in descending order, so that $\lambda_\mathbf{j} \geq \lambda_{\mathbf{j+1}}$ for $\mathbf{j}$ = 1,2,...,$n$. Let $\mathbf{A}$ be a matrix whose rows are formed by the ordered eigenvectors $\lambda_\mathbf{i}$. Suppose that $\mathbf{A}$ is a transformation matrix that maps $\mathbf{x}$ into vectors denoted by $\mathbf{y}$ as in equation 4.4.

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m_x}) \tag{4.4}$$

This equation is the core of the PCA computation. The mean of the y vectors after this transformation is zero.
In most cases in which principal component analysis is used, the first few components contain a large part of the total variance. In that case, the original $\mathbf{M}$ dimensional data set can be approximated by a $\mathbf{P}$ dimensional ($\mathbf{P} < \mathbf{M}$) data set without substantial loss of information by discarding the $\mathbf{p - q}$ highest order principal components.
The matrix manipulation used to compute the $\mathbf{p}$ principal components of $\mathbf{X}$ is given in 4.5.

$$\mathbf{A} = \mathbf{XU} \tag{4.5}$$

where:

- $\mathbf{A}$ is a $\mathbf{N}$-by-$\mathbf{M}$ matrix with $\mathbf{A_{ij}}$ is equal to the value of the $\mathbf{j^{th}}$ principal component;
- $\mathbf{X}$ is a $\mathbf{N}$-by-$\mathbf{M}$ matrix with the original input patterns (equation 4.1);
- $\mathbf{U}$ is the $\mathbf{N}$-by$\mathbf{N}$ matrix whose columns are the eigenvectors of $\mathbf{C}$ arranged from left to right in order of decreasing eigenvalues.

More detailed information on PCA can be found in [11].

## 4.2 Performance estimation

The standard error of regression, or Root Mean Squared Error (RMSE) is a point estimate of the average error of prediction. It is the square root of the average squared error, corrected for degrees of freedom. The RMSE is defined as :

$$\mathbf{RMSE} = \frac{\mathbf{1}}{\mathbf{N}} \sqrt{\sum_{\mathbf{i=1}}^{\mathbf{N}} (\mathbf{x_i} - \mathbf{y_i})^{\mathbf{2}}} \tag{4.6}$$

where

- $\mathbf{x_i}$: Prediction of water height with a neural network
- $\mathbf{y_i}$: True water height
- $\mathbf{N}$: Number of test points
- $\mathbf{i}$: Testpatternnumber

We also use the bias (which is equal to the mean) to test the performance of the network. The bias is defined as:

$$\mathbf{Bias} = \overline{\mathbf{y}} - \overline{\mathbf{x}} \tag{4.7}$$

where $\overline{\mathbf{y}}$ is the average of the real values and $\overline{\mathbf{x}}$ is the average of the estimated values. Obviously, the bias should be as small as possible.
More information on the Principal Error Theory can be found in [5].

## 4.3 Neural network models & techniques

In this section we describe the neural network models and techniques used in this project. First in section 4.3.1 we give an introduction to the idea behind neural networks, in section 4.3.2 we describe the specific feed-forward backpropagation architecture, after this we discuss in section 4.3.3 the Levenberg-Marquardt optimization, used when training with the backpropagation learning rule. In section 4.3.4 we discuss the leave-one-out principle, which is useful when training with a small data set and in section 4.3.5 we give more detail about bagging, bumping and balancing, a collection of methods for training with a feed-forward network, and finally in section 4.3.6 we discuss another network type: the radial basis network.

### 4.3.1 Introduction to Neural Networks

This neural architecture is biological inspired, researchers usually think about the organization of the brain when considering network configurations and algorithm. In figure 4.1 we see the a biological neuron. Each biological neuron is a cell

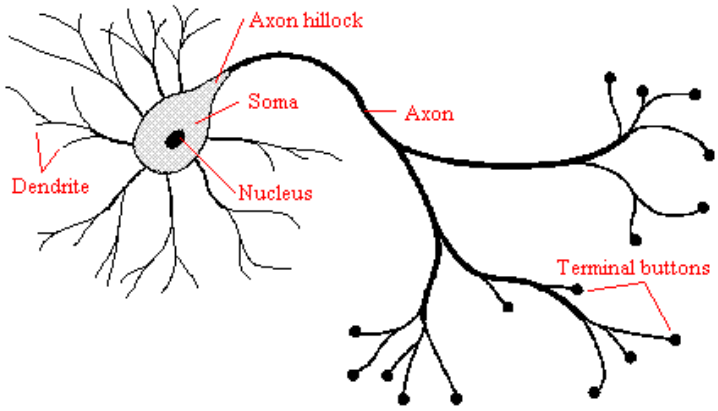


Figure 4.1: Biological Neuron

that uses biochemical reactions to receive, process and transmit information. A neuron's dendritic tree is connected to a thousand neighbouring neurons. When one of those neurons fire, a positive or negative charge is received by one of the dendrites. The strengths of all the received charges are added together through the processes of spatial and temporal summation.
The artificial neuron was designed to mimic the characteristics the biological neuron. In a neural network there a three different type of layers:

- **input layer**, a collection of neuron in which the data is presented initially;

- **output layer**, the layer with collection of nodes which represent the output of the artificial neural network;
- **hidden layer**, a collection of nodes with connections to the input- and output-layer, this layer is optional and can occur more than once.

In general a neural network is a collection processing elements, interconnected according to some plan, and processing information both externally and internally to the processing element. A processing element will take inputs from the outside world (input layer), from other processing elements (hidden and output layer), or from itself. Similar to a biological neuron, the processing element will activate only if the weighted sum of the inputs meets some criteria. It will then generate an output, based on that criteria (transfer function, see figure 4.2), and pass that on to the other processing elements or the outside world (output layer). These outputs might be compared with some overall Criterion Function, and (in

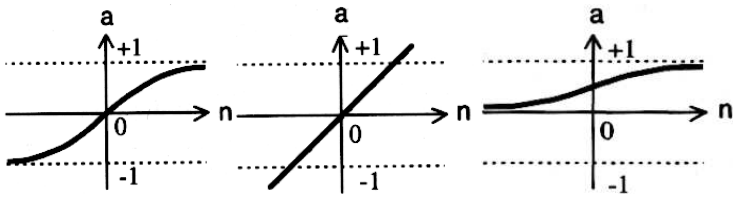


Figure 4.2: Transfer functions

most kinds of nets) an error condition will be established. Finally, the system will act on the error signal, depending upon the training style and its associated training rule.

When training with neural networks, we divide the data set into three different subsets:

- training set: a subset of the data set with those patterns which are suited for training;
- validation set: with this set we compute the error during the training phase;
- test set: when the stopping criteria is reached, the test set will be evaluated on the trained network.

The validation and the training set are in fact the training part, and the test set is the test part of the data set.

In figure 4.3 we give a simple artificial neural network that implements the

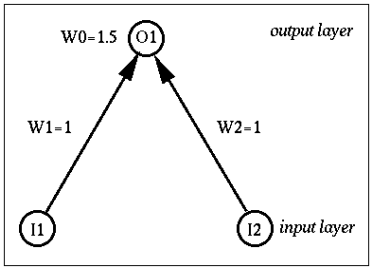


Figure 4.3: Simple Neural Network example

*AND* function (*I1 AND I2*). If *I1* and *I2* is active (value=1) then $W1 * I1$ will be added to $W2 * I2$. If the threshold value *W0* is reached, then the output unit *O1* will be activated.

### 4.3.2 Feed-forward backpropagation

Before starting the training process, all of the weights must be initialized to small random values. After the initialization step we follow the backpropagation training algorithm as is described for example in D.M. Skapura [18]. In this algorithm we use the ordered vector pairs $\{(\mathbf{x_1}, \mathbf{y_1}), (\mathbf{x_2}, \mathbf{y_2}), ..., (\mathbf{x_p}, \mathbf{y_p})\}$ to represent the training patterns, where each $\mathbf{x_i}$ represents an input patterns vector and each $\mathbf{y_i}$ represents an output pattern vector which is associated with the input vector $\mathbf{x_i}$.

