Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Applying data-assimilation and calibration in the field of urban drainage

A thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

DANIEL MULDER

Delft, the Netherlands
April 2014

MSc THESIS APPLIED MATHEMATICS

"Applying data-assimilation and calibration in the field of urban drainage"

Daniel Mulder

Delft University of Technology

Daily supervisor	**Responsible professor**
Dr. C. van Velzen	Prof.dr.ir. A.W. Heemink
Other thesis committee members	
Dr.ir. J.L. Korving	Prof.dr.ir. F.H.L.R. Clemens
Dr. P. Wilders	

April 2014 Delft, the Netherlands

Contents

1 Introduction 1
1.1 The problem . . . 2
1.1.1 General handling of the problem . . . 2
1.1.2 Specified goals . . . 3
1.2 Thesis outline . . . 5

2 Data-assimilation 7
2.1 The basic model . . . 7
2.1.1 Analysis maximum likelihood . . . 8
2.2 The Kalman filter . . . 10
2.2.1 Kalman filter equations . . . 10
2.3 The Ensemble Kalman filter (EnKF) . . . 11
2.3.1 Terms and notation . . . 11
2.3.2 Ensemble Kalman filter equations . . . 12
2.4 DuD . . . 13
2.5 Identifiability analysis . . . 15

3 Model characteristics 17
3.1 Sobek . . . 17
3.1.1 About Sobek . . . 17
3.1.2 Conceptual description . . . 18
3.1.3 Mathematical core . . . 21
3.2 OpenDA . . . 22
3.2.1 Available methods . . . 22
3.2.2 Blackbox setup . . . 25
3.3 OpenDA and Sobek coupling . . . 26
3.3.1 Basic coupling setup . . . 26
3.3.2 Restarting . . . 28
3.3.3 Sobek restart flaws . . . 28
3.4 The network of Delft . . . 31

4 Calibration 33
4.1 Surface areas . . . 34
4.2 Rainfall runoff . . . 48
4.3 Parameter analysis . . . 53

5 Assimilation 59
5.1 Model uncertainty . . . 59
5.1.1 Model state . . . 59
5.1.2 AR(1) process . . . 59

5.1.3 Noise modelling methods 60
5.2 Precipitation 64
5.3 Number of ensemble members 65
5.4 Observation locations 69
5.5 Assimilation results 73
5.5.1 Non-perturbed initial assimilation 73
5.5.2 Sediment levels 77

6 Conclusions and future research 88
6.1 Summary of resolved Sobek blackbox issues 88
6.2 Summary of blackbox results and conclusions 89
6.3 Conclusions 90
6.4 Future research 91

A Noise model implementation methods in OpenDA 94

B Old observations 96

C Bugged blackbox coupling 98

D Modelling blockages through orifices 101

List of Figures

1.1 Data-assimilation example 3
1.2 Process scheme 5

3.1 A simple sewer model 18
3.2 Rainfall runoff concept figure 19
3.3 Visualization of the different types of model wrappers 24
3.4 Map with observation locations 28
3.5 Restart error figures 29
3.6 Graph of the suction side of a pump (error) 30
3.7 Delft full sewer network 31

4.1 Surface water nodes (partial) 36
4.2 Surface 5 calibration 37
4.3 Internal weirs and reference nodes 38
4.4 Calibration of a single surface area parameter (1) 39
4.5 Surfaces 5 and 7 calibration 41
4.6 Calibration of two surface area parameters (1) 42
4.7 Calibration of two surface area parameters (2) 43
4.8 Surface water nodes map (extended version) 44
4.9 Surface 5 calibration 45
4.10 Calibration of one surface area parameter, modified by an amount . . 45
4.11 Calibration of two surface area parameters 46
4.12 Surfaces 5 and 7 calibration 47
4.13 Calibration of two surface parameters, modified by a factor 48
4.14 Calibration of two runoff parameters. 50
4.15 Delay in runoff calibration for two parameters 51
4.16 Delay in runoff calibration for four parameters 53
4.17 Parameter correlation matrix 55
4.18 Columns of the matrix V 56
4.19 Correlation of 8 parameters 56

5.1 Different noise on each point 62
5.2 Noise distributed among blocks 63
5.3 Interpolated noise values 63
5.4 Precipitation used in most simulations 64
5.5 Ensemble standard deviation at time = 30 minutes 66
5.6 Ensemble standard deviation at time = 60 minutes 67
5.7 Standard deviation difference for ensemble sizes of 100 and 20 68
5.8 Example of a node and correlated nodes within a radius of 500 metres 70
5.9 Points that provide a reasonable amount of information 72
5.10 The observation locations 73

5.11 Waterlevel results in one of the observation locations 75
5.12 Average absolute difference of ensemble mean and analysed state . . . 76
5.13 Nodes at which flooding occurs (1) 77
5.14 Location of the general reach type 'Round 600 mm' 79
5.15 The sediment level in the round 600 mm reaches 80
5.16 Waterlevel results in two of the observation locations 81
5.17 Average absolute difference of ensemble mean and analysed state . . . 82
5.18 Nodes at which flooding occurs (2) 83
5.19 Location of a single reach that is added to the state vector 84
5.20 Sediment results for an individual reach 84
5.21 Waterlevel results in two of the observation locations 86
5.22 Nodes at which flooding occurs (3) 87

B.1 Old observation locations 96

C.1 Bugged blackbox results (1) 98
C.2 Bugged blackbox results (1) 99
C.3 Bugged blackbox results (1) 100

D.1 Network with orifices 102
D.2 Orifices locations enlarged 102
D.3 V matrix for the orifices (1) 103
D.4 V matrix for the orifices (2) 104
D.5 V matrix for the orifices (3) 105
D.6 V matrix for the orifices (4) 106

List of Tables

3.1 Rainfall Runoff default parameters, rational method 20
3.2 Rainfall Runoff default parameters, Horton equation 21
3.3 Details of the model of Delft 32

4.1 Total surface areas in the model of Delft 34
4.2 Single surface type calibration (simple case, fixed amount) 37
4.3 Two surface types calibration (simple case, fixed amount) 39
4.4 Two surface types calibration (simple case, factor) 42
4.5 Single surface type calibration (full case, fixed amount) 44
4.6 Two surface types calibration (full case, fixed amount) 46
4.7 Two surface types calibration (full case, factor) 48
4.8 Runoff parameters and surface areas 49
4.9 Calibration of a single runoff parameter (flat open paved) 49
4.10 Calibration of a single runoff parameter (sloped roof) 49
4.11 Calibration of two runoff parameters 50
4.12 Calibration of four runoff parameters 52

5.1 Setup of the assimilation for the first twin experiment 74
5.2 Twin experiment setup for a reach (1) 78
5.3 Twin experiment setup for a reach (2) 83

Chapter 1

Introduction

Urban drainage systems or sewer systems are an important part of society. These systems have two major functions; prevention of flooding of urban areas and improving public health by means of a decrease in exposure to faecal contamination. Both of these goals are accomplished by transporting waste water and storm water to either the surface water or a water treatment plant. So it seems like an excellent idea to invest in improving the sewer systems. However, developing and maintaining such a sewer system will be, like a lot of things, a compromise between the economic and health benefits, and the costs involved.

A good tool in the development and maintenance of sewer systems is the modelling of such systems. There are already several hydrodynamic software packages around that can do this job, for example Sobek or Infoworks. The problem with modelling sewer systems is that it often concerns a simplified version of the real system. A possible consequence of simplifying a real world process for computer simulations is that some influential factors or processes might be omitted. This obviously creates differences in the model output compared to the real system. Another source of difference are the numerical errors that appear when dealing with mathematical models. The latter is not a result from the procedure of simplifying a real world system to a model, but rather the way this model is solved using a certain numerical scheme or software package.

The use of hydrodynamic models has been prevalent in the field of urban drainage systems. Partially due to the fact that there was very little monitoring data available and the seemingly low resource cost of hydrodynamic modelling. Recently, better and cheaper ways, through more reliable and cost-efficient monitoring equipment, have been developed to acquire monitoring data for urban drainage systems. The results show that monitoring systems might be the most reliable way of acquiring information regarding urban drainage systems [Veldhuis and Tait, 2011]

Any decisions regarding maintenance and modification of sewer systems depend mainly on the pipe age and on a first hand look in the actual sewer system by using a moving camera. However, pipe age is not a sufficient criterion and the camera only allows for information coming from inside the sewer system. Sink holes, for example, cannot be detected directly in this way. A different way of approaching decision making in sewer system management is the use of computer models to provide the necessary information required. The modelling of sewer systems is relatively cheap compared to the moving camera method but also generally less accurate. So it comes down to a compromise between cost versus information.

1.1 The problem

The main problem addressed in this thesis is the question whether or not data-assimilation can be applied to urban drainage systems in order to get better predictions for the waterlevels in these systems. Consequently, is it possible to detect and locate certain obstructions in urban drainage systems. Detecting obstructions means concluding that there are obstructions present in the system based on the results and locating means actually pointing out the location of the obstructions more accurately.

Another problem is the calibration of parameters. Depending on the amount of precipitation in the system, rainfall runoff parameters might take slightly different values. These would need to be calibrated in a correct way.

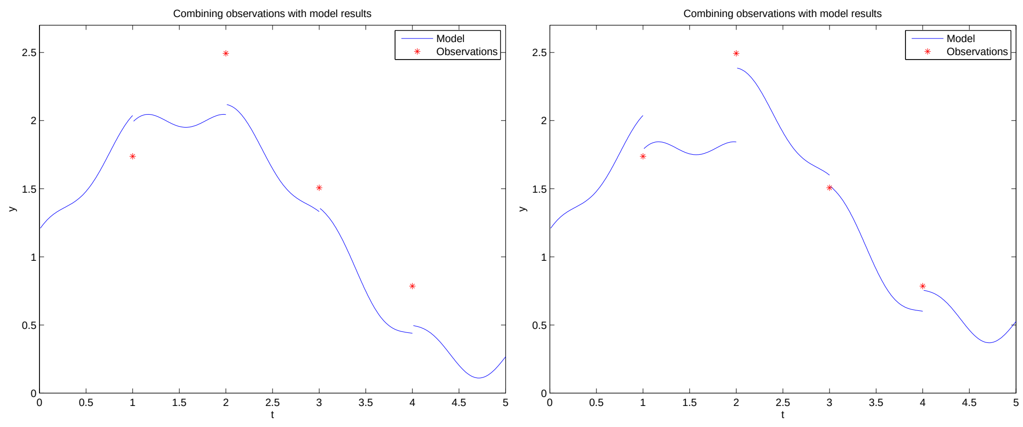
The implementation of a noise model as a means of putting an amount of uncertainty on the waterlevels in the system has to be developed.

1.1.1 General handling of the problem

In this report, the focus is on the modelling part of a certain sewer system model. With the modelling of such systems being a relative cheap source of information, improving this can prove to be quite advantageous to future sewer system management. The particular focus is on improving the predictions of such models by using measurements. The measurements that would be used, are gathered using a monitoring network of sensors placed at different locations in the sewer system. Obviously, placing a sensor in every manhole in the sewer system with a high monitoring frequency would provide a lot of accurate information, though very costly to implement this network and process all the data. This means that the implementation of such a monitoring network is again a matter of cost versus information gained. The design of such a monitoring network has been investigated in [Post, 2012]. Measurement data in this report, however, is not taken from a real monitoring network but instead is generated by the modelling software itself. This is because at this time, no real data is available.

The process of combining model output with measurements is known as data-assimilation. Modelling a sewer system is not as accurate as desired (as mentioned before). So the output of a model comes with some uncertainty. The monitoring network is not perfect, so the data acquired from this system also has some uncertainty. The idea behind data-assimilation is to combine the output from the modelling with the data from the monitoring network to create a new estimate for the state of the system. The uncertainty on the new state estimate should then be smaller than the uncertainty from the modelling output and the uncertainty from the monitoring network. The general idea of data-assimilation can be seen in figure 1.1. In this figure, a model runs up to the observation times, which are $t = 1, \dots, 4$, then the model state is updated with the observations and continued. The figure is only intended to display the general idea in two cases. The first case is where the model state is more reliable than the observations. This can be seen in figure 1.1a, where the model state is updated but remains closer to the model outcome and further from the observations. Figure 1.1b shows what happens in the opposite situation and the uncertainty of the model outcome is larger than that of the observations. In this case, the update will be closer to the observations.

For this report, a model of a sewer network is provided, using the software package Sobek. This is the numerical simulation software. The goal is to combine



(a) (b)

Figure 1.1: A figure showing the concept of data-assimilation. It shows how a model state is updated in time with observations. Figure 1.1a shows the case of a higher uncertainty on the observations while figure 1.1b shows the case in which the uncertainty of the model is larger.

some data-assimilation together with this software. In order to do that, a software package called OpenDA is used. This is an open source package for the application of different kinds of methods of data-assimilation. The combination of both software packages for this purpose will be called the Sobek-OpenDA coupling. It should be possible to run Sobek simulations from OpenDA and assimilate measurements as they become available.

The measurements used in this report are not acquired from a real monitoring network, as mentioned earlier. Instead, the measurements are generated with a simulation performed by Sobek. The output of this simulation will then serve as measurements (or observations) as if they were acquired from a real monitoring network. This process in generating measurements and using it, is called a twin-experiment. Not all the output data is actually used as measurements, because in a real monitoring network, only a few strategically chosen points will have sensors installed. This way the real monitoring network is copied in a realistic way. Another major advantage of the twin-experiment is that all the extra data that is acquired in the output, but not used as measurements, can be used to verify results from the data-assimilation.

1.1.2 Specified goals

As mentioned before, the main objective of this work is to show that calibration and data-assimilation can be applied in the field of urban drainage. In particular, get more accurate results for waterlevels in the system and possibly an idea as to whether or not obstructions are present in the system

The main objective translates to developing a working Sobek-OpenDA coupling with plausible results for data-assimilation and calibration, which is achieved by working on a few different issues:

- First thing needed is the basic Sobek-OpenDA coupling. This is basically a prerequisite for the other parts. The goal is to be able to run the numerical model (Sobek) from OpenDA, so it is necessary to identify the required Sobek executable and the necessary model data. Then there is the implementation of some java classes used as a means to read and/or modify output and input data for the numerical model, in particular the state of the system and several parameter values.

- Visualization of results. The output data from the twin-experiment that is used to form the measurements, is also used to verify results from the data-assimilation method by showing the difference between the data-assimilation results and the original results.

  Since a lot of the data-assimilation will be done with the ensemble Kalman filter, the ensemble standard deviation at the node locations is also a point of interest. A high standard deviation will represent the locations where the uncertainty is highest.

  The model state will be adjusted using the measurements at several different locations. It is interesting to know the influence of each of the observation points on the entire state of the model. This can be done by plotting in each point, the correlation with the observation location of interest.

- Parameter calibration is an important part of the work. When optimal parameter values have been determined, model results that are then created with these new and improved values will have a smaller uncertainty. Simply because one source of possible uncertainty, the parameter values, has been removed or significantly lowered.

- Assimilation with the Kalman filter is another part of the work that is quite important. This way, the waterlevels in the sewer system can be updated in time as measurements become available.

A schematical representation of these issues can been seen in figure 1.2. Most of the work will be done within the blackbox part, however the visualization is not to be underestimated. The visualization concerns itself with processing the output data and provides a way of confirming or discarding results by a visual means (figures and graphs).

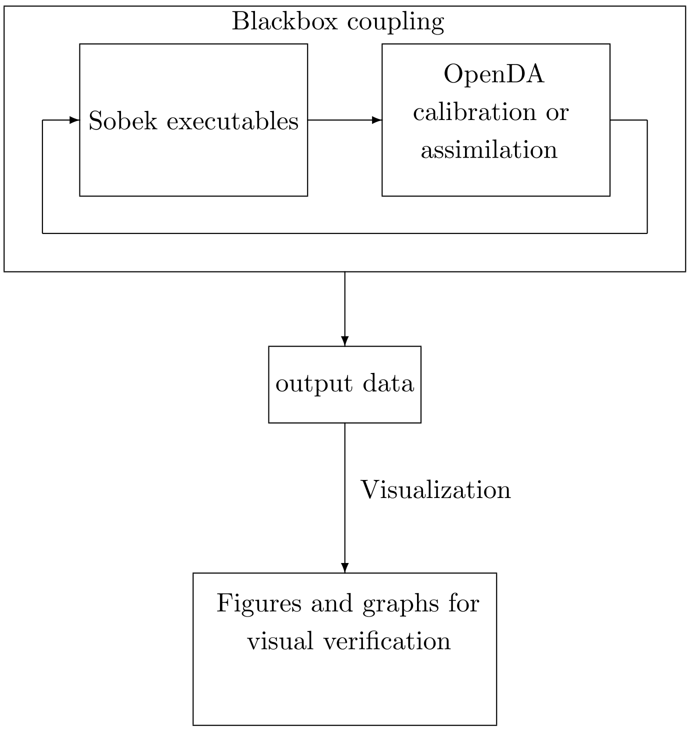

Blackbox coupling
Sobek executables
OpenDA calibration or assimilation
output data
Visualization
Figures and graphs for visual verification

Figure 1.2: A process scheme showing the separate steps taken to reach the objective.