**1** Select the first training vector pair from the set of training vector pairs. Call this first vector pair $(\mathbf{x}, \mathbf{y})$;

**2** Use the input vector $\mathbf{x}$ as the output from the input layer of processing elements;

**3** Use equation 4.8 to compute the activation to each unit on the subsequent layer;

$$\mathbf{net_i(t)} = \sum_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{w_{ij}(t)o_j(t)} \tag{4.8}$$

**4** Apply the appropriate activation function, which we denote as $\mathbf{f(net^h)}$ for the hidden layer and as $\mathbf{f(net^0)}$ for the output layer, to each unit on the subsequent layer. Here, appropriate refers to the one activation function that is best suited to the function to be performed by the layer of units. The selection of this function will vary by layer and by application;

**5** Repeat steps 3 and 4 for each layer in the network;

**6** Compute the error $\delta^{\mathbf{o}}_{\mathbf{pk}}$ for this pattern $\mathbf{p}$ across all $\mathbf{K}$ output layer units by using formula 4.9;

$$\delta^{\mathbf{o}}_{\mathbf{pk}} = (\mathbf{y_k} - \mathbf{o_k})\mathbf{f}'(\mathbf{net}^{\mathbf{o}}_{\mathbf{k}}) \tag{4.9}$$

**7** Compute the error $\delta^{\mathbf{h}}_{\mathbf{pj}}$, for all $\mathbf{J}$ hidden layer units by using the recursive equation:

$$\delta^{\mathbf{h}}_{\mathbf{pj}} = \mathbf{f}'(\mathbf{net}^{\mathbf{h}}_{\mathbf{j}}) \sum_{\mathbf{k=1}}^{\mathbf{K}} \delta^{\mathbf{o}}_{\mathbf{pk}} \mathbf{w_{kj}} \tag{4.10}$$

**8** Update the connection weight values to the hidden layer by using equation 4.11:

$$\mathbf{w_{ji}(t+1)} = \mathbf{w_{ji}(t)} + \eta \delta^{\mathbf{h}}_{\mathbf{pj}} \mathbf{x_i} \tag{4.11}$$

where $\eta$ is a small value used to limit the amount of change allowed to any connection during a single-pattern training cycle;

**9** Update the connection weight values to the output layer by using equation 4.12

$$\mathbf{w_{kj}(t+1)} = \mathbf{w_{kj}(t)} + \eta \delta^{\mathbf{o}}_{\mathbf{pk}} \mathbf{f}(\mathbf{net}^{\mathbf{h}}_{\mathbf{j}}) \tag{4.12}$$

**10** Repeat steps 2 through 9 for all vector pairs in the training set. Call this one training *epoch*.

**11** Repeat steps 1 through 10 for as many epochs as it takes to reduce the sum squared error (SSE) to a minimal value. The calculation of the SSE is performed for the output layer units only, across all $\mathbf{P}$ training patterns. The SSE can be computed like in equation 4.13.

$$\mathbf{SSE} = \sum_{\mathbf{p=1}}^{\mathbf{P}} \sum_{\mathbf{k=1}}^{\mathbf{K}} (\delta^{\mathbf{o}}_{\mathbf{pk}})^{\mathbf{2}} \tag{4.13}$$

In this project we used a linear output function and a sigmoidal transfer function in the hidden nodes.

### 4.3.3 Levenberg-Marquardt optimization

The Levenberg-Marquardt optimization is a more sophisticated and powerful technique than standard backpropagation which was explained in section 4.3. It gives often a better performance, but it requires more memory.
The update rule of Levenberg-Marquardt is

$$\mathbf{\Delta W} = (\mathbf{J^T J} + \mu \mathbf{I})^{-1} \mathbf{J^T e} \tag{4.14}$$

where $\mathbf{J}$ is the Jacobian matrix of derivates of each error to each weight, $\mu$ is a scaler and $\mathbf{e}$ is an error vector. If $\mu$ is very large, the expression 4.14 approximates

backpropagation. If $\mu$ is small the expression becomes the Levenberg-Marquardt method. Because this method is faster but tends to be less accurate when near an error-minima, the scalar $\mu$ is made bigger as long as the error gets smaller. If the error increases, $\mu$ is made smaller.

The increasing of speed when using the Levenberg-Marquardt technique is illustrated on the following small problem[2] :

We will train a two-layer network with five hidden neurons in order to approximate a function with 21 points. The distribution of the points is visualized in figure 4.4.

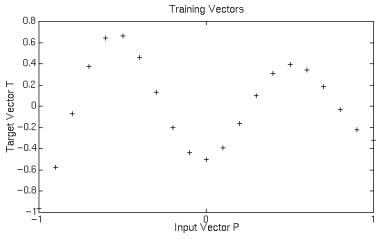


Figure 4.4: Points of the approximated function

When using the standard backpropagation rule it took 536 epochs to train until an acceptable error goal was reached. With the Levenberg-Marquardt optimization it took only 8 epochs to reach this same error goal.

We will use the Levenberg-Marquardt optimization to train the neural networks. More details on this optimization technique can be found in [14] and [12]. An implementation of the Levenberg-Marquardt technique can be found in [17].

### 4.3.4 Leave-one-out principle

Leave-one-out is an improvement on sample validation that allows one to use all of the data for training and still have an estimate of the error on unseen data. The disadvantage of this method is that you have to retrain the net many times. This method is suitable for situations in which not may patterns are available. For example: if we have $\mathbf{n}$ patterns to train a network and we use the standard training method we must split these $\mathbf{n}$ patterns in a train- and a test-part. So many patterns will not be included in the training-phase. If we use leave-one-out to train and test this network we will be able to use all $\mathbf{n}$ patterns for the train set. To achieve this we first pick one pattern $\mathbf{p^i}$ from the whole data set $\mathbf{p}$. The

[2]This problem is an adapted version of a demo from the Matlab package

```
/* 'i' represents the pattern number which is in the test set */
i = 0;

/* Initialize the array realvalues with the real water heights */
realvalues = InitRealValues(dataset);

/* Repeat these statements until all patterns have been used as a test set */
while ( patternr < ComputeLength(dataset) )
{
        /* Pick one pattern i from the data set */
        testset = GetOnePattern(dataset, i);

        /* Creates a traingsset with pattern number i left out */
        trainset = LeavePatternOut(dataset, i);

        /* Train the network using the training set */
        TrainNetwork(trainset);

        /* Test the network using the test set and store the prediction */
        prediction[i] = SimulateNetwork(testset);

        /* Increment counter to process the next pattern */
        i++;
}

/* Evaluate the array prediction, with the predictions done by the network */
EvaluatePredictions(prediction, realvalues);
```

Figure 4.5: An Implementation of the Leave-One-Out algorithm

training set contains every pattern from $\mathbf{p}$ except pattern $\mathbf{p^i}$, after this we train the network with this training set. If training is completed we use pattern $\mathbf{p^i}$ for testing. After the test of this pattern we select another pattern from $\mathbf{p}$, which has not been selected before and repeat the procedure. We continue this process until all patterns have been used as a test pattern. After training $\mathbf{n}$ networks we evaluate the predictions and create statistics, like the Root Mean Squared Error and the Standard Deviation, for the $\mathbf{n}$ predicted water levels and the matching real water levels.
The pseudo-code for the leave-on-out algorithm is given in figure 4.5.

### 4.3.5 Bagging, bumping and balancing

An other technique to improve the results of the predictions with the neural network is boot strapping [4] combined with the estimator bagging, balancing and bumping [9].

It's a popular strategy to stop training to prevent overfitting in neural networks. In general training is stopped when the error on the validation set starts increasing. The resulting network depends on the accidental subdivision in training and validation set, and often also on the, usually random, initial weight configuration and chosen minimization procedure. Small changes in the data or different initial conditions can produce large changes in the estimate. As argued in [1] and [20] it is in this situation advisable to apply the same procedure several times using different subdivisions in training and validation set and trying different initialization values. This technique is called re-sampling. A form of re-sampling is boot strapping.
In the simplest form of boot strapping, one repeatedly analyze subsamples of the data (re-sampling). A boot strap sample is a collection of randomly selected patterns of the whole data set. Depending on what one wants to do, some of the data points will occur once, some twice and some even more than twice in a boot strap sample. All patterns that do not occur in a particular sample constitute the validation set, the boot strap sample itself is the training set. Boot strapping can not only be used for estimating the generalization error but also for estimating confidence bounds for network outputs.
We can combine boot strapping with bagging [1], using this estimator, the prediction on a newly arriving input vector is the average over all network predictions. So it disregards the performance of the individual networks on the data. An other estimator is bumping [24], it throws away all networks except the one with the lowest error on the complete data set. We can also use the intermediate form of these two estimators, so called balancing [9]. A theoretical analysis of this idea can be found in [19].
It has been shown that an ensemble of neural networks does not only give more accurate predictions, but also reveals more information than a single network [9]. Moreover, ensembles of neural networks can be used to give error bars on the predictions.