1.2 Thesis outline

Firstly, the theoretical background of data-assimilation and calibration is presented. Starting with the traditional Kalman filter for linear problems, which was the basis for the ensemble Kalman filter that can be applied to nonlinear problems. The traditional Kalman filter and ensemble Kalman filter have also been the bases of several other filter algorithms. Some of these filter algorithms are also discussed in chapter 2. A calibration method that goes by the name Doesn't use Derivatives (DuD) is also discussed in chapter 2. This method is applied to perform calibration of certain parameters in chapter 4.

For the practical work, two software packages have been used. The open source software package OpenDA for the data-assimilation and the hydrodynamic software package Sobek for the numerical modelling. These two software components have been combined into what is called a blackbox coupling. Both of the software packages as well as the blackbox coupling are discussed in more detail in chapter 3.

Chapter 4 deals with the calibration of the model parameters using the DuD method. This involves a good as possible fitting of the model output to the measured data, resulting in a set of parameters that are optimal for the measured data. One of the parameters that is calibrated is the amount of surface area belonging to some of the nodes in the network. This parameter influences the amount of water that enters the sewer system due to rainfall. Another parameter that is calibrated is the rainfall runoff. This is a parameter that influences the rate at which the water enters the sewer system. The parameter calibration makes use of measurements that are created with a twin-experiment. This means that there is little to no uncertainty in these measurements and the calibration is performed in a very controlled environment.

Data-assimilation is the main subject of chapter 5. Since the model is nonlinear, assimilation is done by means of the ensemble Kalman filter. For each ensemble, the model is run up to a time where measurements are available and these are then assimilated, improving the current state. The measurements used in the assimilation are again generated by means of a twin-experiment, leaving little room for uncertainty. Though naturally, for performance testing purposes, there is uncertainty defined on these measurements as if they were taken in a real sewer system. The imposed uncertainty of the model results is of particular interest. Several different ways of introducing noise on the state vector have been implemented and results compared.

Chapter 6 is the result from the extensively tested blackbox coupling for Sobek and OpenDA. In this chapter the content from the previous chapters, especially 4 and 5, is summarized. Conclusions are drawn regarding the blackbox coupling and its results and some recommendations regarding future research are made.

Chapter 2

Data-assimilation

Data-assimilation is the process of including observations in the modelling of physical processes. Stated simply, the idea is to improve certain model predictions to better match the truth by using measurements. For example, one of the main fields in which data-assimilation is applied is weather forecasting. In weather forecasting, a numerical model is used to create predictions for the weather in the future. Measurements of temperature or atmospheric pressure can be used in time to improve the current state of the system or they can be used to improve the starting point (initial state) of the simulation.

Much like numerical model predictions, measurements are also not 100% accurate. But the idea behind data-assimilation is to create a new state estimate from the combination of model predictions and measurements, that has a better accuracy, or lower uncertainty, than either the model predictions or measurements alone. If there is a rather large difference between the uncertainties of measurements and model predictions, for example, the uncertainty in the observations is very large, then the best guess for the true state is more likely to match that of the model output. On the other hand, if the observation uncertainty is very small and the model uncertainty is rather large then the best guess will be close to the observed one.

Data-assimilation methods can be variational or sequential (for a comparison of the two, see for example [Rabier et al., 1992]). The focus of this report is on the sequential data-assimilation, in particular the Kalman filter. When applying sequential data-assimilation, the state is updated in time as observations become available. This means the model will run up to a time k, where observations y_k are available. These observations are then assimilated at this time. Progressing the model state through time up to k is called the forecast and the result is the forecasted state x_k^f. Then the observations at this time are used to update the forecasted state. This is called the analysis and the result is the analysed state x_k^a.

2.1 The basic model

A model will usually have a form which is similar to

$$x_{k+1} = m(x_k) + G_k w_k \tag{2.1}$$

In this representation the model can be linear or nonlinear, depending on the form of m. The term $G_k w_k$ represents the modelling error or uncertainty in the model, since a model is not likely to produce perfect predictions. When the model is linear, it can be written as

$$x_{k+1} = A_k x_k + G_k w_k \tag{2.2}$$

Besides the model predictions, there are also measurements available. The notation for the measurements is

$$y_k = H_k x_k + z_k \tag{2.3}$$

Here x_k denotes the state at time k, A_k is the (time dependent) linear model operator, G_k is the noise input matrix for the state and H_k is the matrix that maps the state space to the observation space. The terms w_k and z_k represent the error in the state and in the measurements. The assumption is that the model and measurement error statistics have a Gaussian distribution. Which means that the terms w_k and z_k both are i.i.d. Gaussian with means $\mathbb{E}[w_k] = 0$, $\mathbb{E}[z_k] = 0$ and covariance matrices $Q_k = \mathbb{E}[w_k w_k^T]$ and $R_k = \mathbb{E}[z_k z_k^T]$. The matrix R_k is assumed to be constant in time, so $R_k = R$. The covariance and mean of the state propagate in time according to

$$\begin{aligned} \bar{x}_k = \mathbb{E}[x_k] &= \mathbb{E}[A_{k-1}x_{k-1} + G_{k-1}w_{k-1}] \\ &= A_{k-1}x_{k-1} + G_{k-1}\mathbb{E}[w_{k-1}] \\ &= A_{k-1}\bar{x}_{k-1} \end{aligned} \tag{2.4}$$

The state covariance is $P_k = \mathbb{E}[(x_k - \bar{x}_k)(x_k - \bar{x}_k)^T] = \mathbb{E}[SS^T]$. Evaluating S_k using equations (2.2) and (2.4) gives

$$\begin{aligned} S_k = x_k - \bar{x}_k &= A_{k-1}x_{k-1} + G_{k-1}w_{k-1} - A_{k-1}\bar{x}_{k-1} \\ &= A_{k-1}(x_{k-1} - \bar{x}_{k-1}) + G_{k-1}w_{k-1} \\ &= A_{k-1}S_{k-1} + G_{k-1}w_{k-1} \end{aligned}$$

and so the state covariance matrix at k becomes

$$\begin{aligned} P_k = \mathbb{E}[S_k S_k^T] &= A_{k-1}\mathbb{E}[S_{k-1}S_{k-1}^T]A_{k-1}^T + A_{k-1}\mathbb{E}[S_{k-1}w_{k-1}^T]G_{k-1}^T \\ &\quad + G_{k-1}\mathbb{E}[w_{k-1}S_{k-1}^T]A_{k-1}^T + G_{k-1}\mathbb{E}[w_{k-1}w_{k-1}^T]G_{k-1}^T \\ &= A_{k-1}P_{k-1}A_{k-1}^T + G_{k-1}Q_{k-1}G_{k-1}^T \end{aligned} \tag{2.5}$$

2.1.1 Analysis maximum likelihood

Equations 2.2 and 2.3 can be regarded as being probability densities. It can be said that the model forecast has a normal distribution with mean the true value. The same goes for the measurements, these are normally distributed as well where the mean is the true value at the observation location. These are results from the properties of w_k and z_k. An approach to determine the analysis is to look for the most probable solution given the forecasted values and the measurements. This approach makes use of Bayes' theorem.

Bayes' theorem and Gaussian distributions

Bayes' theorem gives an important relationship between the probabilities of X and Y, and the conditional probabilities X given Y and Y given X:

$$\mathbb{P}(X|Y) = \frac{\mathbb{P}(Y|X)\mathbb{P}(X)}{\mathbb{P}(Y)}$$

or when dealing with probability density functions

$$f_{X|Y} = \frac{f_{Y|X} f_X}{f_Y} \tag{2.6}$$

This theorem is very useful because, as mentioned before, both the model state and measurements are assumed to be Gaussian and so the multiplication of probability densities also has a Gaussian shape. Division by f_Y in Bayes' theorem is done to turn $f_{X|Y}$ into a proper density function which will integrate to 1 but has no influence on the mean or covariance of the event X given Y. At some random time where measurements are available, the meaning of these probability density functions in terms of data-assimilation is as follows: $f_{X|Y}$ is the probability density of the analysed state, x_a. $f_{Y|X}$ is the probability density of the measurements and f_X is the density of the state without measurements, which is the forecast state since this is acquired by propagating the previous (usually analysed) state forward in time.

A multivariate Gaussian distribution with mean μ and covariance matrix C has the following density function

$$f(x) = \frac{1}{\sqrt{\det(C)(2\pi)^n}} \mathrm{e}^{-\frac{1}{2}(x-\mu)^T C^{-1}(x-\mu)}$$

Taking logarithms to eventually define a cost function

$$\begin{aligned}
-\ln(f(x)) &= -\frac{1}{2}(x-\mu)^T C^{-1}(x-\mu) + \mathit{constant} \\
J(x) &= -\frac{1}{2}(x-\mu)^T C^{-1}(x-\mu) \\
\nabla J(x) &= C^{-1}(x-\mu) \\
\frac{\partial^2 J(x)}{\partial x^2} &= C^{-1}
\end{aligned}$$

$J(x)$ is determined by μ and C and the Hessian, $\frac{\partial^2 J(x)}{\partial x^2}$, is the inverse of the covariance matrix C, which is a useful property.

Ignoring the time component for now and applying Bayes' theorem to the probability density of the analysis given the forecast and observations gives

$$f_{x|y} = \frac{f_{y|x} f_x}{f_y}$$

From this equation, a cost function, $\tilde{J}$, can be defined as

$$\tilde{J}(x) = -\log(f_{x|y}) = -\log(f_{y|x}) - \log(f_x) + constant$$

where f_x has a $\mathcal{N}(x, P^f)$ distribution, where the superscript indicates that this is the forecast, and $f_{y|x}$ has a $\mathcal{N}(Hx, R)$ distribution

$$\tilde{J}(x) = \frac{1}{2}(y_k - H_k x)^T R^{-1}(y_k - H_k x) + \frac{1}{2}(x - x_k^f)^T (P_k^f)^{-1}(x - x_k^f) + constant$$

the minimum of this cost function is the point of interest, so the constant can be disregarded. Then the final cost function becomes

$$J(x) = \frac{1}{2}(y_k - H_k x)^T R^{-1}(y_k - H_k x) + \frac{1}{2}(x_k^f - x)^T (P_k^f)^{-1}(x_k^f - x) \tag{2.7}$$

Minimization of this function will give the analysis, x_k^a. In order to accomplish this, the gradient of $J(x)$ is calculated, set equal to 0 and solved for x. It is convenient to use $\delta x = x - x_k^f$. The cost function $J(\delta x)$, the gradient and Hessian then become

$$\begin{aligned}
J(\delta x) &= \frac{1}{2}(\delta x)^T (P_k^f)^{-1}(\delta x) \\
&\quad + \frac{1}{2}(y_k - H_k\delta x - H_k x_k^f)^T R^{-1}(y_k - H_k\delta x - H_k x_k^f) \\
\nabla J(\delta x) &= (P_k^f)^{-1}(\delta x) - H_k^T R^{-1}(y_k - H_k\delta x - H_k x_k^f) \\
\frac{\partial^2 J(\delta x)}{\partial(\delta x)^2} &= (P_k^f)^{-1} + H_k^T R^{-1} H_k = (P_k^a)^{-1}
\end{aligned}$$

Solving $\nabla J(\delta x) = 0$ gives

$$\begin{aligned}
\nabla J(\delta x) = (P_k^f)^{-1}\delta x - H_k^T R^{-1}(y_k - H_k\delta x - H_k x_k^f) &= 0 \\
((P_k^f)^{-1} + H_k^T R^{-1} H_k)\delta x &= H_k^T R^{-1}(y_k - H_k x_k^f) \\
\delta x &= ((P_k^f)^{-1} + H_k^T R^{-1} H_k)^{-1} H_k^T R^{-1}(y_k - H_k x_k^f) \\
x &= x_k^f + \Big((P_k^f)^{-1} + H_k^T R^{-1} H_k\Big)^{-1} H_k^T R^{-1}(y_k - H_k x_k^f)
\end{aligned}$$

2.2 The Kalman filter

The Kalman filter was introduced by Rudolph Kalman [Kalman, 1960]. It is a recursive solution to the minimization problem where J in equation 2.7 is the cost function. For more information on (the derivation of) the Kalman filter, see also [Kalman, 1960, Simon, 2001, Wilders and Heemink, 2011].

2.2.1 Kalman filter equations

The Kalman filter has a predictor-corrector structure. The prediction being the propagation of the state through time to a point where measurements are available. The corrector part will update the state at a certain time by incorporating measurements. The prediction part will result in a so called forecast probability density function for the state $f_{x(k)}$. The corrector part will use the forecast density together with the probability density of the measurements $f_{y(k)|x(k)}$ to determine the probability density of the analysed state by applying Bayes' theorem

The set of Kalman filter equations are then
time-step (forecast)

$$\begin{aligned} x_k^f &= A_{k-1}x_{k-1}^a \\ P_k^f &= A_{k-1}P_{k-1}^a A_{k-1}^T + G_{k-1}Q_{k-1}G_{k-1}^T \end{aligned} \tag{2.8}$$

measurement-step (analysis)

$$\begin{aligned} x_k^a &= x_k^f + K_k(y_k - H_k x_k^f) \\ K_k &= \left((P_k^f)^{-1} + H_k^T R^{-1} H_k\right)^{-1} H_k^T R^{-1} \\ &= P_k^f H_k^T \left(H_k P_k^f H_k^T + R\right)^{-1} \\ P_k^a &= \left((P_k^f)^{-1} + H_k^T R^{-1} H_k\right)^{-1} \\ &= (I - K_k H_k)P_k^f \end{aligned}$$

2.3 The Ensemble Kalman filter (EnKF)

The ensemble Kalman filter was first introduced by Evensen in [Evensen, 1994b] and is examined and discussed in various articles after that, see for instance [Houtekamer and Mitchell, 1998], [Evensen, 1997], [Gillijns et al., 2006], [Burgers et al., 1998]. Section 2.1 in [Evensen, 2004] lists quite a lot of applications and research done on the ensemble Kalman filter for the interested reader.

The ensemble Kalman filter allows nonlinear models to be used in a general Kalman filter setup. The ensemble Kalman filter propagates an ensemble of model states forward in time. Whereas the basic Kalman filter propagates a single model state, or rather mean and covariance of the model state (possible because of the use of a linear model), forward in time. When using the EnKF, the mean and covariance are calculated from the ensemble of forecasts, this is the forecast step. In the analysis step, the sample mean and covariance are then used to calculate a Kalman gain matrix in order to assimilate the observations, resulting in the analysed state.

2.3.1 Terms and notation

Starting with a nonlinear model

$$x_{k+1} = m(x_k) + G_k w_k \tag{2.9}$$

In this equation, m is the nonlinear propagation operator. Because of the nonlinearity of the model, the mean and covariance cannot be simply propagated in time. Instead an ensemble of states is created and each of these is propagated in time. Let N be the size of this ensemble. The sample mean will then be

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

For readability, the time subscript k, that was used before in the derivation of the Kalman filter, is omitted. The subscript i used in this expression denotes a particular ensemble member. The covariance matrix is calculated in a similar way as done in the Kalman filter, namely

$$P_e = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})(x_i - \bar{x})^T = \frac{1}{N-1}SS^T \tag{2.10}$$

where $S = [S_1, \ldots, S_N]$ is the ensemble of perturbations

$$S_i = x_i - \bar{x}$$

2.3.2 Ensemble Kalman filter equations

The ensemble Kalman filter equations consist, as in the basic Kalman filter case, of a forecast step and an analysis step. In the forecast step, the ensemble is propagated in time according to the nonlinear model equation (2.9), so for each ensemble member x_i

$$x_{i,k}^f = m(x_{i,k-1}^a) + G_{k-1}w_{i,k-1}$$

$$P_{e,k}^f = \frac{1}{N-1}S_k^f(S_k^f)^T$$

In the analysis step, it is essential that observations are treated as random variables with mean equal to the prescribed observation value and covariance equal to R [Evensen, 2003]. So an ensemble of observations can be defined as

$$y_i = y + \epsilon_i \tag{2.11}$$

where i runs from 1 to the number of ensemble members and ϵ_i denote the random measurement error which have mean 0. The measurement ensemble covariance matrix is then

$$R_e = \mathbb{E}[\epsilon\epsilon^T]$$

Updating the state of each ensemble member is done in the following way

$$x_i^a = x_i^f + K_e(y_i - Hx_i^f) \tag{2.12}$$
$$K_e = P_e^f H^T(HP_e^f H^T + R_e)^{-1} \tag{2.13}$$

in which the perturbed measurements y_i are used as well as the covariance matrix R_e belonging to these measurements. Equation (2.12) implies that

$$\bar{x}^a = \bar{x}^f + K_e(\bar{y} - H\bar{x}^f) \tag{2.14}$$

where $\bar{y} = y$. This means that the relation between the forecast mean and the analysed mean in the ensemble Kalman filter is the same as the relation between the forecast state and the analysed state in the standard Kalman filter, apart from the use of ensemble notations for covariances $P_e^{f,a}$, R_e instead of $P^{f,a}$ and R. Equations (2.12) and (2.14) result in

$$x_i^a - \bar{x}^a = (I - K_e H)(x_i^f - \bar{x}^f) + K_e(y_i - \bar{y}) \tag{2.15}$$

which is used to calculate the covariance matrix for the analysis step

$$\begin{aligned} P_e^a &= \mathbb{E}[(x^a - \bar{x}^a)(x^a - \bar{x}^a)^T] \\ &= (I - K_e H)P_e^f(I - H^T K_e^T) + K_e R_e K_e^T \\ &= P_e^f - K_e H P_e^f - P_e^f H^T K_e^T + K_e(H P_e^f H^T + R_e)K_e^T \\ &= (I - K_e H)P_e^f \end{aligned} \tag{2.16}$$

2.4 DuD

The algorithm called DuD , which stands for "doesn't use derivatives", is an algorithm used to solve a nonlinear least squares problem [Ralston and Jennrich, 1978]. This algorithm, with some minor adjustments, is used in OpenDA to solve the weighted nonlinear least squares problem

$$Q(\theta) = \sum_{i=1}^{n} W_{ii}(y_i - f_i(\theta))^2 \tag{2.17}$$

where y_i are elements of an observation vector, $W_{ii} = 1/\sigma_i^2$ is the uncertainty of each of the observed data points and $f_i(\theta)$ are elements of a vector valued response function. As the name suggests, DuD does not use derivatives to solve this least squares problem. Instead, the function $f(\theta)$ is approximated by a linear function $l(\alpha)$ which agrees with $f(\theta)$ at $p + 1$ previous values of the parameter vector θ. p will be equal to the number of elements in the parameter vector.

The linear approximation is written as a function of the vector α, which has a relation to the parameter vector θ in the following way

$$\theta = \theta_0 + \Delta\Theta\alpha \tag{2.18}$$

where $\theta_0, \ldots, \theta_p$ are estimates of the parameter vector at previous iterations and the i^{th} column of the matrix $\Delta\Theta$ is given by

$$\Delta\Theta_i = \theta_i - \theta_0, \quad i = 1, \ldots, p$$

The linear approximation is then given by

$$l(\alpha) = f(\theta_0) + \Delta F \alpha$$

Where the i^{th} column of the matrix ΔF is given by

$$\Delta F_i = f(\theta_i) - f(\theta_0)$$

In one iteration, the algorithm attempts to minimise a new weighted cost function.

$$\tilde{Q}(\alpha) = \sum_{i=1}^{n} \frac{1}{\sigma_i^2}(y_i - l_i(\alpha))^2 = (y - l(\alpha))^T R^{-1}(y - l(\alpha)) \tag{2.19}$$

Where R^{-1} is the matrix containing $1/\sigma_i^2$ on the diagonal and zeros elsewhere since the observation errors are uncorrelated.

$$\alpha = (\Delta F^T R^{-1} \Delta F)^{-1} \Delta F^T R^{-1}(y - f(\theta_0)) \tag{2.20}$$

A new value for the parameter vector, θ_{new} can then be calculated from (2.18). In the case that $Q(\theta)$ is not reduced, DuD makes use of a line search to find a new parameter vector until $Q(\theta_{\text{new}}) < Q(\theta_0)$ or until the maximum number of line search attempts, m, has been reached.

$$\begin{aligned} \theta_{\text{new}} &= d\theta_{\text{new}} + (1-d)\theta_0 \\ d_i &= \left\{ \begin{array}{ccc} (\beta)^i & i = 1, \ldots, m, & i \leq k \\ (-1)^{i-k}(\beta)^i & i = 1, \ldots, m, & i > k \end{array} \right. \end{aligned}$$

If after k line search iterations, which is a number defined by the user, no improvement for the cost is found, then the line search will also be performed in the negative search direction (which follows directly from definition of d_i). The parameter β is a shortening factor indicating at what distances from θ_0 and θ_{new} the line search is performed.

After each iteration, the parameter vector with the highest cost is replaced by the new parameter vector and then the parameter vectors are ordened according to the cost function such that $Q(\theta_0) \leq Q(\theta_1) \leq \ldots \leq Q(\theta_p)$.

Startup

The algorithm needs, at each iteration, $p+1$ previous estimates of the parameter vector and one new estimate is calculated. Since the algorithm starts with no estimates, the first $p+1$ estimates for this parameter vector are generated in the startup phase. The respective cost that comes with these parameters needs to be determined as well, which is a rather time consuming part of the algorithm because this involves a model simulation for each of the parameter vectors. The first parameter vector will be the user supplied starting vector, θ_0. The next p vectors are acquired by displacing the i^{th} component of θ_0 by a nonzero amount, usually specified by the user as an uncertainty (or offset) for this specific parameter. Once these $p+1$ vectors are available, they are ordened according to cost and the above iterative process begins.

In the configuration of the algorithm, the uncertainty of the parameters can be defined in two different ways. Let the uncertainties be $\lambda_1, \lambda_2, \ldots, \lambda_p$. The first way in which these uncertainties can be used is where the uncertainty itself represents the offset value that will be used in the creation of the first p parameter vectors. This can be seen in equation (2.21), where every i^{th} component of the parameter vector θ_i has been modified by an amount λ_i.

$$\text{for} \quad i = 1, \ldots, p: \quad \theta_i(j) = \left\{ \begin{array}{cc} \theta_0(j) & j \neq i \\ \theta_0(j) + \lambda_j & j = i \end{array} \right. \tag{2.21}$$

The other way of using the uncertainties is where they are used as an offset factor. The way in which the initial parameter vectors are created then can be seen in equation (2.22). In this case, for each parameter vector θ_i, the i^{th} component is modified by an amount $\lambda_i \theta_0(i)$.

$$\text{for} \quad i = 1, \dots, p: \quad \theta_i(j) = \begin{cases} \theta_0(j) & j \neq i \\ \theta_0(j) + \lambda_j \theta_0(j) & j = i \end{cases} \tag{2.22}$$

Stop criteria
The algorithm has several stop criteria:

- when the maximum number of iterations has been reached
- when $|Q(\theta_{\text{new}}) - Q(\theta_0)| \leq T_1$
- when $\frac{|Q(\theta_{\text{new}}) - Q(\theta_0)|}{Q(\theta_0)} \leq T_2$

Where the maximum number of iterations, T_1 and T_2 are user defined.

2.5 Identifiability analysis

An important part of parameter estimation is identifiability and uniqueness of parameters. Parameter identification gives information about the ability to estimate parameters individually (or possibly in groups). Resulting from this analysis, if parameters are identifiable, is that one set of parameters leads to one possible result. Note that this does not imply the uniqueness of the acquired result, as different parameter sets can lead to different results.

There are several methods of performing an identifiability analysis, one of which is called the 'a priori global identifiability'. This method is applied in [Stolze, 2008], which in turn is based on the method described in [Evans et al., 2002].

Another method of performing an identifiability analysis, and the method of interest in this report, is called 'a posteriori identifiability analysis' [Britta, 2000]. This analysis is performed after the parameters are estimated, assuming the optimal parameter vector is found. This identifiability analysis is based on the singular value decomposition of the Jacobian. The Jacobian J is an n x k matrix where n is the number of measurement values available and k is the number of parameters, i.e. the number of elements in the parameter vector θ. J can be seen as a sensitivity matrix, as it describes how the model is influenced by changes in the different parameters at some optimal value for the parameter vector, θ_0. The Jacobian is constructed by performing a run with the calibrated set of parameters θ_0 followed by k model runs, each with a parameter set that differs only in a single element compared to the calibrated one. This yields a matrix ∂Y. From this the Jacobian can be calculated by dividing the residues (in each column) by the change in the corresponding parameter, which looks like

$$J(\theta_0) = \frac{\partial Y(\theta_0)}{\partial \theta} = \begin{bmatrix} \frac{f(\theta_1) - f(\theta_0)}{\Delta p_1} & \dots & \frac{f(\theta_k) - f(\theta_0)}{\Delta p_k} \end{bmatrix} \tag{2.23}$$

A singular value decomposition of the Jacobian will yield an expression for the Jacobian using three different matrices according to equation (2.24).

$$J(\theta) = U \Sigma V^{\mathrm{T}} \tag{2.24}$$

In this equation, Σ is an n x k matrix with all singular values on the diagonal and zero elsewhere. The singular values are the square roots of the eigenvalues of the matrix $J^{\mathrm{T}}J$. V is a unitary k x k matrix where each column is an eigenvector of $J^{\mathrm{T}}J$.

This singular value decomposition is used in the identification analysis of the different parameters. It is particularly useful in determining the influence of individual (or sets of) parameters on model outcome. Using (2.23) and (2.24) the following relation is found

$$U\Sigma V^{\mathrm{T}} = \frac{\partial Y(\theta_0)}{\partial \theta}$$
$$U\Sigma V^{\mathrm{T}}\partial\theta = \partial Y(\theta)$$

This relation indicates that for a singular value that is zero, the corresponding parameters do not influence the model outcome. When the singular value is not exactly zero but rather very small, a change in the corresponding parameters results in a very small change in model outcome. The parameters that correspond to a singular value are found via the inner product of the corresponding column of the matrix V with the parameter vector. Parameters or parameter combinations that belong to the largest singular values are the most identifiable.

Using the Jacobian, there is also other information that can be extracted. The correlation between parameters is an example of this. In order to acquire the correlation, first the covariance matrix is calculated according to equation (2.25)

$$\mathrm{cov} = \left(J^{\mathrm{T}}J\right)^{-1} \tag{2.25}$$

Note that cases may occur in which zero valued singular values appear. In these cases, adaptions are made in order to calculate the covariance matrix. However, in this report the singular values are not equal to zero and as such, equation (2.25) holds. Using the covariance matrix, the coefficients in the correlation matrix are calculated as follows

$$\mathrm{cor}(i,j) = \frac{\mathrm{cov}(i,j)}{\sqrt{\mathrm{cov}(i,i)\mathrm{cov}(j,j)}} \tag{2.26}$$

Through visualisation of the column vectors of the matrix V and the correlation matrix, parameter relations and identifiability become clear. The column vectors of V belonging to the largest singular values provide information on the most identifiable parameters.

The methods described form a basis for parameter identication. For more information, especially on parameter identification in urban drainage modelling, the reader is refered to the works of [Clemens, 2001], [Post, 2012], [Stegeman, 2012].

Chapter 3

Model characteristics

Besides the theoretical part, there is a practical part involved in the work in this report. This means working with two software packages: Sobek, which is a hydrodynamic modelling package and OpenDA, an open source package for the application of data-assimilation and calibration in numerical modelling. The objective is to successfully model and apply data-assimilation on (a part of) the sewer network of the city Delft using Sobek and OpenDA.

The different software packages are discussed in this section, as well as coupling these by means of a blackbox wrapper. This section also contains some information specific to the network of Delft, that is used in the simulations.

3.1 Sobek

The modelling software used to simulate the behaviour in the sewer system is called Sobek. This is a rather large software package and since only parts of the package are used, the following sections only contain information that is relevant to the modelling of these sewer systems. Further infomation on Sobek can be found in [Deltares, 2013].

3.1.1 About Sobek

The following description is taken from the Sobek manual [Deltares, 2013] and seems an appropriate introduction of the used software package Sobek. Since it captures quite well the essence of modelling software; providing predictions as accurate as possible.

Sobek is named after the ancient Egyptian crocodile river god. Crocodiles were believed to have predictive powers, as they would lay their eggs just above the level of the upcoming Nile flood.

Sobek is an integrated software package for river, urban or rural management. Seven program modules work together to give a comprehensive overview of waterway systems keeping you in control. Its integrated framework also means that Sobek can link river, canal and sewer systems for a total water management solution. This program is very easy to configure and quick to learn. It guides you in obtaining correct model descriptions. The graphically oriented interface makes it easy for the user to quickly set up a model. An example of a simple model for a sewer network connected to the open surfacewater can be seen in figure 3.1.

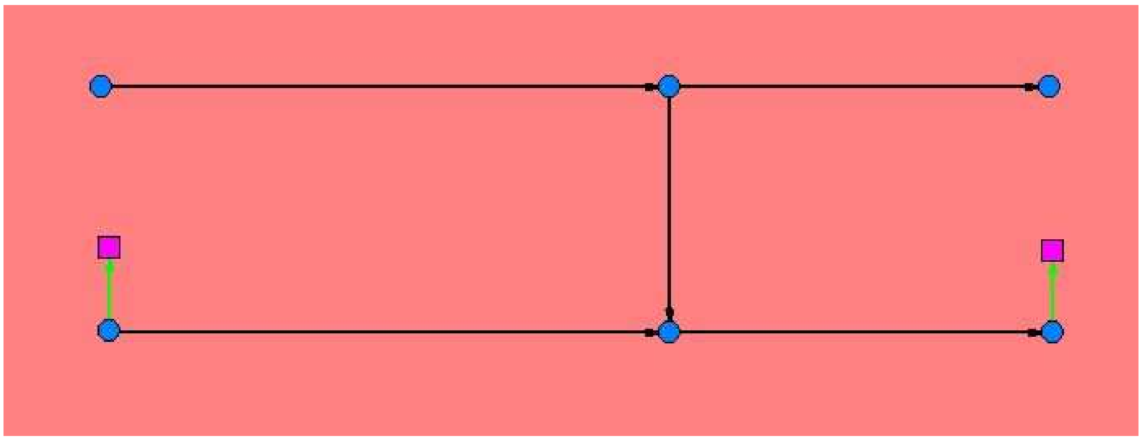


Figure 3.1: This is what a (simple) sewer model looks like in the Sobek graphical interface.

A (part of a) sewer network can been seen in this figure. Graphically the network consists of nodes and reaches (connections). There are, however, lots of different types of nodes and reaches. The blue nodes represent manholes with runoff, the purple nodes are boundaries with a fixed waterlevel, representing open surface water. The two different connections are regular pipes between the nodes (the black reaches) and some internal weirs (green reaches) connecting the sewer system to the open surfacewater. Simply put, a larger sewer network will have a similar setup but likely with more and different types of nodes and reaches.

3.1.2 Conceptual description

The drainage system modelling in this report makes use of two of the program modules from Sobek. Namely the Rainfall-Runoff (RR) module and the 1DFlow (Urban) module. Stated simply, these modules do exactly what their names imply, i.e. the RR module simulates the effects from the precipitation on the sewer system and the Urban module simulates the behaviour in the sewer system itself.

Rainfall Runoff concept

The basic concept of the rainfall runoff can be explained easiest with a picture. Figure 3.2 shows the process starting with rainfall to sewer system.