### 4.3.6 Radial basis network

A different approach is to use *Radial Basis* functions in the neural network. This is an unsupervised competitive learning technique which was examined first by Moody and Darken ([15], [16]). Radial basis networks may require more neurons than the standard feed-forward backpropagation networks, but often they can be designed in a fraction of the time it takes to train standard feed-forward networks. In generally they work best when many training vectors are available.

Radial functions is a characteristic type of function, their response increases or decreases monotonically with distance from a central point. A typical radial func-

tion is the Gaussian which, in the case of a scalar input, is given in equation 4.15.

$$\mathbf{h(x) = exp}\left(-\frac{\mathbf{(x-c)^2}}{\mathbf{r^2}}\right) \tag{4.15}$$

Its parameters are its center $\mathbf{c}$ and its radius $\mathbf{r}$. Figure 4.6 illustrates a Gaussian radial basis function with center $\mathbf{c = 0}$ and radius $\mathbf{r = 1}$. A radial basis network

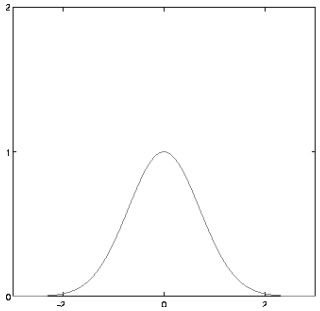

Figure 4.6: Gaussian Radial Basis Function

consist of two layers: a hidden layer and a linear output layer. In figure 4.7 we show a typical radial basis network. In this figure each of $\mathbf{n}$ components of the

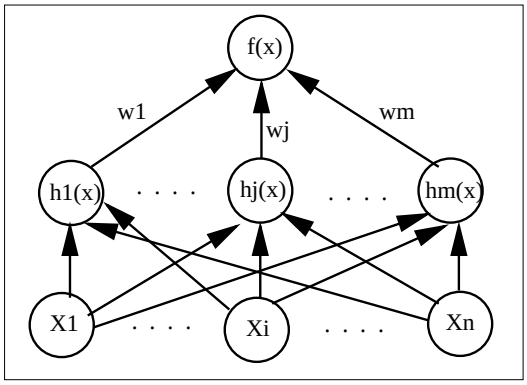


Figure 4.7: Radial Basis Network

input vector $\mathbf{x}$ feeds forward to $\mathbf{m}$ basis functions whose outputs are linearly combined with weights $\mathbf{w_j}$, where $\mathbf{1 \leq j \leq m}$. We can understand how this network behaves by following an input vector through the input layer to the output neuron. If we present an input vector to such a network, each neuron in the hidden layer (which contains radial functions) will output a value according to how close the input vector is to each neuron's output vector. The result is that the neurons

in the hidden layer with weight vectors different from the input vector will have outputs near zero. These small outputs will have only a negligible effect on the linear output neurons. If, however, the weight vector is very close to the input vector the hidden neuron will output a value near one. If a neuron has an output of one, its output weights in the second layer pass their values to the output layer. More information on radial basis networks can be found in [8].

## 4.4 Using Matlab

We used Matlab and its Neural Network Toolbox for creation of the neural networks except for the routines described in section 4.3.5, which where developed at the University of Leiden. The Neural Network Toolbox is a collection of Matlab functions for the design, training, and simulation of neural networks. It supports a wide range of network architectures with an unlimited number of processing elements and interconnections. The toolbox is delivered as Matlab M-files, enabling users to see the algorithms and implementations. A M-file is a file which includes a Matlab program which can be interpreted in Matlab. To illustrate the computational power of Matlab we show how to compute the covariance matrix (Y) using the following three Matlab statements:

- **[n,p] = size(X);**
- **X = X - ones(n,1) * mean(X);**
- **Y = X' * X/(n-1);**

In Matlab we developed several train- and simulation-programs which can be used for further research. More information on these routines can be found in [23].

# Chapter 5

# Results

In this chapter we present the results obtained using the network architectures. First in section 5.1 we discuss the configuration used in the test- and train phase. In section 5.2 we discuss the simulations with the radial basis architecture. Hereafter, in section 5.3, we give more detail about the simulations with the feed-forward network. In section 5.4 we discuss some of the problems that occurred during these simulations. And finally in section 5.5 we give some conclusions on the simulations.

## 5.1 Configuration

In this section we discuss the possible configurations which we examined for our neural architectures.

### 5.1.1 Network parameters

In order to find the optimal network we first identify the parameters which can be adjusted before the training phase. These parameters are:

- **data-selection**: the water level-, astronomical-, air pressure- and wind data which defines the input of the network (chapter 3);
- **reduction of input nodes**: the number of reduced nodes after the application of PCA (section 4.1);
- **distribution of validation-, test- & training set**: the number of patterns in each set;
- **network architecture**: the number of hidden nodes, and the transfer functions.

### 5.1.2 Validation-, test- and training set

To qualify the generalization ability of the network for storm-situations we split the input data into three categories, C1, C2 and C3. These categories are composed as follows:

**C1** *waterheight* $\leq$ 165 (in cm above NAP[1]). The patterns in this category are not that important for the prediction of the storm.There are many patterns in this category, so we take a few patterns from this category to train. We put the other patterns into the validation set.

**C2** 165 $<$ *waterheight* $<$ 180 (in cm above NAP). This category is more important (section 3.6). Every available pattern is a member of the data set.

**C3** *waterheight* $\geq$ 180 (in cm above NAP). This category is the most important category. Just like in category **C2**, every available pattern is a member of the data set.

During the development stage of this project we used many configurations. In section 5.2 and 5.3 we give more detail about these configurations applied to different types of neural architectures. We used the categorization to create the validation-, test- and training set. To create these sets we take a defined part of each category, so that the most severe storms occurs in the training set.

### 5.1.3 Selection of input data

In order to train the network to an acceptable level we have to carefully select the data from the database. To achieve this, we made different configurations files which extract the data from the database. In short, the configurations contains the following information:

- **CFA**: configuration based on previous water prediction models;
- **CFB**: Omit the wind data;
- **CFC**: Omit the atmospheric pressure data;
- **CFD**: Configuration based on CFA, but less data in it;
- **CFE**: Omit location North-Shields.

[1]Normaal Amsterdams Peil

The definition files for these five configurations are included as Appendix B. In total there are 207 situations (input patterns) with a water height greater than 150 centimeter above NAP. During data-selection 29.5 till 37.7 per cent of the total amount of patterns is lost. In Table 5.1 is shown that some locations, in particular Aukfield Platform (AUK), Northshield (NTS), Lowestoft (LOW) and Partition-South (VZU), do not function well during high water situations.

| | | | Unavailable Data at Location | | | |
|---|---|---|---|---|---|---|
| Configuration | # Input nodes | # Available Patterns | AUK | NTS | LOW | VZU |
| CFA | 94 | 129 | 25 | 47 | 3 | 3 |
| CFB | 60 | 146 | 11 | 47 | 3 | 0 |
| CFC | 80 | 140 | 14 | 47 | 3 | 3 |
| CFD | 77 | 141 | 13 | 47 | 3 | 3 |
| CFE | 86 | 170 | 26 | 0 | 6 | 5 |

Table 5.1: Data Loss using Different Configurations

In Table 5.2 the percentage of data loss per location and per data type is shown.

| Location | Atmospheric Pressure | Astronomical | Real | Wind |
|---|---|---|---|---|
| AUK | 6.17 | 52.5 | 71.6 | 5.9 |
| DFZ | 100.0 | 0.0 | 0.0 | 100.0 |
| EPF | 0.0 | 0.0 | 1.0 | 0.0 |
| HVH | 0.0 | 0.0 | 0.0 | 0.0 |
| K13 | 100.0 | 0.0 | 0.0 | 0.0 |
| LOW | 100.0 | 0.0 | 1.9 | 100.0 |
| NTS | 100.0 | 0.0 | 20.1 | 100.0 |
| VLI | 100.0 | 0.0 | 0.0 | 100.0 |
| VR | 100.0 | 100.0 | 32.8 | 2.6 |
| VZU | 1.3 | 100.0 | 100.0 | 1.0 |

Table 5.2: Percentage of Data Loss Per Location

### 5.1.4 Feed-forward network architecture

In order to find the appropriate architecture we examined various architectures, the results are presented in section 5.2 and 5.3.

If we consider a layered network with continuous-valued units, with activation function $\mathbf{g(u) = u}$ for output units and $\mathbf{g(u) = \frac{1}{1+e^{-u}}}$ for hidden units, a network implements a set of functions $\mathbf{y_i = F_i\{xk\}}$ from input variables $\mathbf{x_k}$ to output variables $\mathbf{y_i}$, where $\{\mathbf{x_k}\}$ means $\mathbf{x_1, x_2, x_3, \cdots, x_N}$. Considering a network with no hidden layers, it computes the equation described in 5.1.

$$\mathbf{y_i} = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{w_{ik} x_k} - \theta_\mathbf{i} \tag{5.1}$$

A network with one hidden layer computes:

$$\mathbf{y_i} = \sum_{\mathbf{j=1}}^{\mathbf{N}} \mathbf{W_{ij} g}(\sum_{\mathbf{k=1}}^{\mathbf{N}} \mathbf{w_{jk} x_k} - \phi_\mathbf{j}) - \theta_\mathbf{i} \tag{5.2}$$

and so on. Cybenko examined in [2] the number of hidden layers and hidden units we need to approximate a particular set of functions like $\mathbf{F_i\{x_k\}}$, to a given accuracy. He concluded that we need at most two hidden layers, with arbitrary accuracy obtainable given enough units per layer. It has also been proved in [3] that only one hidden layer is enough to approximate any continuous function. Of course the number of hidden layers depend on the used hidden units. In many cases the necessary number of hidden units is not known.

### 5.1.5 Model reduction

In order to achieve an acceptable forecast of the water level we have to reduce the number of input nodes associated with the size of the input matrix. We used Principal Component Analysis to achieve this (section 4.1).
If we apply Principal Component Analysis to a network input file with 94 input parameters we get information about the variance of the corresponding principal component. If we make a plot then we can determine a reduction factor. In figure 5.1 we present the resulting plot of the eigenvalues. In this figure we see that component one till ten have great influence on the total variance of the data set. Component 11 till 26 has less influence in the total variance of the data set, the rest of the components can be neglected because they add very little extra information to the data set.

We tried different reduction factors, in the sections 5.2 and 5.3 we give an overview of the most important results using different data reduction factors.

## 5.2 Radial basis network

This section gives results for the radial basis architecture (see also section 4.3.6). First we give an overview of the different configurations used. In section 5.2.2 we

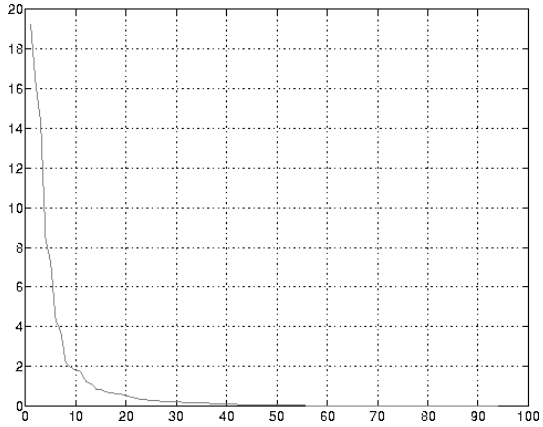


Figure 5.1: Plot of Eigenvalues created by Principal Component Analysis

discuss the results when training with the standard radial basis technique. Finally in section 5.2.3 we apply the leave-one-out technique to radial basis training.

### 5.2.1 Configuration

In order to predict as accurate as possible we used the five configurations described in Table 5.3 for the Radial Basis architecture, as explained in section 4.3.6.

This table contains the following columns:

| Configuration | #train. set | | | #test set | | | #val. set | | | DataConfig |
|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | C2 | C3 | C1 | C2 | C3 | C1 | C2 | C3 | |
| SR1 | 23 | 26 | 25 | 28 | 8 | 6 | 6 | 4 | 3 | CFA |
| SR2 | 10(180) | 7(190) | 9(200) | 2 | 2 | 2 | 2 | 2 | 2 | CFA |
| SR3 | 47 | 23 | 31 | 14 | 6 | 10 | 7 | 3 | 5 | CFB |
| SR4 | 46 | 23 | 29 | 14 | 6 | 8 | 7 | 3 | 4 | CFC |
| LR1 | 57 | 38 | 34 | 0/1 | 0/1 | 0/1 | - | - | - | CFA |

Table 5.3: Used Configurations for Radial Basis Training

- **Configuration**: This id of the configuration used;
- **#train. set**: The number of patterns in the training set;
- **#test set**: The number of patterns in the test set, each time only one of the three categories provide a pattern in this set;

- **#val. set**: The number of patterns in the validation set;

The configurations *SR1* and *SR2* use the data from data configuration *CFA* (section 5.1.3), where *SR2* uses the non-standard categorization where the water level of the storm situations in category *C1* is between 180 and 190 centimeter above NAP, the water level in *C2* is between 190 and 200 centimeter above NAP. The water level of the storms in *C3* is greater than 200 centimeter above NAP. This new categorization is used to make better predictions on the severe storm situations. The configurations *SR3* and *SR4* uses more patterns from category *C1* in the training set. This is done to increase the number of patterns in the learning phase of the network. The configuration *LR1* is used for training with leave-one-out (see section 4.3.4).

### 5.2.2 Standard radial basis training

The train- and test results using the configurations presented in 5.3 are shown in Table 5.4. In this overview only the results of training with standard radial basis are presented, this means training with a train-, test- and validation set. In this table we see that the standard deviation and root mean square error (section 4.2) are always better for the categories *C1* and *C2*.

| | Standard Deviation | | | | RMSE | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Configuration | C1 | C2 | C3 | all | C1 | C2 | C3 | all | Input | Epochs |
| SR1 | 17.33 | 12.26 | 21.56 | 20.87 | 18.60 | 11.68 | 25.88 | 21.16 | 18 | 38 |
| SR1 | 15.99 | 11.12 | 23.56 | 21.54 | 16.32 | 10.61 | 28.72 | 22.15 | 20 | 39 |
| SR1 | 18.00 | 12.27 | 24.99 | 22.97 | 18.40 | 11.69 | 30.27 | 23.67 | 22 | 42 |
| SR1 | 17.33 | 12.26 | 21.56 | 20.87 | 18.60 | 11.68 | 25.88 | 21.16 | 24 | 37 |
| SR1 | 10.78 | 11.10 | 21.70 | 19.09 | 11.37 | 11.17 | 27.11 | 20.25 | 26 | 35 |
| SR2 | 11.99 | 20.29 | 26.85 | 24.39 | 11.49 | 22.23 | 40.68 | 27.51 | 24 | 3 |
| SR3 | 31.17 | 390.66 | 385.90 | 326.29 | 29.97 | 396.80 | 399.39 | 336.50 | 14 | 58 |
| SR4 | 22.79 | 10.47 | 31.47 | 30.02 | 26.32 | 13.50 | 38.73 | 31.02 | 18 | 57 |

Table 5.4: Standard Radial Basis Training

### 5.2.3 Leave-one-out principle

The leave-one-out principle (section 4.3.4) is a method for training when one has a small amount of data. The results of training using this principle are presented in Table 5.5.