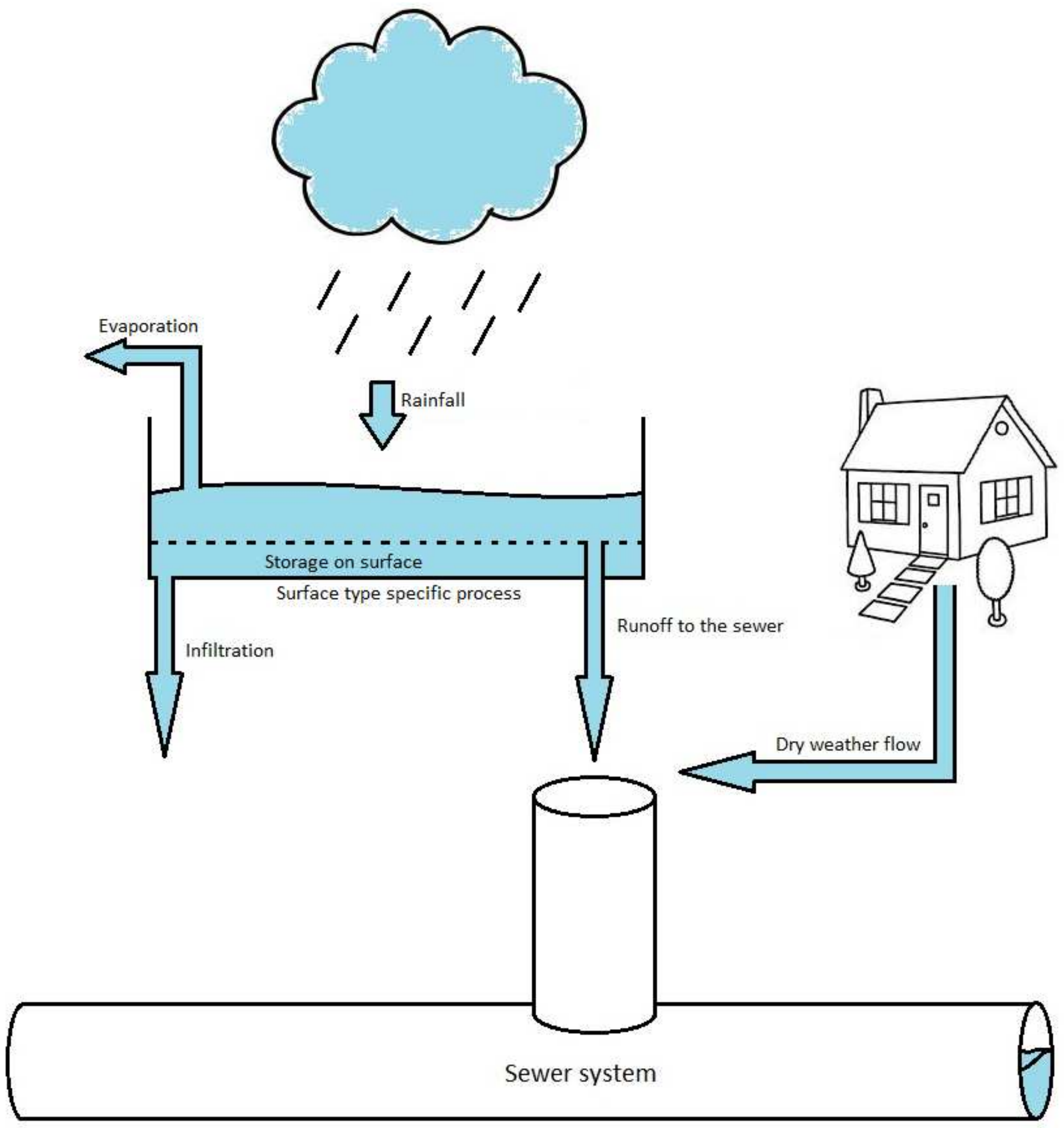


Figure 3.2: A simplified visualisation of the rainfall runoff process in Sobek.

The process starts with a rainfall event with a certain intensity, usually denoted in mm/hr. Using this intensity together with the amount of surface area that is present, an amount of water is caught and saved in an imaginary reservoir. From this reservoir, one part of the water runs off into the sewer system and another part is lost due to infiltration into the ground. The runoff and infiltration processes are dependent on the surface type that is dealt with. A sloped surface will, for example, runoff faster than a flat surface and infiltration will be greater when dealing with unpaved areas rather than the roof of a building.

Next to these surface area specific processes there is the dry weather flow. This is the amount of (waste) water produced by households that enters the system.

These different processes are discussed in more detail below.

Runoff delay
As discussed in [Deltares, 2013, p. 662-664] with some minor adjustments.
Runoff delay depends on the average distance to the inflow location in the sewer

system, the slope and geometry of the catchment. The runoff into the sewer is described according the formula of the Rational Method

$$q = ch$$

where

q	inflow into the sewer, mm/min
c	runoff factor, 1/min
h	rainfall, dynamic storage on catchment, mm

The runoff factor c is a function of length, roughness and slope. Both c and h are defined for twelve different surface types that are described in the "Dutch Guidelines for sewer systems computations, hydraulic functioning" [Deltares, 2013, p. 662]. All twelve types and the default values are listed in table 3.1.

Surface number	Area type	Runoff type	Runoff factor c	Surface storage h
1	closed paved	sloped	0.5	0.0
2		flat	0.2	0.5
3		stretched flat	0.1	1.0
4	open paved	sloped	0.5	0.0
5		flat	0.2	0.5
6		stretched flat	0.1	1.0
7	roof	sloped	0.5	0.0
8		flat	0.2	2.0
9		stretched flat	0.1	4.0
10	unpaved	sloped	0.5	2.0
11		flat	0.2	4.0
12		stretched flat	0.1	6.0

Table 3.1: The default parameter values for c and h (rational method) for the twelve different surface types

Infiltration is the process by which water infiltrates from the surface into the ground. Water that is infiltrated does not end up in the sewer system. The description of the infiltration in the Urban version of the Rainfall Runoff module is based on the formula of Horton:

Decreasing infiltration capacity:

$$f_t = f_e + (f_b - f_e)e^{-k_a t}$$

Recovering infiltration capacity:

$$f_t = f_e - (f_b - f_e)e^{-k_h t}$$

where

Symbol	**Unit**	**Meaning**
f_t	mm/h	infiltration capacity at time t
f_b	mm/h	maximum infiltration capacity at time $t = 0$
f_e	mm/h	minimum infiltration capacity
k_a	$1/h$	time factor decreasing infiltration capacity
k_h	$1/h$	time factor recovering infiltration capacity
t	h	time

The values belonging to these parameters are defined for all the surface areas. It can be noted, however, that the parameter values depend on the area type and not on the runoff type of the specific area. Meaning that for, for example, closed paved area types, the values of these parameters are constant independent of whether it is sloped, flat or stretched flat. All values can be seen in table 3.2.

Nr	Area type	Runoff type	f_b	f_e	k_a	k_h
1, 2, 3	closed paved	any	0.0	0.0	0.0	0.0
4, 5, 6	open paved	any	2.0	0.5	3.0	0.1
7, 8, 9	roof	any	0.0	0.0	0.0	0.0
10, 11, 12	unpaved	any	1.0	1.0	3.0	0.1

Table 3.2: The default parameter values for the Horton equation for the twelve different surface types

Dry weather flow (DWF, [Deltares, 2013, p. 602]) in the model represents the return flow from domestic use like showers, washing, flushing toilets etc. DWF can depend on the number of inhabitants at the specified node and a specific DWF definition. The following options are available for the DWF:

- number of inhabitants multiplied with the constant flow (m^3/s) per day per person
- number of inhabitants multipled with the variable flow (m^3/s, varying every hour) per hour per person
- constant flow during the day
- variable flow (varying every hour) during the day

3.1.3 Mathematical core

[Deltares, 2013, p. 496-497]
The water flow is computed in the 1DFlow module by solving the complete De Saint Venant equations (shallow water equations). That is, it solves the following equations

- continuity equation 1D
- momentum equation 1D

These equations are solved numerically using the Delft-scheme.

Continuity equation (1D)

The continuity equation is

$$\frac{\partial A_f}{\partial t} + \frac{\partial Q}{\partial x} = q_{lat} \tag{3.1}$$

Momentum equation (1D)

The momentum equation looks like

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{Q^2}{A_f}\right) + gA_f\frac{\partial h}{\partial x} + \frac{gQ|Q|}{C^2RA_f} - w_f\frac{\tau_{wind}}{\rho_w} = 0 \tag{3.2}$$

In this equation the first term describes the inertia. The second term describes the convection. The third term describes the water level gradient. The fourth term describes the bed friction and the fifth term describes the wind friction.
The symbols in these equations represent the following

Symbol	Unit	Meaning
A_f	m^2	Cross sectional flow area
Q	m^3/s	Discharge
q_{lat}	m^2/s	Lateral discharge per unit length
x	m	Distance
g	m/s^2	Acceleration due to gravity (≈ 9.81)
h	m	Water level (with respect to reference level)
τ_{wind}	N/m^2	Wind shear stress
ρ_w	kg/m^3	Water density
w_f	m	Cross sectional width at water level

The information contained in section 3.1 up to this point only concerns the relevant information from the manual for the modules used in this report. For a more in depth look into Sobek, a more detailed view of the things discussed above and/or information regarding the other modules, see the Sobek manual [Deltares, 2013].

3.2 OpenDA

OpenDA is an open source software package consisting of a set of tools for the application of data-assimilation and/or calibration to numerical models. The basic framework allows the application of several different data-assimilation methods or calibration methods to be applied to a numerical model that has been prepared for use with OpenDA. This allows users to quickly switch between methods to, for example, compare the performance of these different methods to a particular numerical model.

3.2.1 Available methods

The available methods in OpenDA can be split in two classes, i.e. data-assimilation methods and parameter estimation (calibration) methods. For this report the data-assimilation method used is the ensemble Kalman filter. The method used for calibration is DuD. Besides these two methods there is a range of several other available assimilation and calibration methods in OpenDA:

Data-assimilation methods

- ensemble Kalman filter (EnKF) - the method that is mainly used to create the results in this report
- ensemble square root Kalman filter (EnSR)
- steady state Kalman filter
- particle filter
- 3DVar

Parameter estimation (calibration) methods

- Dud - the parameter estimation method used in this report
- sparse Dud
- simplex
- Powell
- gridded full search
- shuffled complex evolution (SCE)
- generalized likelihood uncertainty estimation (GLUE)
- (L)BFGS
- conjugate gradient: Fleetjer-Reeves, Polak-Ribiere, steepest descent

OpenDA defines certain interfaces, which indicate how different components of OpenDA communicate with each other. For the modelling component, OpenDA has two interfaces: a model interface and a stochmodel interface. The model interface defines the functionalities that a deterministic model should implement. The stochmodel interface defines the stochastic extension of the deterministic model.

When using OpenDA in combination with a model, the model should be extended by implementing these interfaces, called wrapping the model. There are a few ways in which a model can be wrapped for use with OpenDA. The simplest way is to make use of a complete blackbox wrapper. This way of wrapping the model requires the least (to none) interference with the model code. The idea of this blackbox wrapper is that it is only necessary to implement input and output functionalities for input and output data from the model. Once these functionalities are ready, the blackbox model and stochmodel instances can be used to complete the (stochastic) model required for data-assimilation.

The blackbox wrapper is also a natural choice to implement for new software or new numerical models that require a data-assimilation component. There is a minimal amount of work needed to set up the wrapper and there is no modification in the source code of the numerical model. This means that this wrapper is a good first test for the application of data-assimilation with the specific model. OpenDA functionalities have been extensively tested on quite a lot of different models and over time it has been applied in operational processes as well. If then the modelling software (which is Sobek in this case) has also been tested and deemed (nearly) flawless, any errors that then occur using the blackbox wrapper are likely to originate somewhere in coupling of the two software packages, i.e. in the (stoch)model configuration or the input/output functionalities. Naturally, no matter how extensively tested, there can still be some flaws in the modelling software that appear once the blackbox wrapper is in use. Examples of these are described in section 3.3.3.

The full blackbox wrapper still has a clear distinction between modelling software and the OpenDA components, and those are being linked by the implemented functionalities for modifying input and output. There are other methods of wrapping a model that involve more overlap between the modelling software and the OpenDA functionalities. The methods of wrapping a model described next are also visualized in figure 3.3, where the full blackbox wrapper is number 5. Another way to wrap a model, number 1 in the figure, would be to implement a numerical model from scratch in java, matching the requirements that OpenDA imposes on the model code. Several of these models are contained within the distribution of OpenDA, and are developed to test and illustrate several of the OpenDA applications. An other way to wrap models is to combine native code with a wrapping java extension, this is number 2 in figure 3.3. This way the computational core of the code will not be tampered with and the model will simply be extended with the OpenDA wrapper. It is also possible to implement only parts of these interfaces in the model code and make use of a blackbox setup for the other part. For example writing a model in java and also implementing the model instance interface but using the stochastic blackbox utilities for the stochmodel extension. This is number 3. The last option, number 4 in the figure, is the same as number 3 apart from the fact that the computations are done using native code.

The wrapping method used for the coupling of Sobek and OpenDA is number 5 in the figure, the full blackbox wrapper.

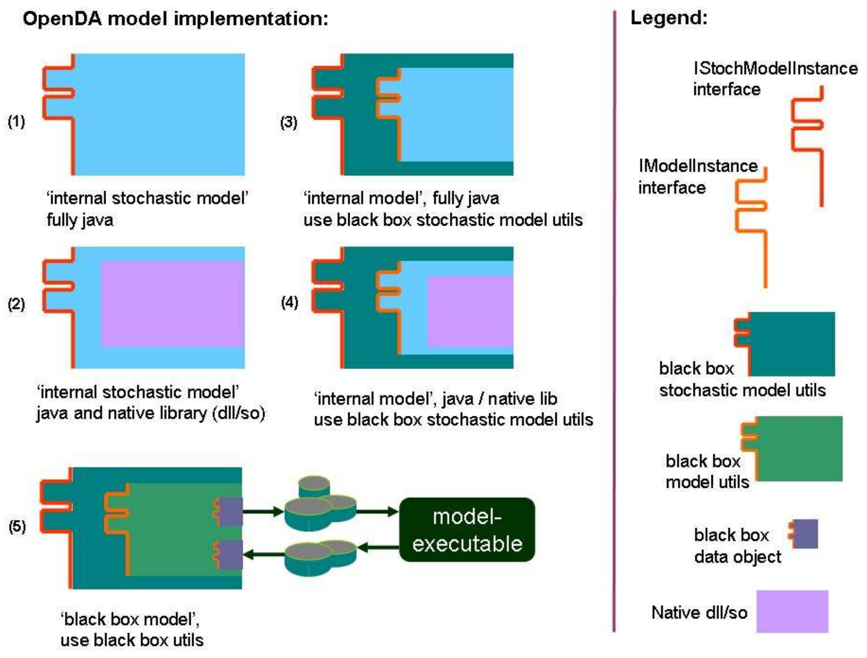

OpenDA model implementation:
(1)
'internal stochastic model' fully java
(3)
'internal model', fully java use black box stochastic model utils
(2)
'internal stochastic model' java and native library (dll/so)
(4)
'internal model', java / native lib use black box stochastic model utils
(5)
model-executable
'black box model', use black box utils
Legend:
IStochModelInstance interface
IModelInstance interface
black box stochastic model utils
black box model utils
black box data object
Native dll/so

Figure 3.3: Visualization of the different types of model wrappers

3.2.2 Blackbox setup

The configuration of a blackbox setup for OpenDA is done with one main config file in which a set of config files is defined. For the model part, this usually consists of three files; a model config file, a stochmodel config file and a wrapper config. Apart from the model part, there is also a config file for the algorithm used and a config file to indicate what observations or measurements are used. Time for a quick overview of these files to comprehend what is going on in each of them and how to use them properly.

The wrapper config file can be seen as the inner layer of the model part config files. The wrapper has a part called aliasDefinitions, in which aliases defined in the model config part can be extended with a prefix or suffix. Then these so called alias keys can be used in other parts of the configuration. The next part in the wrapper is the "run" part. Starting with initialActions that are needed to start up the OpenDA simulation, followed by (a list of) computeActions. These computeActions are usually calls to model executables but can also contain some extra actions like calls to a certain java class, .bat file, .pl file etc. For example, for testing purposes it might be handy or even necessary to perform some action inbetween a few model executable or maybe at the end or a list of executables. This wrapper setup lets the user set up such comuteActions in a very simple and clear way. After a list of computeActions, there can be a list of finalizeActions. This part can be explained by taking the name very literally, i.e. actions performed at the end of a simulation. An example of this could be some processing of output data generated during the simulation. The last part of the wrapper consists of defining several input and output objects. Creating these ioObjects is most of the work in creating a blackbox wrapper, because it is done by implementing java classes according to the interface standard that is compatible with OpenDA. The ioObjects are given an Id tag for use within the OpenDA framework (especially in the model config).

The next config file is the model config file. One of the first things in this file is the reference to the wrapperConfig file, simply to point out what it is called and linking these two files. The next part is the aliasValues section. Several alias keys are defined and values are assigned to these. These alias keys are the same as the ones used in the wrapper config file. Time info is defined in the next part. This can be done by putting a fixed starttime and endtime in the file or by using an exchangeItem for the time info. An exchangeItem will take a starttime and endtime from one of the ioObjects. ExchangeItems are explained next. The exchangeItems part of this file lists exactly what input and output data to retrieve from the model files. The vector id determines what the retrieved data is called within OpenDA or it can mean that simply all the data from this particular ioObject is retrieved. The ioObjectId refers to the ioObjects defined in the wrapper.

The third and last official part of the model config files is the stochmodel config file. This file starts by indicating what the model config file is called, in order to link it to this file. Naturally, through the model config file it is also linked to the wrapper config file. The next part is the vectorSpecification, which in turn consists of a state, parameters and predictor part. The state is simply the state as far as Kalman filtering is concerned. The vectors identified in this part are subject to the filter and will be adjusted accordingly. A part of the state is also the definition of noise models, in order to model the uncertainty in certain state variables. The older noise models used by OpenDA would be defined and all parameters of these would be given in this part. The new noise models for OpenDA will actually be partially defined here and also refer to a separate noise model config file, containing noise model specific parameters. Parameters specific to the noise model will be in the outside config file, like noise model period, standard deviation, time correlation. Inside the stochmodel config file, the link is created that defines which exchange items use a particular noise model.

Besides the model configuration there is also a stochastic observer configuration. This is done with two config files. The first config file is the stochObsConfig. This file has a part that points towards another file in which uncertainties on the measurements are defined, more about this file in a bit. The next part in the stochObsConfig is/are ioObjects. Measurements are, like other data, taken from some file by means of exchange items. This is where the ioObjects come into play. Here the java class and filename are given that are used to retrieve observational data.

In the stochObsUncertainties file, the data from the ioObject is used. Because the measurements are treated as stochastic variables, there will be a probability distribution defined over these. For each of the data points a mean and standard deviation is defined and a type of probability density function, usually normal. This means that the observations are defined as

$$y = y_{\text{file}} + \mathcal{N}(\mu, \sigma^2)$$

turn the observer into a stochastic observer, where the y_{file} is the file containing the observational data. Usually the best guess for the measurements is the value taken from the exchange item, which means that the mean $\mu = 0$. Then σ is the uncertainty of the value of the specific observation point.

The last file needed for the blackbox setup is an algorithm configuration. In this config file, the particular configuration settings for the algorithm chosen can be defined.

3.3 OpenDA and Sobek coupling

One of the goals was to create a blackbox coupling for Sobek with OpenDA. In particular for the simulation of the sewer network of Delft. The basic part of the blackbox coupling had already been created, partially by Johan Post for his MSc work [Post, 2012] and an extension to the blackbox coupling, in particular the input and output object implementation for the state vector, is described in [Mulder, 2013].

3.3.1 Basic coupling setup

The basic coupling setup is a minimal working blackbox wrapper, consisting of the model executables and a few implemented ioObjects. The resulting blackbox wrapper from [Post, 2012] and [Mulder, 2013] yielded bad results for the sewer network model of Delft, however, the basic blackbox wrapper was functional. Meaning that the technical part of the blackbox coupling works, i.e. the communication between Sobek and OpenDA via the exchange items is present. The results from running a simulation with data-assimilation, however, were not plausible. In the wrapper, three Sobek executables and a script are called:

- sobek_3b.exe
- parsen.exe
- delftflow.exe
- rename_restart.bat

The Sobek executables are all the executables necessary to simulate a sewer network. `sobek_3b.exe` and `parsen.exe` are executables to simulate the rainfall runoff module and to parse parameters. `delftflow.exe` is the executable for the flow in the network. The script, `rename_restart.bat`, is used to delete the file sobek.rda and then rename the newly created sobek.nda to sobek.rda. The sobek.rda file is the file that contains the state of the sewer network, containing among other things the waterlevels and velocities inside the network. This file is used as input for the simulation. When the simulation is run a file sobek.nda is created that contains the new state of the system. This file needs to be renamed to sobek.rda to serve as the new input for the next time interval. This is why the script has been made. After every simulation, the old sobek.rda is removed and replaced with the newly created sobek.nda.

The basic blackbox wrapper contains a few ioObjects as a minimal working version to test with. These are

- timeFile - provides information about the start- and endtime for the partial runs.
- reachSegValues - these are the values from the calcpnt.his file. In the blackbox wrapper, these are used as predictions. With the creation of the state ioObject, this could be replaced with prediction values from the stateDate.
- stateData - provides the water level in all nodes. The file used for this ioObject is also the input for the new run, so these exchange items will be put in the state.
- crossSections - this ioObject creates exchange items for several parameters. Runoff delay parameters are some of the most useful in this.

Last part of the blackbox wrapper for the sewer network of Delft are the observations. These are taken from a calcpnt.his file. There are 14 observation locations that have been determined by Johan Post as described in [Post, 2012]. The basic goal in determining the observation locations was to minimize cost of measurement equipment while maximizing the information value. Figure 3.4 shows the city centre part of the sewer network. The highlighted nodes in red are the chosen observation locations.

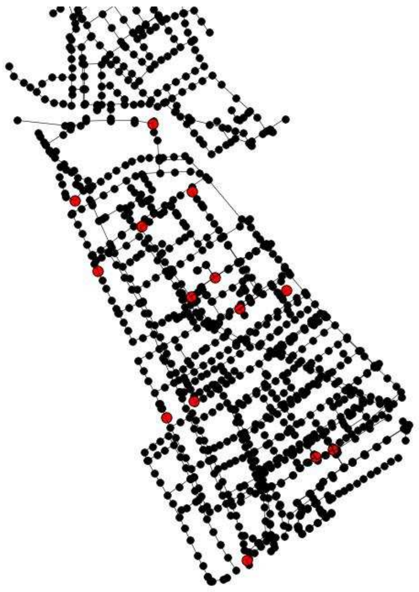


Figure 3.4: Big part of the sewer netwerk with the observation locations highlighted in red.

3.3.2 Restarting

A very important and necessary feature of modelling software that would be used in combination with OpenDA, is the ability to restart. This means that it must be possible to stop the simulation in progress, write restart files containing information about the current state of the system and use these files as input to continue the simulation. Looking at figure 3.3, wrapper option 5, restarting occurs everytime data is transfered from the model executables to the OpenDA blackbox utils and then back to the model executables. Both Sobek modules that are used, have the option to write and read restart files. Enabling these options and giving the read and write restart files the same name, allows the ability to restart simulations from within OpenDA. The only manual labor necessary concerning restarting is what the `rename_restart.bat` file is for, removing the old restart file and renaming the new one so that is has the correct file extension.

3.3.3 Sobek restart flaws

Is there such a thing as the perfect, flawless software package? Probably not. The question here is not regarding the accuracy or robustness of a numerical scheme but the occurrence of rather small and seemingly insignificant programming errors.

While working with Sobek and OpenDA, two errors of this kind have been uncovered. The first error has been named "restart error", a name given because it first appeared when using Sobek restarts in OpenDA. The other is a pump error, this one has to do with the pumps that can be defined in a Sobek network. Both of these errors are still present in Sobek, however, a workaround for both of them has been devised to minimize or even negate the effect those errors will have on the OpenDA simulations.

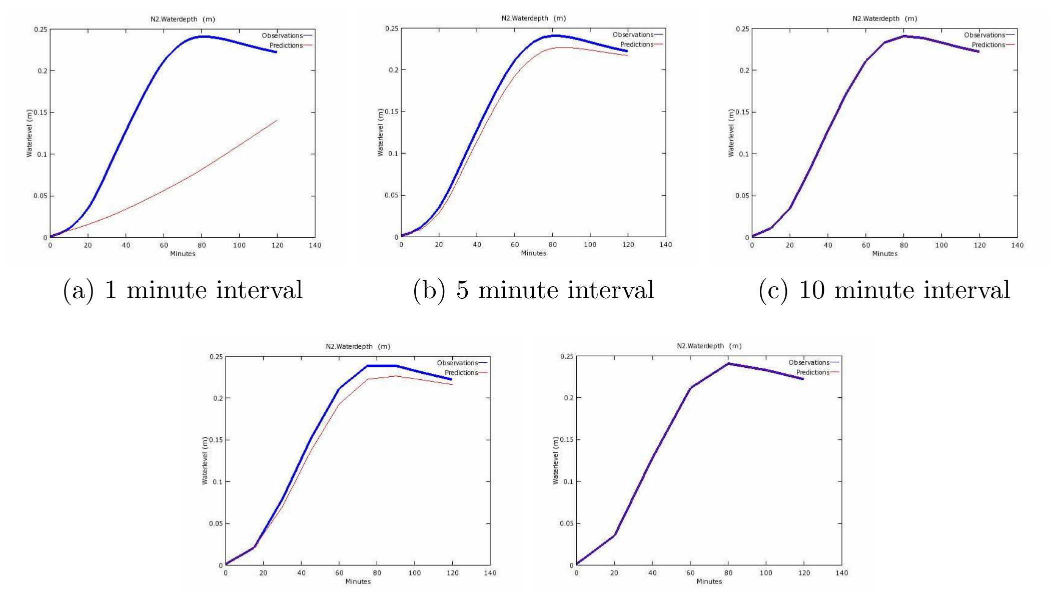
Restart error

The restart error has been investigated in the internship report [Mulder, 2013]. The problem first appeared as a discrepancy between a Sobek simulation and a restarted

OpenDA-Sobek simulation. Given times t_i, $i = 0, 1, \ldots, N$. The Sobek simulation would run from t_0 to t_N, the restarted simulations with OpenDA and Sobek would run from t_0 to t_1, restart files will be written at this point and reused in the next simulation timespan, t_1 to t_2, and so on until the end time t_N is reached.

A lot of effort went into finding the cause of this error. Starting with defining a precise time span for the simulations. The precipitation event, and thus the time span used in the simulation, has a duration of two hours and starts at january 1st 1995, at 00:00 (hh:mm). The times are defined in minutes from start, thus $t_0 = 0$, $t_N = 120$, intermediate times and the value of N will vary. The restart interval length will be denoted in minutes and constant over the entire run, meaning $h_t := t_i - t_{i-1} = constant$ for all $i = 1 \ldots N$. h_t will only take integer values. The very first simulations in OpenDA were done with $h_t = 3$, and it was after the first interval that differences in results were noticed.

Simulations with different time intervals were run and compared to a full (without restarts) run. In figure 3.5, taken from [Mulder, 2013], the results for five different time intervals is plotted.



(a) 1 minute interval (b) 5 minute interval (c) 10 minute interval

(d) 15 minute interval (e) 20 minute interval

Figure 3.5: Results of simulations using different restart time intervals.

In these figures, the full run is the blue graph, the restarted run is the red line. If the problem would be an error like missing a certain time step at restart points, then common sense dictates it would appear no matter the length of the time interval. However, the graphs in figure 3.5 contradict this. Figures 3.5a, 3.5b and 3.5d exhibit faulty behaviour, whereas in figures 3.5c and 3.5e the restarted run results match the full run results.

Further research reveals that the error has actually nothing to do with the fact that it concerns a restarted run or that the simulations are performed from OpenDA (using Sobek). In fact, when using a different start time for a simulation in Sobek, for example $t_0 = 5$, Sobek uses incorrect precipitation information. With this in mind, the conclusion is that this cannot be fixed at this time. There is, however, a way to avoid this error by avoiding (re)starting at any of the times that give a bad result. Thus the simulations with OpenDA will be done starting at $t_0 = 0$ and an

interval time of $h_t = 10$ as to avoid any of the "bad" starting times.
The suggested workaround for the start and interval times has been applied throughout this report.

Pump error

Another error that came into play when running simulations with the sewer system model of Delft concerns pumps. The pumps in the model of Delft are structures that link two nodes and are used to transfer water from one node to another.

The pumps in the model of Delft are links between two nodes. Each link has a direction in the network, but this is not necessarily the pump direction. The pump direction can be positive (in the direction of the link) or negative (oposite to the direction of the link). This defines the suction side and the delivery side. The pumps all have a single stage when it is on, meaning it is either on or off. For this stage there is a certain capacity that the pump can handle, given in m^3/s. Activation or deactivation of a pump depends on the waterlevel at the suction side. The pump is activated if the waterlevel at this node rises above the defined switch-on level. Then it will remain on until the waterlevel has dropped below the switch-off level.

The problem with the pumps occurs when making use of restart files. The restart files seem to be missing information on the status of the pump at the last known time, whether it was on or off. So the pump status is evaluated using the current waterlevel at the suction side instead. It is then put on if the waterlevel is above the switch-on level and it is set to off otherwise. If the pump was on at the end time of the previous interval, but the waterlevel has dropped below the switch-on level, it is switched off at the next time step. A simple and clear visualization of this event can be seen in figure 3.6.

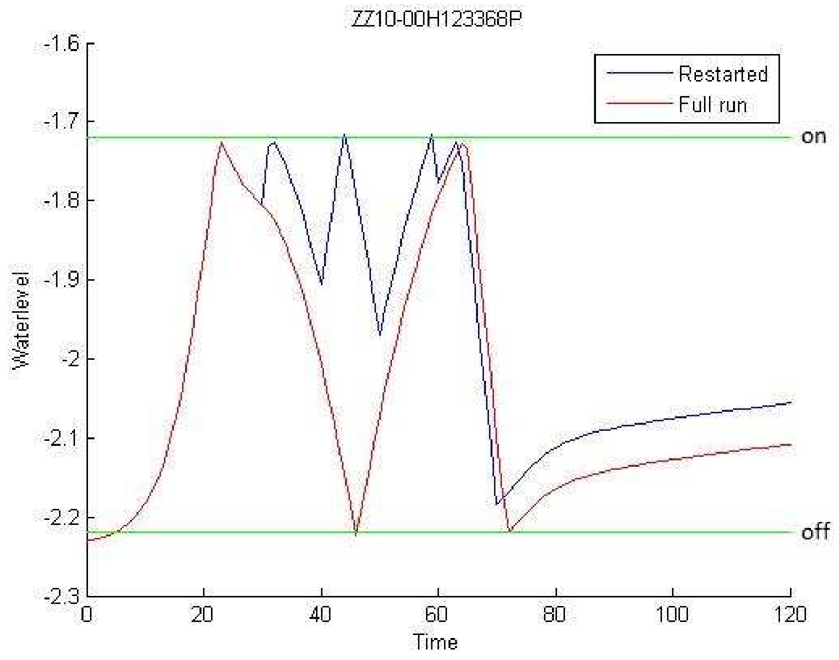


Figure 3.6: Graph of the waterlevel at the suction side of a pump exhibiting an error.

This figure is a good representation of the error. The switch-on and switch-off levels are represented by the green horizontal lines, these are obviously constant in time. The full run is represented by the red line and the restarted run by the blue line.

The first difference in the two runs occurs at $t = 30$. The waterlevel just before this time was dropping, because the pump was turned on. However, $t = 30$ is a restart time, so when the restarted run continues, the pump state is evaluated by the current waterlevel which is below the switch-on level. And so for the restarted run, the pump is set to off and the waterlevel starts rising again, explaining the difference between the two runs. From there on out, it can be seen that the pump switches on when it reaches the switch-on level and it switches off either when it reaches the switch-off level (which never happens in the restarted run) or when the run is restarted.

This error concerning the pumps is still present in Sobek, however, a way to avoid the effects of this error on the OpenDA simulations has been implemented. Instead of taking the observations from the Sobek simulation directly, these observations are taken from a stochastic run with OpenDA. This means that if the node plotted in figure 3.6 is an observation point, the observed waterlevels would be taken from the restarted OpenDA simulation (blue line) instead of the original Sobek simulation (red line).