We used all available patterns in the training- and testing phase. So we are more likely to say that the predictions using a network which is trained with this technique is more stable on more different situations. In Table 5.5 we see that the

| Configuration | Standard Deviation | | | | RMSE | | | | Input |
|---|---|---|---|---|---|---|---|---|---|
| | C1 | C2 | C3 | all | C1 | C2 | C3 | all | |
| LR1 | 16.14 | 16.68 | 25.54 | 19.40 | 16.54 | 16.52 | 28.49 | 19.33 | 18 |
| LR1 | 17.44 | 17.46 | 21.46 | 19.71 | 17.84 | 17.18 | 23.87 | 19.67 | 20 |
| LR1 | 15.20 | 17.22 | 19.15 | 17.95 | 15.77 | 17.05 | 21.50 | 17.93 | 22 |
| LR1 | 15.67 | 17.31 | 19.47 | 18.26 | 16.04 | 17.52 | 22.03 | 18.33 | 24 |
| LR1 | 15.08 | 13.28 | 19.87 | 17.66 | 15.50 | 13.22 | 23.34 | 17.74 | 26 |

Table 5.5: Radial Basis Training using Leave-One-Out

standard deviation compared with the results when training with the standard radial basis technique from Table 5.4, in most cases is smaller. If we take a look at category *C3* then we see that the most optimal result is achieved when using 22 input nodes.

## 5.3 Feed-forward network

In this section we present the results for training with a feed-forward network using the backpropagation learning rule. In section 5.3.1 we discuss the different types of configurations used. In section 5.3.2 we present the results when training with standard backpropagation. And finally in section 5.3.3 we discuss the results of training using the leave-one-out technique.

### 5.3.1 Configuration

We used the configurations in Table 5.6 for training with feed-forward backpropagation. This table contains the same columns as described in section 5.2.1.
We used the following configuration id's:

- **SB1** the same configuration for leave-one-out and simple training using backpropagation.
- **SB2** more patterns from C3, C2 and C1;
- **SB3** Omit the wind data;
- **SB4** Omit the atmospheric pressure data;
- **SB5** Configuration based on CFA, but less data in it
- **SB6** Omit AukField Platform;

| Configuration | #train. set | | | #test set | | | #val. set | | | DataConfig |
|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | C2 | C3 | C1 | C2 | C3 | C1 | C2 | C3 | |
| SB1 | 23 | 26 | 25 | 28 | 8 | 6 | 6 | 4 | 3 | CFA |
| SB2 | 10(180) | 7(190) | 9(200) | 2 | 2 | 2 | 2 | 2 | 2 | CFA |
| SB3 | 47 | 23 | 31 | 14 | 6 | 10 | 7 | 3 | 5 | CFB |
| SB4 | 46 | 23 | 29 | 14 | 6 | 8 | 7 | 3 | 4 | CFC |
| SB5 | 49 | 23 | 26 | 14 | 6 | 8 | 7 | 3 | 4 | CFD |
| SB6 | 66 | 26 | 31 | 16 | 8 | 10 | 8 | 4 | 5 | CFE |
| BL1 | 43 | 26 | 25 | 0/1 | 0/1 | 0/1 | - | - | - | CFA |
| BL2 | 49 | 37 | 22 | 0/1 | 0/1 | 0/1 | - | - | - | CFA |
| BL3 | 26 | 29 | 24 | 0/1 | 0/1 | 0/1 | - | - | - | CFA |
| BL4 | 57 | 38 | 34 | 0/1 | 0/1 | 0/1 | - | - | - | CFA |
| BL5 | 14(180) | 11(190) | 13(200) | 0/1 | 0/1 | 0/1 | - | - | - | CFA |

Table 5.6: Configurations for Backpropagation Training

- **BL1** the same configuration for leave-one-out and simple training using backpropagation.
- **BL2** more patterns from C3, C2 and C1;
- **BL3** all patterns from C3, but some patterns from C2 and C1 are left out;
- **BL4** all available patterns are in this configuration;
- **BL5** only train on situations greater than 180 cm above NAP;

### 5.3.2 Standard backpropagation training

After various trials we discovered that a three-layer network with 18-24 input units and hidden layer one with 14-20 units, hidden layer two with 4-6 layers performs best. So we focussed on networks with such architecture. In Table 5.7 we present the results when training with standard backpropagation.

Configuration *SB1* worked out very good for the predicting of storms, specially the architecture with 24 input nodes, 15 units in hidden layer one and 6 units in hidden layer two. This network can predict storms from category *C3* with a standard deviation of 10.75. Because for this test we had to split the whole data set into a train-, test- and validation-part the prediction on storms in category *C3* consists of 6 patterns. So we tested this architecture with the leave-one-out technique, in order to create a more stable network.

### 5.3.3 Leave-one-out principle

In this section we present the results of the research that was done with a feed-forward network using backpropagation and the leave one out technique (section