3.4 The network of Delft

The model that is used throughout this thesis is the sewer network of Delft, part of which has already been shown in figure 3.4. The "full" network, provided by [Post, 2012], that is used can be seen in figure 3.7.

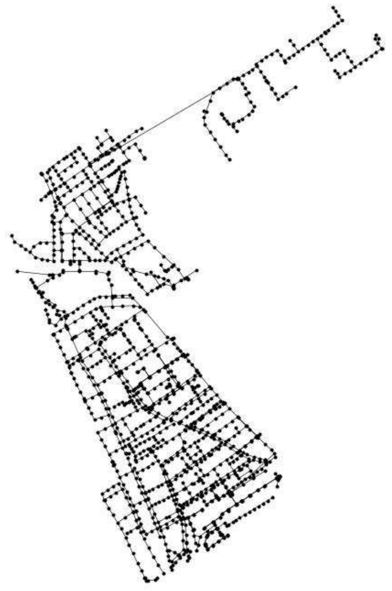


Figure 3.7: Full sewer network of Delft as it is used in all simulations.

This figure shows the network, as it is used in the flow module, but there is also a part in the model, used in the rainfall runoff module, that is a bit harder to represent in this figure. Most of the nodes shown are manholes with runoff and for this runoff to make sense, there has to be some amount and type of surface area defined for each of those nodes. As it turns out, the network of Delft only uses four of the twelve possible surface areas. The total amount of surface area can be seen in table 3.3

Total inhabitants	13850
Total surface area	893650
Flat closed paved	58240
Flat open paved	396570
Sloped roof	406540
Flat roof	32300

Table 3.3: Details of the model of Delft

Another attribute in a network like this is the number of inhabitants. This number influences the amount of (waste)water to enter the sewer system, which is known as dry weather flow. In the network of Delft the dry weather flow is defined as the number of inhabitants multiplied with a constant amount. The number of inhabitants is distributed over the nodes in the system.

Chapter 4

Calibration

The sewer network model of Delft contains many (constant) parameters. Some of these parameters have values based on the specific network of Delft, such as the amount of surface that contributes to the runoff into the sewer system, and other parameters that are more general like runoff coefficients belonging to different surface types. An important part of applying data-assimilation to any model is the calibration of such parameters. No matter how well the parameters have been determined from a theoretical point of view, in practice there can be differences. This chapter will deal with such parameters and investigates the calibration of these parameters by means of performing twin-experiments. The method of calibration that is applied is the DuD algorithm discussed in section 2.4. The applied stopcriteria are like the ones at the end of section 2.4 where $T_1 = T_2 = 0.001$, the maximum number of iterations for the outer loop is 15 and the maximum number of iterations for the inner loop, i.e. the line search part, is 5.

The first parameter which will be calibrated is the amount of surface area in the runoff model. Delft is a rather flat city containing a lot of surface water (canals) all throughout the city centre and a lot of surface area defined in the runoff model is very close to this surface water. Which leads to the question: does all of the defined surface area in the model actually produce sewer inflow, or is part of this surface area running off straight into the surface water? This question is the base of section 4.1, where the calibration of the amount of surface area is investigated by doing a twin-experiment. Note that the results will give no information about the actual surface areas in the model, but merely an indication of the performance of the calibration method. In order to determine the real parameters for the model of Delft, real data would be needed as well as an extensively tested data-assimilation and calibration method.

Another parameter that can be calibrated is at what rate precipitation will runoff into the sewer system. Unlike the amount of surface area, this parameter has no influence on the total amount of water that will actually enter the sewer system but merely the speed at which the sewer system fills up. Calibration of this parameter will again be tested by means of a twin-experiment.

Twin-experiment setup
As mentioned, twin-experiments will be performed to investigate the performance of the calibration method. A short summary of the set up of a twin-experiment is shown below.

1. The model is run with the so called "original input", where the input usually

consists of several parameters, initial conditions etcetera. The output of this model run will then serve as measurements or the so called observational data for the model runs where data-assimilation is applied. The observation locations that are used in this specific chapter have been provided by [Post, 2012], this is a set that has been specifically developed for calibration.

2. One or more parameters or initial conditions are perturbed in the input data, yielding a set of initial parameters for the calibration.

3. Starting with these initial parameters, calibration runs are performed in order to calibrate these specific parameters.

4. Waterlevel results and parameter results from the original run are compared with the results from the calibration run.

4.1 Surface areas

In the rainfall-runoff part of the model, different surface areas are defined (sloped roof, paved flat, etc.). The amounts defined in the model resemble the real surface areas in Delft. However, all of the defined surface area in the model will eventually runoff into the sewer system, while in practice it is likely that a part of the surface area will runoff straight into the surface water instead of the sewer system. Finding the amount of surface area that actually contributes is done by introducing the surface areas as parameters and then calibrate them using the DuD method.

The sewer model of Delft consists of over 1300 nodes. Each of these nodes contains 12 different surface types. An exchange item is made for every surface type for every node, yet not every one of these needs to be actually calibrated. First of all it should be noted that the model of Delft consists of mostly flat open paved surface (as in roads) and sloped roof, see also table 4.1. The rest of the total surface area consists of flat closed paved and flat roof area, 8 of the 12 possible surface areas are not present at all in the model. So it is a plausible assumption that only parts of the flat open paved and sloped roof surfaces will eventually runoff into the surface water instead of the sewer system and the other 10 surface area types are disregarded. Another method to reduce the number of parameters that needs to be calibrated is to notice that not all the nodes in the model are close to surface water. It can be assumed that nodes which are not located next to surface water will completely runoff into the sewer system. This will reduce the surfaces to be calibrated to only those which are next to surface water.

Area type	Surface area m^2
flat closed paved	58240
flat open paved	396570
sloped roof	406540
flat roof	32300

Table 4.1: Total surface area in the model of Delft

Setup for surface area calibration

OpenDA allows for basically two different types of parameter modification while calibrating, see section 2.4. The first is to increase or decrease parameters by certain amounts. Different parameters that are grouped together and turned into one parameter would then be modified by the same amount. A small example: three parameters "Node1-surface5", "Node2-surface5" and "Node3-surface5" with values 100, 200, 300 are grouped. This means that in the corresponding parameter vector θ, all three of these surface areas are contained in one element of the vector. If this parameter is then modified by an amount $\lambda = 50$, this means for the individual values

$$\begin{aligned} \text{Node1-surface5:} &\quad 100 \rightarrow 150 \\ \text{Node2-surface5:} &\quad 200 \rightarrow 250 \\ \text{Node3-surface5:} &\quad 300 \rightarrow 350 \end{aligned}$$

The other option is to modify parameter values with a factor. If the uncertainty is $\lambda = 0.1$, this way of modifying the parameter will yield

$$\begin{aligned} \text{Node1-surface5:} &\quad 100 \rightarrow 110 \\ \text{Node2-surface5:} &\quad 200 \rightarrow 220 \\ \text{Node3-surface5:} &\quad 300 \rightarrow 330 \end{aligned}$$

Note that this is the case when different surface areas are gathered and considered to be one single parameter

When dealing with the city of Delft, calibration by a factor would seem more logical. When dealing with different amounts of surface area for each node in the model, it is more likely that a percentage of the surface area will runoff into surface water, rather than a fixed amount regardless of how much area is defined at that particular node. The other effect of perturbing surface areas by a fixed amount is that the relative change can be extremely large. Both methods of perturbing the parameter(s) will be examined.

The results from calibrating the surface areas can be seen in tables 4.2 - 4.7. At the start of a calibration run, the parameter values are modified by either adding an amount to the original values or multiplying the original values by a factor, yielding different input for the calibration compared to the original run. Then the actual calibration is performed. When discussing results for these kind of surface parameters, there is no original parameter value that can be refered to. This, because the surface parameter actually is a group of several surface areas, each with their own original value. For this reason the offset from the original parameter is used, since this offset will be equal for all surface areas within the group. In the calibration results, the value in the column denoted by 'twin param.' is the parameter offset that is used to create the observations. Where 0 (m^2) or 1 (factor) means that this is the parameter value with no offset. These offset values are also the target values for the calibration. If the offset after calibration is the same as the offset of the twin-experiment parameters, then these parameters are the same and the calibration is successful. The column 'Perturbed' contains information about the offset of the start values for the calibration compared to the original values, where a value in m^2 means that in every node the amount of surface area is increased by that amount. When the column contains a factor, this means that the original value is multiplied by this factor, resulting in a smaller area if this is smaller than one and a larger area if the factor is greater than one. The 'calibrated' column is the result from the calibration. This value should be compared to the original.

The first experiment is one where only one surface type is calibrated, this will be the flat open paved surface. There are 77 nodes for which the surface amounts are perturbed and the amount adjusted is the same for each node, see figure 4.1 for a map where the red nodes indicate the ones where calibration will be applied. It should be noted that surface amounts cannot be negative, so when adjusted by a negative amount which would eventually result in a negative surface, that surface is set to 0 instead. For the first calibration, the flat open paved surfaces for all 77 nodes are grouped together and can be considered as one parameter.

The results of the calibration of this surface are displayed in table 4.2. When the surface areas are increased by 50, 100 or 300 m^2 in each node, the calibration yields good results.

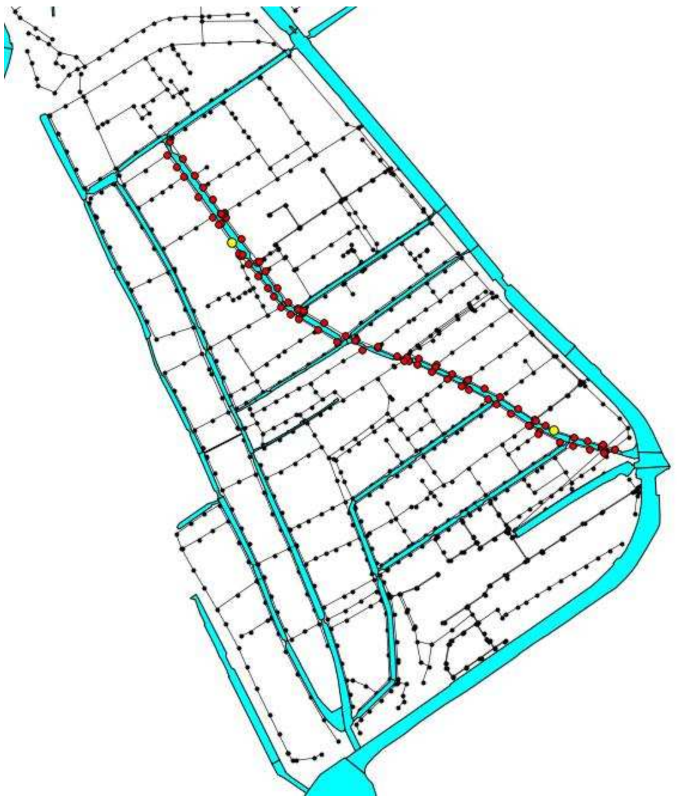


Figure 4.1: Network used for the calibration of surface areas in the simple case. The red nodes are the ones where the surfaces are actually calibrated. The two yellow nodes are the locations used in the visualization of waterlevels in time.

Twin param. offset (m^2)	Initial param. offset (m^2)	Calibrated offset (m^2)
0	50	0.0709
0	100	-0.0477
0	300	-0.0018

Table 4.2: Calibration results for a single surface type (flat open paved). The surface is modified by a fixed amount for the twin-experiment.

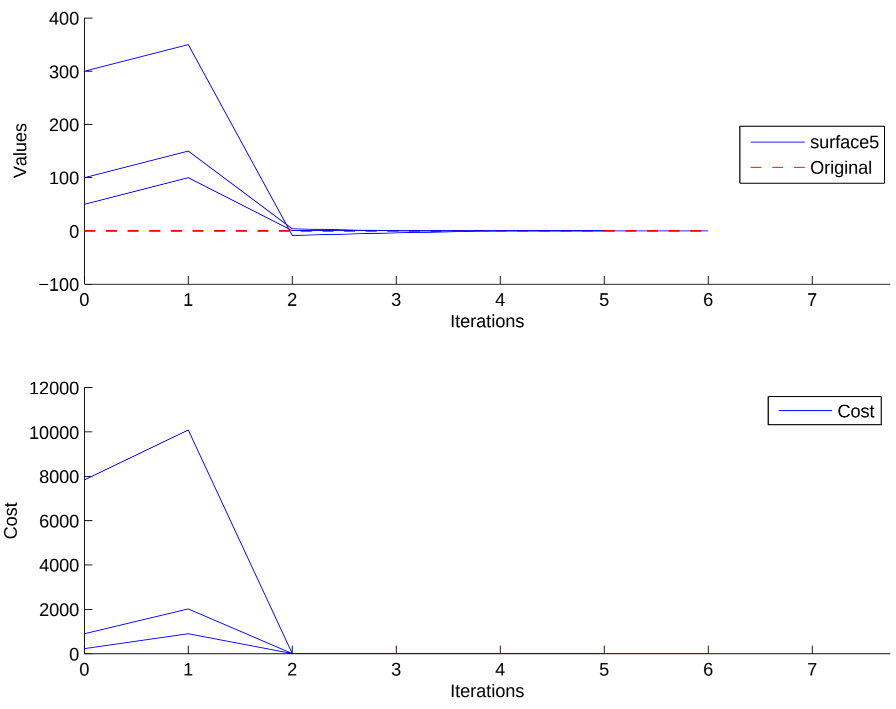


Figure 4.2: Results from the calibration of only surface 5

Figure 4.2 and the table 4.2 show the calibration results for the three different cases. The parameter name 'surface5' is the name that belongs to the flat open paved parameter. Because this first calibration run is done with only a single parameter, it yields results rather quickly. The calibration of a single parameter is just a first step. Next, two paramters will be calibrated and the results discussed.

Throughout this chapter, the waterlevels in two nodes will be visualised whenever the calibration yields some questionable results. Because the algorithm will attempt to minimise a cost function based on residuals of the observed and predicted waterlevels, having a look at the waterlevels when the calibration is completed gives some insight into the performance of this calibration. The location of the nodes of which the waterlevels are visualised can be seen in figure 4.3. The locations are indicated as black dots. In this same figure, the internal weirs that are present in the system are also visualised by the red dots. This to give an indication of the distance of the location of the nodes to the internal weirs, which are in turn connected to the surface water. The first example of such figures can be seen in figure 4.4

In figure 4.4, three graphs are shown of which two coincide. The graph labeled 'original' corresponds to the parameter values of the twin-experiment, with which the observations are created. The 'perturbed' graph corresponds to the initial parameter value for the calibration. The graph labeled 'Calibrated' corresponds to the parameter value after calibration. The calibrated results coincides with the original, which is exactly the goal of the calibration, namely: finding parameters such that the defined cost function is minimal. Where the cost function is defined as in equation 2.17.