| Configuration | Standard Deviation | | | | RMSE | | | | Epochs | Nodes Used in Layer | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | C2 | C3 | all | C1 | C2 | C3 | all | | Input | One | Two |
| SB1 | 15.39 | 20.32 | 24.22 | 20.21 | 19.95 | 15.05 | 29.76 | 20.18 | 7 | 18 | 14 | 6 |
| SB1 | 19.29 | 23.32 | 21.36 | 21.74 | 23.28 | 20.84 | 22.17 | 21.96 | 6 | 18 | 15 | 5 |
| SB1 | 14.70 | 19.60 | 23.14 | 18.48 | 19.05 | 14.40 | 24.82 | 18.34 | 7 | 18 | 17 | 5 |
| SB1 | 13.80 | 6.38 | 27.94 | 18.71 | 5.420 | 25.29 | 28.98 | 24.62 | 3 | 20 | 14 | 6 |
| SB1 | 17.08 | 18.75 | 19.53 | 21.41 | 16.97 | 19.02 | 29.10 | 20.90 | 13 | 20 | 15 | 8 |
| SB1 | 19.36 | 12.90 | 36.69 | 23.19 | 15.51 | 21.21 | 42.51 | 23.30 | 13 | 20 | 18 | 4 |
| SB1 | 17.13 | 7.66 | 39.54 | 25.41 | 15.63 | 16.91 | 48.65 | 25.93 | 5 | 20 | 19 | 4 |
| SB1 | 15.33 | 2.89 | 38.93 | 19.30 | 20.41 | 15.39 | 40.27 | 19.07 | 9 | 20 | 20 | 6 |
| SB1 | 17.83 | 18.67 | 34.80 | 22.36 | 19.07 | 20.29 | 33.36 | 22.89 | 13 | 22 | 14 | 5 |
| SB1 | 19.91 | 27.00 | 29.84 | 24.71 | 30.06 | 23.10 | 28.20 | 26.65 | 8 | 22 | 14 | 6 |
| SB1 | 24.25 | 25.19 | 29.46 | 25.81 | 24.46 | 24.38 | 30.08 | 25.55 | 11 | 22 | 15 | 7 |
| SB1 | 17.46 | 13.68 | 29.57 | 21.23 | 13.33 | 19.48 | 33.16 | 21.02 | 10 | 22 | 19 | 6 |
| SB1 | 21.71 | 12.84 | 49.43 | 30.84 | 13.77 | 20.99 | 59.42 | 31.14 | 4 | 24 | 14 | 6 |
| SB1 | 9.38 | 4.97 | 18.84 | 14.94 | 11.54 | 20.17 | 27.70 | 20.29 | 6 | 24 | 14 | 8 |
| SB1 | 13.09 | 14.34 | 10.75 | 14.91 | 13.41 | 15.39 | 14.15 | 14.71 | 7 | 24 | 15 | 6 |
| SB1 | 13.39 | 15.05 | 17.57 | 14.60 | 14.56 | 13.27 | 21.89 | 15.57 | 13 | 24 | 15 | 8 |
| SB1 | 7.90 | 6.11 | 33.81 | 17.25 | 12.06 | 9.53 | 32.44 | 16.94 | 10 | 24 | 16 | 4 |
| SB1 | 26.81 | 13.55 | 28.60 | 29.99 | 12.16 | 34.42 | 34.19 | 30.57 | 3 | 24 | 16 | 6 |
| SB1 | 7.14 | 18.63 | 37.91 | 19.52 | 16.81 | 16.75 | 33.05 | 20.87 | 4 | 24 | 16 | 7 |
| SB1 | 16.03 | 4.70 | 26.30 | 18.99 | 16.34 | 16.49 | 41.35 | 18.83 | 12 | 24 | 18 | 8 |
| SB1 | 14.29 | 16.88 | 21.09 | 19.75 | 19.53 | 13.77 | 34.91 | 20.77 | 4 | 24 | 20 | 5 |
| SB1 | 11.16 | 8.38 | 16.49 | 15.22 | 19.63 | 11.12 | 31.48 | 18.74 | 13 | 24 | 20 | 5 |
| SB1 | 10.74 | 25.75 | 27.24 | 21.60 | 27.15 | 15.16 | 27.72 | 22.47 | 8 | 26 | 15 | 4 |
| SB1 | 17.93 | 15.59 | 28.97 | 20.88 | 15.31 | 20.66 | 29.57 | 21.03 | 9 | 26 | 15 | 7 |
| SB1 | 17.08 | 19.40 | 25.74 | 23.43 | 19.49 | 19.37 | 35.68 | 23.27 | 4 | 26 | 17 | 8 |
| SB2 | 12.29 | 7.10 | 29.76 | 19.21 | 14.92 | 13.97 | 31.84 | 18.91 | 6 | 24 | 14 | 5 |
| SB2 | 20.65 | 7.78 | 19.82 | 19.50 | 11.07 | 20.26 | 23.49 | 19.20 | 11 | 24 | 16 | 4 |
| SB2 | 21.43 | 7.06 | 18.18 | 21.12 | 16.01 | 20.71 | 29.11 | 21.62 | 13 | 24 | 17 | 5 |
| SB3 | 19.28 | 21.92 | 87.58 | 40.93 | 45.05 | 25.40 | 25.69 | 90.23 | 9 | 14 | 11 | 4 |
| SB3 | 16.30 | 24.30 | 44.28 | 27.01 | 23.60 | 19.19 | 44.07 | 26.89 | 13 | 24 | 13 | 6 |
| SB3 | 13.20 | 16.30 | 35.20 | 21.96 | 17.02 | 16.45 | 37.55 | 22.06 | 7 | 14 | 17 | 8 |
| SB3 | 18.69 | 18.64 | 36.42 | 24.10 | 18.76 | 22.30 | 36.41 | 24.48 | 8 | 14 | 18 | 8 |
| SB4 | 14.21 | 18.13 | 32.49 | 26.80 | 17.68 | 18.01 | 49.16 | 26.54 | 4 | 18 | 12 | 8 |
| SB4 | 19.93 | 21.24 | 31.77 | 27.20 | 20.69 | 23.84 | 41.06 | 26.92 | 6 | 18 | 14 | 4 |
| SB4 | 20.39 | 17.14 | 38.93 | 30.04 | 16.66 | 24.31 | 52.71 | 29.73 | 4 | 18 | 15 | 8 |
| SB4 | 15.70 | 24.92 | 35.31 | 28.87 | 24.32 | 19.47 | 48.57 | 28.57 | 4 | 18 | 15 | 6 |
| SB5 | 15.77 | 20.52 | 22.92 | 19.44 | 21.05 | 21.03 | 21.66 | 21.15 | 7 | 20 | 15 | 4 |
| SB5 | 15.57 | 14.03 | 21.15 | 17.00 | 13.62 | 15.94 | 23.25 | 16.82 | 6 | 20 | 16 | 6 |
| SB5 | 19.89 | 20.81 | 25.17 | 22.37 | 20.19 | 22.52 | 25.93 | 22.43 | 5 | 20 | 18 | 4 |
| SB5 | 18.90 | 19.15 | 30.76 | 22.70 | 20.35 | 20.24 | 31.94 | 22.88 | 6 | 20 | 20 | 8 |
| SB6 | 18.86 | 25.49 | 24.73 | 22.04 | 24.82 | 18.57 | 23.35 | 21.81 | 12 | 24 | 13 | 7 |
| SB6 | 13.58 | 19.99 | 36.55 | 21.07 | 20.18 | 14.06 | 34.46 | 21.27 | 11 | 24 | 15 | 5 |
| SB6 | 13.07 | 17.04 | 26.33 | 18.14 | 16.92 | 12.81 | 30.41 | 18.61 | 5 | 24 | 17 | 7 |
| SB6 | 21.23 | 19.37 | 29.01 | 21.75 | 19.66 | 21.09 | 27.56 | 21.96 | 9 | 24 | 15 | 8 |
| SB6 | 14.01 | 14.04 | 21.79 | 15.38 | 14.43 | 14.33 | 21.91 | 16.02 | 4 | 24 | 19 | 5 |

Table 5.7: Training Results using Standard Backpropagation

4.3.4). Results presented in section 5.3.2 suggest a three layer network with 24 input units, 15 units in hidden layer one, 6 units in hidden layer two and one output unit. In Table 5.8 we present these results. We used different data sets to train this network (in Table 5.6 we give more detail about this contents of the different data sets). In this table we see that the best results are obtained if we train with all the 129 available patterns.

### 5.3.4 Bagging, bumping and balancing

In this section we discuss the results when applying balancing, bagging and bumping (section 4.3.5) together with a feed-forward backpropagation network on the data set. These routines where developed at Leiden University. In Table 5.9 we present the average results when training with 100 networks, repeated 50 times, because of the exactness of the predictions. The whole data set is divided into

| Configuration | Standard Deviation | | | | RMSE | | | | Bias | Epochs |
|---|---|---|---|---|---|---|---|---|---|---|
| | C1 | C2 | C3 | all | C1 | C2 | C3 | all | | |
| BL1 | 17.46 | 19.48 | 25.06 | 21.77 | 18.61 | 18.99 | 27.26 | 21.64 | 0.56 | 7 |
| BL1 | 18.25 | 18.62 | 25.29 | 21.71 | 19.14 | 18.20 | 27.31 | 21.59 | 0.36 | 6 |
| BL2 | 17.45 | 18.14 | 23.56 | 20.85 | 18.46 | 17.79 | 25.40 | 20.76 | 0.42 | 7 |
| BL3 | 19.37 | 22.74 | 27.32 | 22.36 | 19.15 | 21.94 | 27.10 | 22.13 | 0.93 | 6 |
| BL3 | 19.70 | 21.98 | 29.84 | 25.58 | 20.54 | 21.55 | 29.53 | 25.42 | -0.51 | 7 |
| BL2 | 17.83 | 17.76 | 23.90 | 20.97 | 18.78 | 17.41 | 25.56 | 20.88 | 0.26 | 8 |
| BL4 | 18.37 | 14.99 | 15.01 | 18.12 | 20.01 | 14.74 | 17.27 | 18.10 | -1.38 | 6 |
| BL4 | 17.62 | 16.41 | 15.83 | 18.39 | 19.71 | 16.13 | 17.90 | 18.40 | -1.72 | 7 |
| BL4 | 19.24 | 17.44 | 14.18 | 18.88 | 20.43 | 17.49 | 17.03 | 18.81 | 0.22 | 8 |
| BL5 | 30.69 | 25.57 | 34.98 | 33.96 | 29.61 | 29.28 | 40.37 | 33.60 | 2.42 | 6 |
| BL5 | 28.17 | 28.98 | 23.97 | 29.36 | 28.04 | 29.37 | 29.65 | 28.99 | 0.91 | 7 |
| BL5 | 30.69 | 25.57 | 34.98 | 33.96 | 29.61 | 29.28 | 40.37 | 33.60 | 2.42 | 8 |

Table 5.8: Feed-forward Network using Leave-One-Out

a test- (25 patterns), and train part (104 patterns for training and validation). Please note that the data set is not compressed with Principal Component Analysis.

| RMSE | Bagging | | Bumping | | Balancing | |
|---|---|---|---|---|---|---|
| | RMSE | % decrease | RMSE | % decrease | RMSE | % decrease |
| 21.819 | 20.391 | 6.87% | 17.701 | 18.87% | 17.556 | 19.54% |

Table 5.9: Average of 50 training phases using 100 networks for Bagging, Bumping and Balancing

In Table 5.9 we compare the following four methods for combining the network outputs:

- **standard training**, training with the standard feed-forward backpropagation rule with no special boot strapping technique;
- **bagging** the average of all predictions of the 100 networks which will lead to one prediction;
- **bumping** The network with the best predictions (smallest RMSE) will be chosen out of the 100 trained networks;
- **balancing** A weight will be added to each of the 100 trained networks, with these weights we compute the weighted average of the predictions, the predictions are the weighted average of the 100 networks.