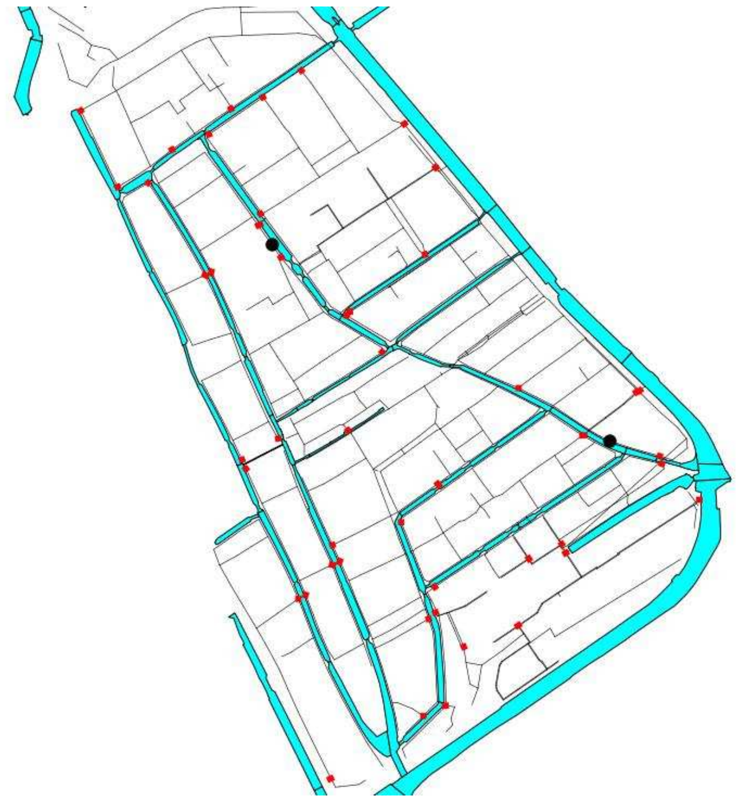
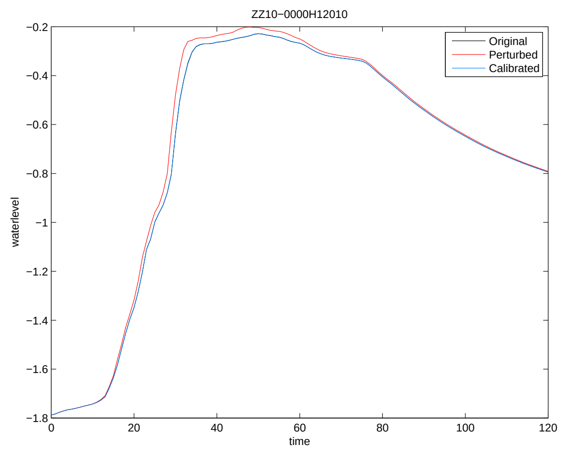
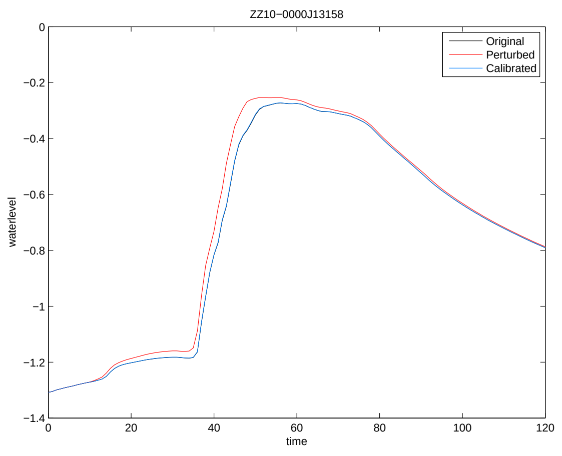


Figure 4.3: The red dots are the locations of the internal weirs. The two black dots are the location of the nodes which have been used to visualise the waterlevels. The figure shows where the internal weirs are with compared to the location of the plotted nodes.



(a)



(b)

Figure 4.4: These figures show the waterlevels in two nodes for three situations: with the original surface area, the perturbed amount and calibrated amount. The perturbation applied is adding 300 m^2 to the flat open paved area, corresponding to the bottom row in table 4.2.

Calibrating a single parameter yields good results, so the next step is to calibrate two different parameters. To this end, two different surface types are used as parameters, sloped roof and flat open paved. The flat open paved surface area contained in several different nodes is again grouped into one parameter and the same is done for the sloped roof areas, which will serve as the second parameter. The above mentioned experiments are repeated, but now for two parameters instead of just one. Both parameters are modified by an amount, independent of eachother, and then calibrated, yielding the calibration of the parameter vector for different starting vectors. It can be argued that both parameters will have roughly the same effect on the simulation, since both represent surface area that is used in the rainfall-runoff module to simulate the amount of water that ends up in the sewer system. The only difference between these parameters being the delay with which the water ends up in the system. Nevertheless, the calibration is carried out and the results can be seen in tables 4.3 and 4.4.

Twin param. offset (m^2)		Initial param. offset (m^2)		Calibrated offset (m^2)	
f.o.p.	s.r.	f.o.p.	s.r.	f.o.p.	s.r.
0	0	50	50	0.0662	-0.0288
0	0	50	100	0.0621	-0.0647
0	0	100	50	-21.8619	8.0368
0	0	100	100	131.0344	-121.7470
0	0	300	300	27.9488	-15.3364

Table 4.3: Calibration results for two surface types (flat open paved (f.o.p.) and sloped roof(s.r.)). The surfaces are modified by a fixed amount for the twin-experiment.

The calibration of these parameters is better visualised in figure 4.5. Each subfigure corresponds to a row in table 4.3. The x-axis in the figure represents the number of iterations it took for the algorithm to reach the calibration result as it can be seen in the table. The number of iterations is the total of all iterations, inner and outer iterations of the DuD algorithm. Outer iterations are the updates of the parameter vector and the inner iterations are the linesearch iterations as described in section 2.4.

From these five different starting vectors, the first two are calibrated quite well. However, the last three results catch the eye. In the table, row 3, 4 and 5 show that the calibration does not return the twin parameter offset (with which the observations were created). For the calibration results or row 4 and 5, figure 4.6 shows the resulting waterlevels in two nodes. From these figures quickly follows that the perturbation applied in the parameters does not have any drastic effect on the waterlevels in these two nodes. The calibration, despite not being perfect, does result in a very close match of the calibrated waterlevels with the original waterlevels.

The calibration of the surface areas as it is described above works quite well. It is very interesting to see that in table 4.3 the 4th row shows calibration values that deviate further from the original values, leading to believe that the calibration fails. However, when visualising the waterlevel results in the reference nodes, figure 4.6, the waterlevels belonging to the calibrated values are very close to the original waterlevels, which is an indication that the cost function, which is based on the sum of squares of the residuals, has decreased.

When looking at the calibration results of row 4 in table 4.3 the assumption is that these two parameters have a high, be it negative, correlation. Meaning that if one is increased by a certain amount and the other is decreased by roughly the same amount, the result will be approximately the same as before the perturbation.

All in all, the calibration is succesful in finding a set of parameters in such a way that a stop criterion is satisfied. The parameters that are found are not necessarily the same as the ones used in the twin with which the measurements were created, as can be seen in table 4.3.

When applying a perturbation with a factor, table 4.4 and figure 4.7, the perturbation has little impact on the waterlevels as can be seen in the graphs of the waterlevels, which is likely due to the difference between the twin parameter and the initial parameter for the calibration not being high enough. A larger offset for the initial parameter to start the calibration with would result in larger differences between waterlevels, however, perturbing parameters too much will not be very realistic. Based on the result of this calibration there are no ground breaking conclusions to be made.

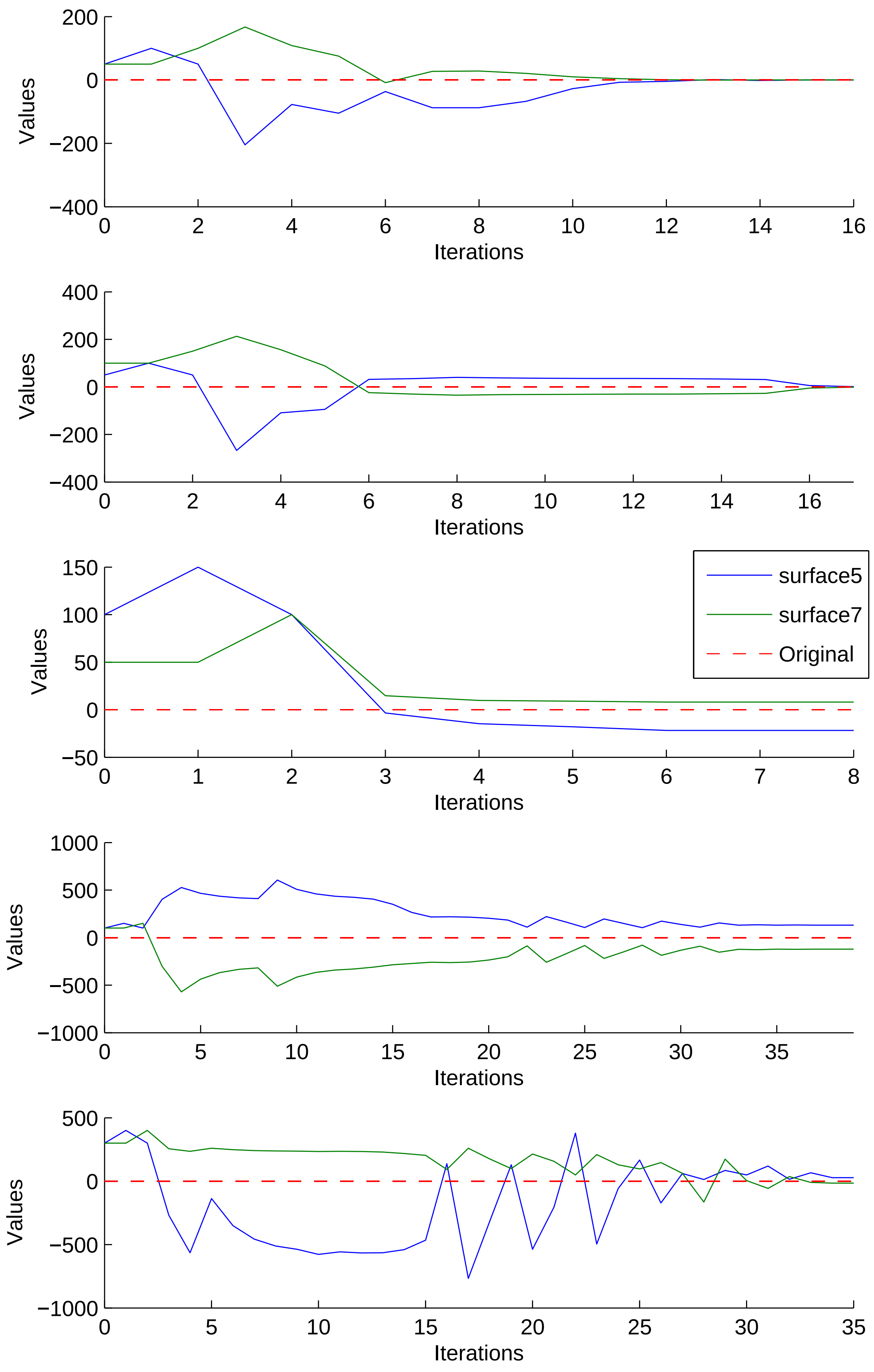
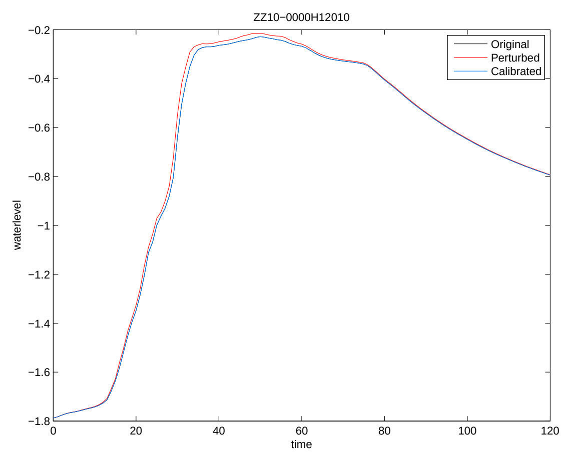
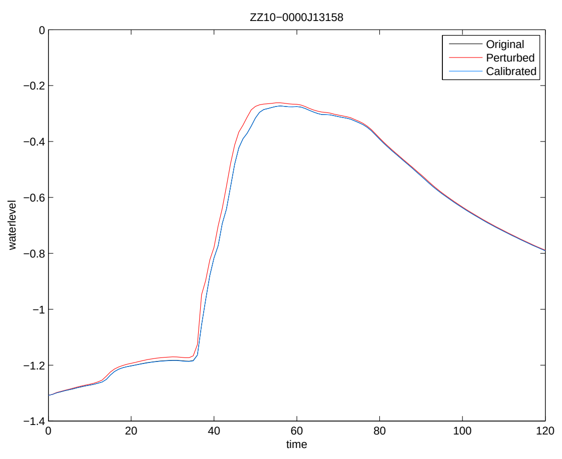


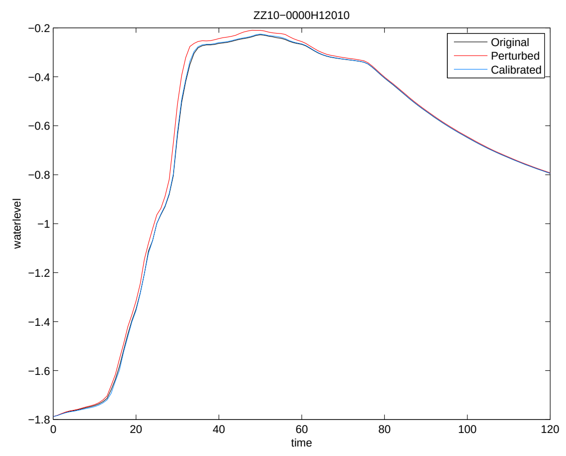
Figure 4.5: Results from the calibration two surfaces, namely surface5 (flat open paved) and surface7 (sloped roof)



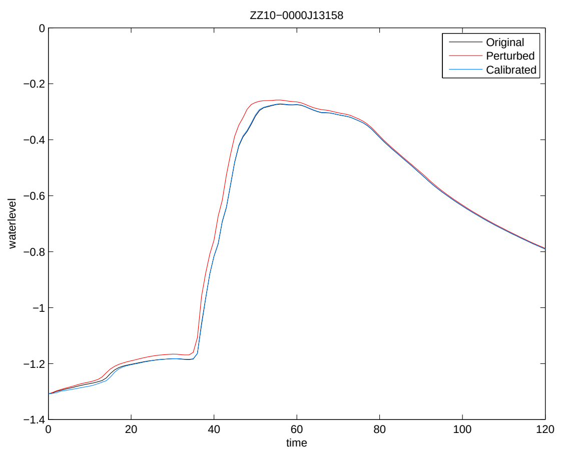
(a)



(b)



(c)

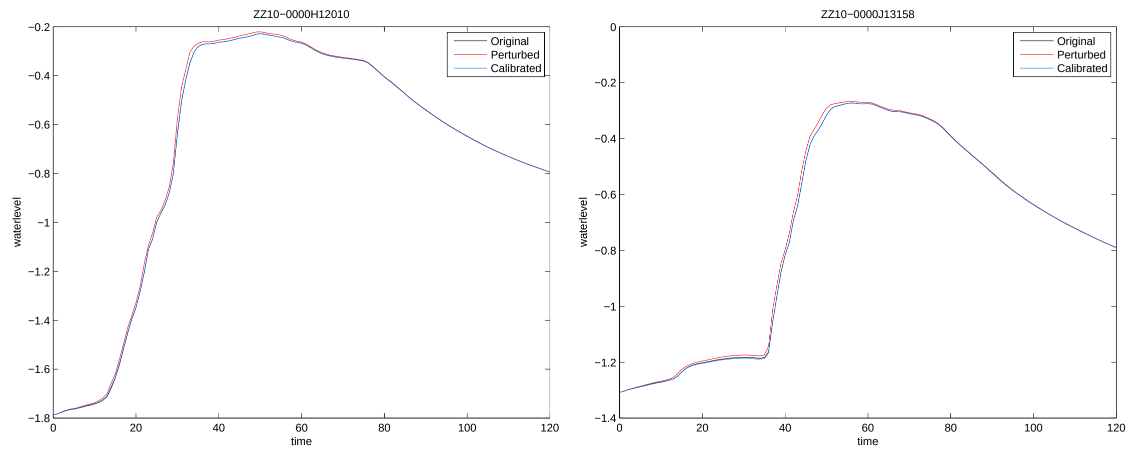


(d)

Figure 4.6: Visualisation of waterlevels in 2 different nodes in the case of 2 different starting values for the calibration. The top figures, (a) and (b), correspond to row 3 of table 4.3. Figures (c) and (d) correspond to row 4 of table 4.3.

Twin param. offset (factor)		Initial param. offset (factor)		Calibrated offset (factor)	
f.o.p.	s.r.	f.o.p.	s.r.	f.o.p.	s.r.
1	1	1.1	0.9	1.1104	0.9648
1	1	0.9	1.1	0.90046	1.0263
1	1	0.9	0.9	0.9468	1.0017
1	1	1.2	1.2	1.08	0.96

Table 4.4: Calibration results for two surface types (flat open paved (f.o.p.) and sloped roof (s.r.)). The surfaces are modified by a factor for the twin-experiment.



(a) (b)

Figure 4.7: Waterlevel results in the case that two different parameters are calibrated, corresponding to the bottom row of table 4.4, where a perturbation is applied by multiplying the original amount with 1.2.