Note: The %-sign in this table represents the decrease of the generalization error, using the specified technique.

| | C1 | | C2 | | C3 | |
|---|---|---|---|---|---|---|
| Std.dev. | Std.dev. | RMSE | Std.dev. | RMSE | Std.dev. | RMSE |
| 19.485 | 20.043 | 12.129 | 16.908 | 12.819 | 20.352 | 27.244 |

Table 5.10: Categorized Average of 50 training phases using 100 networks for Balancing

In table 5.10 we split the results for balancing into the three previously defined categories *C1*, *C2* and *C3* (section 3.5. We see that the predictions in the category with severe storms (*C3*) are less better than expected from the result on the whole test set.

## 5.4 Problems

The data set contains continuous information from the period January 1990 until March 1996. It is for the Storm Waring Service very important to predict the severe storm situations with a great exactness. Not many severe storm situations are available in the data set, therefore we had to apply the categorization on the data set. The category *C3* does include the severe storms, but also less severe storms which are not very important for the prediction mechanism. Besides categorization we applied leave-one-out to the neural networks, with this method we can say more about the exactness of the predictions.

## 5.5 Conclusions

If we compare the results from Radial basis training (Table 5.4 and 5.5) with training with a feed-forward backpropagation network (Table 5.7 and 5.8) we see that training with a multi layer feed-forward network performs better on storms of category *C3*. The results of training with a feed-forward network with 24 input units, 15 units in hidden layer one and 6 units in hidden layer one are better than training with the same architecture and using leave-one-out. The results from training with leave-one-out are less better because it uses all patterns from the data set. On the contrary, standard backpropagation training splits the data set into a train-, test- and validation part. If we take a look at training with feed-forward backpropagation using bagging, bumping and balancing, we also see good results using balancing and bagging. But if we categorize these results in the

three defined categories *C1*, *C2* and *C3* we see that the network predicts well on medium storms, but the prediction on the important severe storms is worser than the results from predicting with feed-forward backpropagation (section 5.3.3).

# Chapter 6

# Project evaluation and conclusion

In this chapter we evaluate the different aspects of this project. In section 6.1 we discuss the working environment at RIKZ and in section 6.2 we discuss the development of the neural network. After that we evaluate in section 6.3 the results presented in chapter 5. Finally we discuss future research in section 6.4.

## 6.1 Working environment

As discussed in section 1.2 the research was done at the research department of the department Information, Technology & Systems (ITS) at the National Institute for Coastal and Marine Management, in The Hague. This department is a great environment to do research because one gets the freedom to choose the direction in which the project is going to. The available computer infrastructure can be used very well for performing a project like this. We learned many things about working in a professional research environment.

## 6.2 Data selection and network development

The database which we used in this projects contains only data from storm situations in the period January 1990 until March 1996 (section 3.5). In this period only a few severe storms occurred, therefore we had to enlarge the data set with less severe storms, which are not that important for the prediction of the water level during severe storm situations. To keep track on the heaviness of the storms we introduced a categorization (section 5.1.2). With this categorization we can give a more exact picture of the storm predictions. To improve the quality of the predictions we also used leave-one-out (section 4.3.4) and principal component analysis (section 4.1). In general we can say that we used techniques to 'recover' from the lack of severe storm patterns in the neural network input set. As we look at the results, which are presented chapter 5, we see that it is very important how many patterns we use in the input set of the neural network. We can

conclude that the number of patterns in the input set are a very important factor for the improvement of the quality of the predictions. Specially the patterns from severe storms are interesting because more than half of the input patterns is filled with medium storm situations (category *C1* and *C2* which are described in the categorization of section 5.1.2).

The training and simulation software of the bagging, bumping and balancing method which is used in this project is developed at Leiden University The other neural network software is written at the National Institute for Coastal and Marine Management using the Matlab package (section 4.4). The use of Matlab is very intuitive, the commands are very well described and are extendable. The usage of the developed software is described in[23]

## 6.3 Network results

It has been shown that training with a feed-forward network is functioning well. Because of the lack of severe storm patterns we used leave-one-out for a better estimate of the quality of the results. When training with this technique we use all patterns in the data set as a test pattern, so every resulting water level of a storm situation is compared with the predicted water level. The reliability of training with leave-one-out is more secure. With this technique we reach a water level prediction for the storms of category *C3* with a standard deviation of 14.18 centimeter and a Root Mean Squared Error of 17.03 centimeter. Despite of the good results previously reached with bumping, bagging and and balancing the results of training with these three methods did not give a better result than training with leave-one-out. The best results are reached when training with balancing: a standard deviation of 20.352 centimeter on severe storms.
If we compare the results for training with leave-one-out with the predictions which are done by the current prediction methods, for approximately the same period (Table 1.4), we see that the standard deviation of the neural network is considerable smaller than the Storm Warning Service (17.2 centimeter), the Royal Dutch Meteorological Institute (19.8 centimeter) and the Continental Shelf Model (24.6). We can conclude that the predictions done by the neural network are more exact than the predictions done by current prediction methods.

However, we have to be careful in drawing this conclusion. First the network predictions are done with real wind data, and not with predicted wind data as in a real situation. Second, the database does not contain many severe storm situations and probably not all types of storms are covered in the database. If more storm situations become available, it will be advisable to retrain the network. Lastly, the data base contains reconstructed values from several measuring locations (section 3.7), in an on-line situation, a reconstruction of the data will

be a very intensive process.

## 6.4 Future research

To increase the quality of the predictions it is necessary to gather more storms before the year 1990, it can also be useful to fill the gaps in the data such that there are more storm situations in the database.

Because of the lack of predicted meteorological data, the networks in this project uses only real wind data. In the future it is worthwhile to do research with predicted meteorological data, because this method is used in real, on-line storm predictions.

The research on bagging, bumping and balancing is done in a short period of time. It seems to be promising to combine these three methods and specially balancing with leave-one-out and principal component analysis. The results when training on the uncompressed data set are such that further research is recommended.

For the usability of this storm prediction model it is useful to design a connection between the real time data base with water levels, the database of the Dutch Royal Meteorological Institute which contains meteorological data and the storm prediction system. The predictions described in this thesis are only done with past storm situations.
In this project we also used reconstructed data. If this reconstruction can be done on-line (during a storm-situation) then a connection between a neural network in an operational environment can be realized.

# Bibliography


[1] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[2] G. Cybenko. *Continuous Valued Neural Networks with Two Hidden Layers Are Sufficient. Technical Report.* Department of Computer Science, Tufts university, 1988.

[3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signal and Systems*, 2:303–314, 1989.

[4] B Efron and R. Tibshirani. *An Introduction to the Bootstrap.* Chapman & Hall, London, 1993.

[5] C.R. Greenwalt and M. E. Shultz. *Principals of Error Theory and Cartographic Applications.* Aeronautical Chart and Information Center Technical Report No. 96, 1962.

[6] N.S. Heaps. Storm surges. *Oceanogr. Marine Biology, Ann. Rev.*, 5:11–58, 1967.

[7] P.F. Heinen. *Singuliere Spectrum Analyse toegepast op tijdreeksen, rapport DGW-92.031.* Ministerie van Verkeer en Waterstaat, 1992.

[8] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the theory of Neural Computation.* Addison-Wesley, 1991.

[9] T. Heskes. *Balancing between bagging and bumping.* RWCP Novel Functions SNN Laboratory, University of Nijmegen, Nijmegen, 1997.

[10] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educ. Psychol.*, 27:4171–441, 1933.

[11] I.T. Jolliffe. *Principal Component Analysis.* Springer Verlag, 1986.

[12] Levenberg K. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[13] K. Karhunen. *Uber Lineare Methoden in der Wahrscheinlichkeitsrechnung, A137.* Ann. Acad. Sci. Fennicae, 1947.

[14] D. Marguardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.

[15] J. Moody and C. Darken. Learning with localized receptive fields. *Proceedings of the 1988 Connectionist Models Summer School*, pages 133–143, 1988.

[16] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1:281–294, 1989.

[17] J.J. More. The levenberg-marquardt algorithm: implementation and theory. *Lecture Notes in Mathematics*, 630:105–116, 1977.

[18] D.M. Skapura. *Building Neural Networks*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1996.

[19] P. Sollich and A. Krogh. Learning with ensembles: How over-fitting can be useful. *Advances in Neural Information Processing Systems*, 8:190–196, 1996.

[20] R Tibshirani and K. Knight. *Model search and inference by bootstrap "bumping"*. University of Toronto, Toronto, 1995.

[21] RijksInstituut voor Kust en Zee. *Gemiddelde Getijkromme*. SDU, 1994.

[22] RijksInstituut voor Kust en Zee. *Getijtafels voor Nederland 1997*. SDU, 1996.

[23] M.C. van de Weg. *Installatie & Gebruik Stormnet*. Rijksuniversiteit Leiden / RijksInstituut voor Kust en Zee, 1997.

[24] D. Wolpert and W. Macready. *Combining stacking with bagging to improve a learning algorithm. Technical report*. Santa Fe Institute, 1996.

# Appendix A: Translation Dictionary

| *English* | *Dutch* |
|---|---|
| residual | opzet |
| astronomical water level | astronomische water stand |
| Directorate-General of Public Works and Water Management | Rijkswaterstaat (RWS) |
| National Institute for Coastal and Marine Management | RijksInstituut voor Kust en Zee (RIKZ) |
| Principal Component Analysis (PCA) | hoofdcomponenten analyse |
| Royal Meteorological institute | Koninlijk Meteorologisch Instituut (KNMI) |
| Storm Warning Service | Stormvloed Waarschuwings Dienst (SVSD) |

# Appendix B: Used Configurations

## CFA:

```
#INPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 1 0 0 0 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 1 0 0 0 34 38 42
nts ../stormnet/data/storm.nts 1 0 0 0 28 34 42
low ../stormnet/data/storm.low 1 0 0 0 34 38 42
epf ../stormnet/data/storm.epf 1 0 0 0 34 38 42
k13 ../stormnet/data/storm.k13 1 0 0 0 34 38 42
hvh ../stormnet/data/storm.hvh 0 1 0 0 26 28 30 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 0 1 0 0 30 32 34 36 38 40 42
nts ../stormnet/data/storm.nts 0 1 0 0 24 28 32 38 42
low ../stormnet/data/storm.low 0 1 0 0 30 34 38 42
hvh ../stormnet/data/storm.hvh 0 0 1 0 28 34 38 42
auk ../stormnet/data/storm.auk 0 0 1 0 26 36 44 54
k13 ../stormnet/data/storm.k13 0 0 1 0 34 44 54
vzu ../stormnet/data/storm.vzu 0 0 1 0 36 42 50
auk ../stormnet/data/storm.auk 0 0 0 1 28 36 60
k13 ../stormnet/data/storm.k13 0 0 0 1 34 42 54 68
epf ../stormnet/data/storm.epf 0 0 0 1 38 42
hvh ../stormnet/data/storm.hvh 0 0 0 1 34 36 38 42
vzu ../stormnet/data/storm.vzu 0 0 0 1 36 40 42 50
#OUTPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 0 1 0 0 48
```

## CFB:

```
#INPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 1 0 0 0 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 1 0 0 0 34 38 42
nts ../stormnet/data/storm.nts 1 0 0 0 28 34 42
low ../stormnet/data/storm.low 1 0 0 0 34 38 42
```

```
epf ../stormnet/data/storm.epf 1 0 0 0 34 38 42
k13 ../stormnet/data/storm.k13 1 0 0 0 34 38 42
hvh ../stormnet/data/storm.hvh 0 1 0 0 26 28 30 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 0 1 0 0 30 32 34 36 38 40 42
nts ../stormnet/data/storm.nts 0 1 0 0 24 28 32 38 42
low ../stormnet/data/storm.low 0 1 0 0 30 34 38 42
hvh ../stormnet/data/storm.hvh 0 0 1 0 28 34 38 42
auk ../stormnet/data/storm.auk 0 0 1 0 26 36 44 54
k13 ../stormnet/data/storm.k13 0 0 1 0 34 44 54
vzu ../stormnet/data/storm.vzu 0 0 1 0 36 42 50
#OUTPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 0 1 0 0 48
```

## CFC:

```
#INPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 1 0 0 0 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 1 0 0 0 34 38 42
nts ../stormnet/data/storm.nts 1 0 0 0 28 34 42
low ../stormnet/data/storm.low 1 0 0 0 34 38 42
epf ../stormnet/data/storm.epf 1 0 0 0 34 38 42
k13 ../stormnet/data/storm.k13 1 0 0 0 34 38 42
hvh ../stormnet/data/storm.hvh 0 1 0 0 26 28 30 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 0 1 0 0 30 32 34 36 38 40 42
nts ../stormnet/data/storm.nts 0 1 0 0 24 28 32 38 42
low ../stormnet/data/storm.low 0 1 0 0 30 34 38 42
auk ../stormnet/data/storm.auk 0 0 0 1 28 36 60
k13 ../stormnet/data/storm.k13 0 0 0 1 34 42 54 68
epf ../stormnet/data/storm.epf 0 0 0 1 38 42
hvh ../stormnet/data/storm.hvh 0 0 0 1 34 36 38 42
vzu ../stormnet/data/storm.vzu 0 0 0 1 36 40 42 50
#OUTPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 0 1 0 0 48
```

## CFD:

```
#INPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 1 0 0 0 34 40 42
vli ../stormnet/data/storm.vli 1 0 0 0 38 42
nts ../stormnet/data/storm.nts 1 0 0 0 28 34 42
```

```
low ../stormnet/data/storm.low 1 0 0 0 38 42
epf ../stormnet/data/storm.epf 1 0 0 0 38 42
k13 ../stormnet/data/storm.k13 1 0 0 0 38 42
hvh ../stormnet/data/storm.hvh 0 1 0 0 30 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 0 1 0 0 30 32 34 36 38 40 42
nts ../stormnet/data/storm.nts 0 1 0 0 28 32 38 42
low ../stormnet/data/storm.low 0 1 0 0 30 34 42
hvh ../stormnet/data/storm.hvh 0 0 1 0 28 34 42
auk ../stormnet/data/storm.auk 0 0 1 0 26 36 44 54
k13 ../stormnet/data/storm.k13 0 0 1 0 34 44 54
vzu ../stormnet/data/storm.vzu 0 0 1 0 42 50
auk ../stormnet/data/storm.auk 0 0 0 1 28 36 60
k13 ../stormnet/data/storm.k13 0 0 0 1 34 42 54 68
epf ../stormnet/data/storm.epf 0 0 0 1 38 42
hvh ../stormnet/data/storm.hvh 0 0 0 1 34 38 42
vzu ../stormnet/data/storm.vzu 0 0 0 1 36 42 50
#OUTPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 0 1 0 0 48
```

## CFE:

```
#INPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 1 0 0 0 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 1 0 0 0 34 38 42
low ../stormnet/data/storm.low 1 0 0 0 34 38 42
epf ../stormnet/data/storm.epf 1 0 0 0 34 38 42
k13 ../stormnet/data/storm.k13 1 0 0 0 34 38 42
hvh ../stormnet/data/storm.hvh 0 1 0 0 26 28 30 32 34 36 38 40 42
vli ../stormnet/data/storm.vli 0 1 0 0 30 32 34 36 38 40 42
low ../stormnet/data/storm.low 0 1 0 0 30 34 38 42
hvh ../stormnet/data/storm.hvh 0 0 1 0 28 34 38 42
auk ../stormnet/data/storm.auk 0 0 1 0 26 36 44 54
k13 ../stormnet/data/storm.k13 0 0 1 0 34 44 54
vzu ../stormnet/data/storm.vzu 0 0 1 0 36 42 50
auk ../stormnet/data/storm.auk 0 0 0 1 28 36 60
k13 ../stormnet/data/storm.k13 0 0 0 1 34 42 54 68
epf ../stormnet/data/storm.epf 0 0 0 1 38 42
hvh ../stormnet/data/storm.hvh 0 0 0 1 34 36 38 42
vzu ../stormnet/data/storm.vzu 0 0 0 1 36 40 42 50
#OUTPUT VALUES
#LOC FILE ASTRO REAL PRESS WIND HOURS
hvh ../stormnet/data/storm.hvh 0 1 0 0 48
```

# Index