Extending the surface area calibration

With good results for the smaller calibration case, it is time to include all the nodes in the city centre of Delft that are next to the surface water. The map indicating all such nodes can be seen in figure 4.8. Involving more nodes in the perturbation will result in a larger difference in waterlevels between the original and perturbed simulations which means larger residuals and so a higher value for the cost function. This should be very obvious when looking at the resulting waterlevels in the calibration.

As an initial test of the calibration of this extended surface area, the calibration is again performed on a single parameter, namely the flat open paved surface area. The calibration yields some good results as can be seen in table 4.5.

The calibration is again expanded to include two different parameters. Starting with several different values for the initial paramters, the results can be seen in table 4.6. Overall the results look good, however, the last two rows of this table would indicate a less succesful calibration. To illustrate the effects of the calibration results in the last two rows of this table, the waterlevels in two nodes have been plotted. The graphs of these waterlevels can be seen in figure 4.10. These figures show that, despite the calibration not yielding the desired parameters, the resulting parameters do make sure for a very good match between the waterlevels of the original simulation and the simulation with the calibrated parameters.

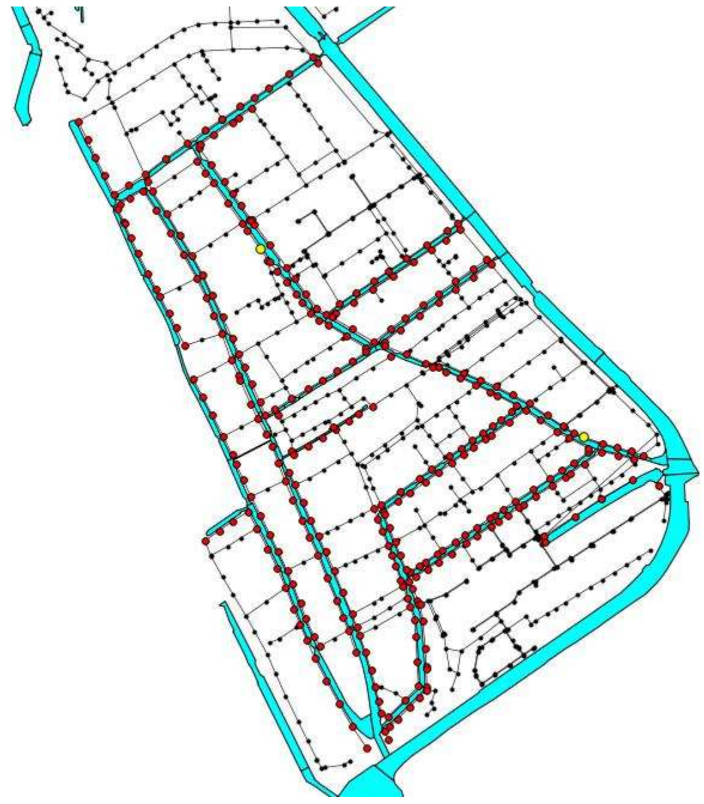


Figure 4.8: Network used for the calibration of surface areas in the full case. The red nodes are the ones where the surfaces are calibrated. The yellow nodes are again the locations that are used in the visualization.

Twin param. offset (m^2)	Initial param. offset (m^2)	Calibrated offset (m^2)
0	50	0.0136
0	100	0.0298
0	300	0.0357

Table 4.5: Calibration results for a single surface type (flat open paved). The surface is modified by a fixed amount for the twin-experiment.

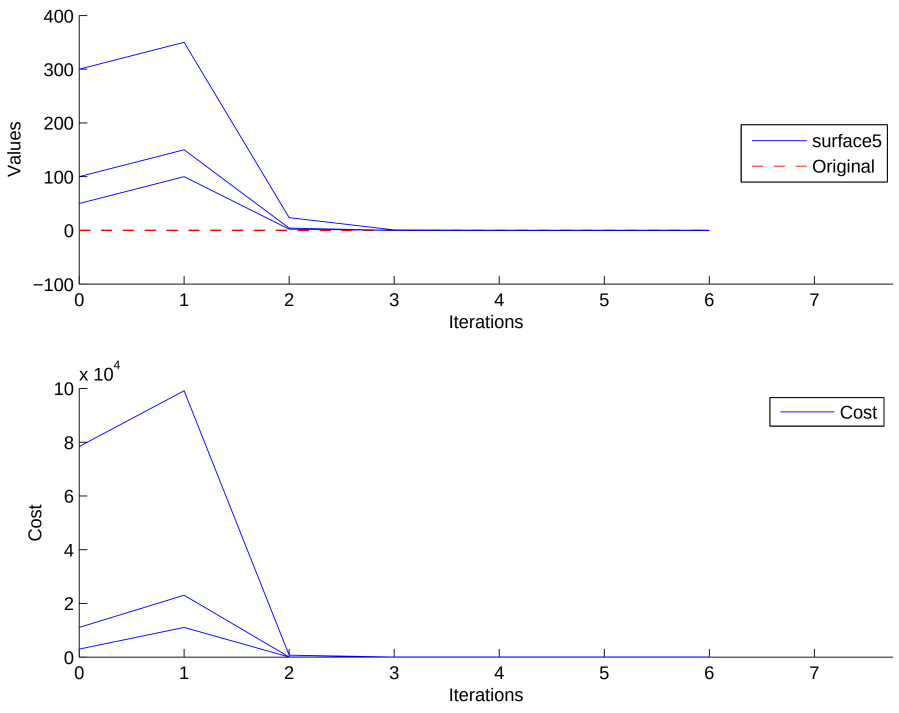


Figure 4.9: Results from the calibration of only surface 5

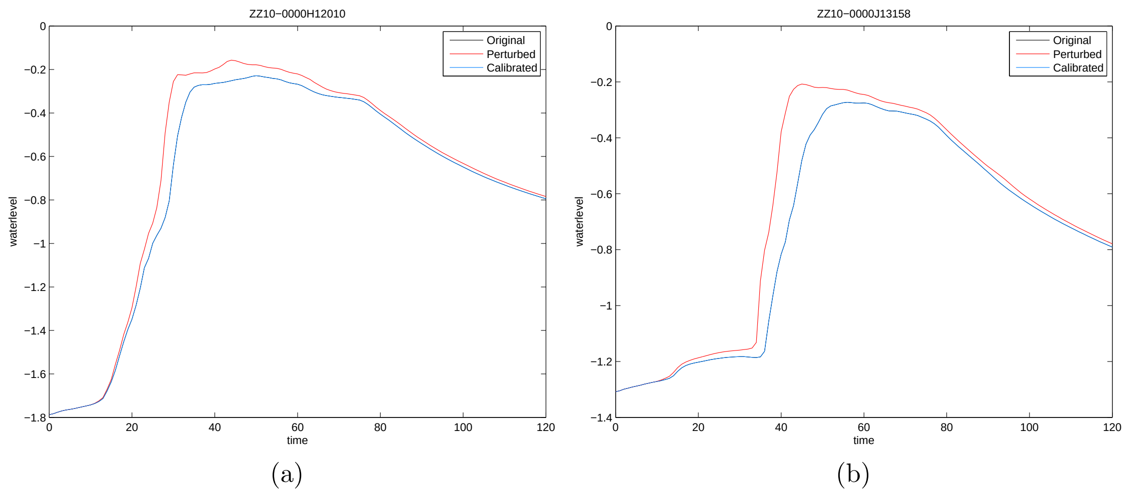
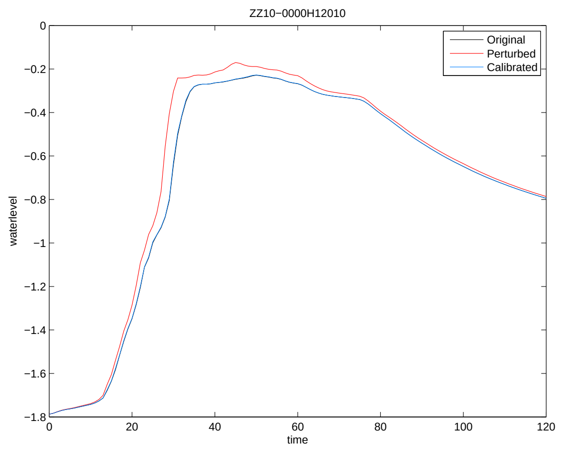


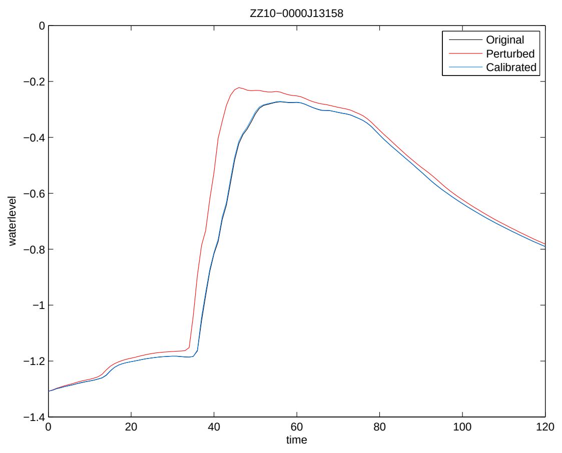
Figure 4.10: The waterlevels in the case of the calibration of a single surface area parameter.

Twin param. offset (m^2)		Initial param. offset (m^2)		Calibrated offset (m^2)	
f.o.p.	s.r.	f.o.p.	s.r.	f.o.p.	s.r.
0	0	50	50	-15.9527	8.9122
0	0	50	100	0.9031	-0.1437
0	0	100	50	0.8822	-0.9312
0	0	100	100	-35.5291	-18.7187
0	0	300	300	-252.4313	77.8533

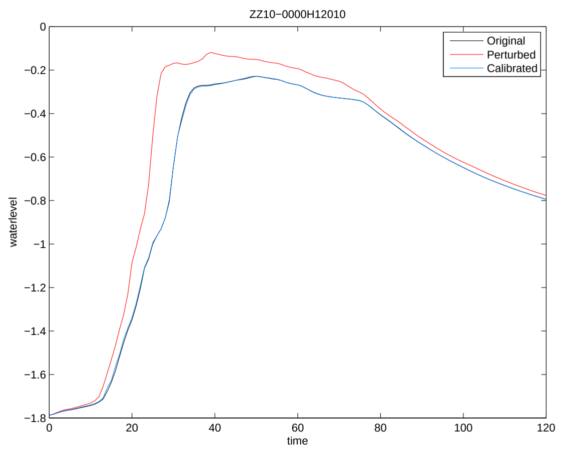
Table 4.6: Calibration results for two surface types (flat open paved (f.o.p.) and sloped roof (s.r.)). The surfaces are modified by a fixed amount for the twin-experiment.



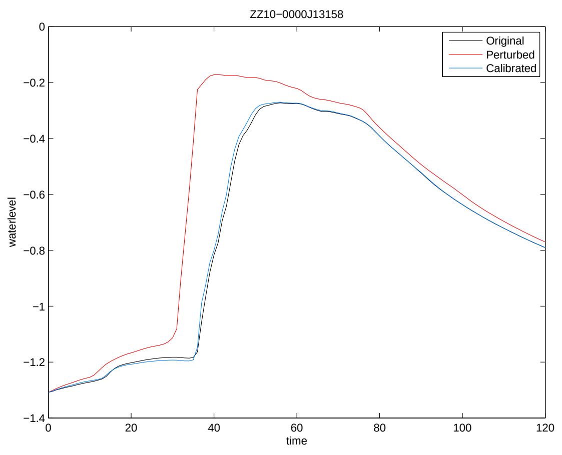
(a)



(b)



(c)



(d)

Figure 4.11: Resulting waterlevels in two nodes when calibrating two surface area parameters. The figures (a) and (b) correspond to the 4th row of table 4.6 and figures (c) and (d) correspond to the 5th row of table 4.6.

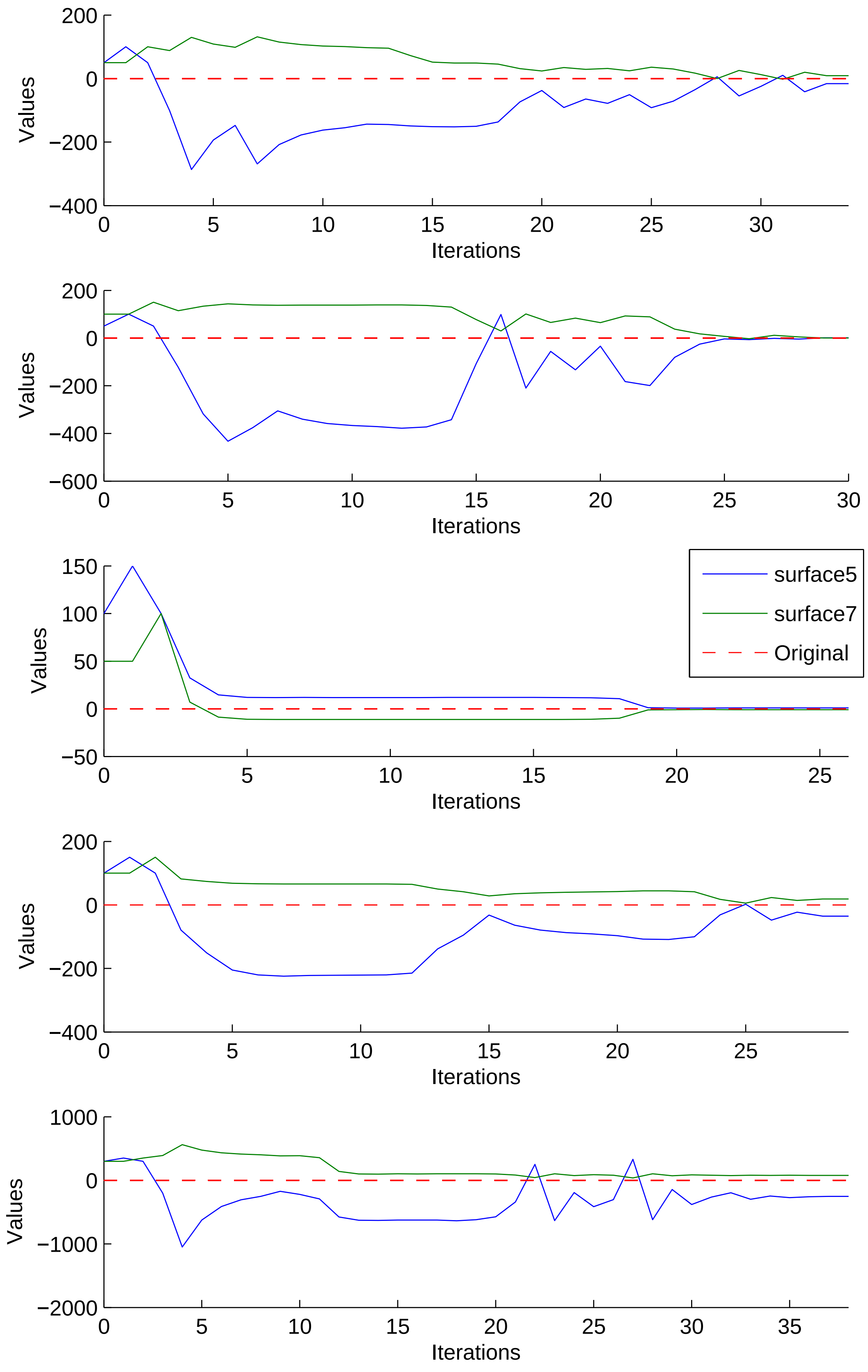


Figure 4.12: Results from the calibration two surfaces, namely surface5 (flat open paved) and surface7 (sloped roof)

Figure 4.12 shows the calibration of these two parameters as the number of iterations progresses. Again, the number of iterations is the total of the inner and outer iterations as performed by the DuD method. Each subfigure corresponds to the specific row in table 4.6

Twin param. offset (factor)		Initial param. offset (factor)		Calibrated offset (factor)	
f.o.p.	s.r.	f.o.p.	s.r.	f.o.p.	s.r.
1	1	0.9	0.9	0.8583	1.0366
1	1	1.1	1.1	0.9994	0.9962
1	1	1.05	1.2	0.8514	1.1027
1	1	1.2	1.05	1.4745	1.1916
1	1	1.2	1.2	0	1.2648

Table 4.7: Calibration results for two surface types (flat open paved (f.o.p.) and sloped roof (s.r.)). The surfaces are modified by a factor for the twin-experiment.

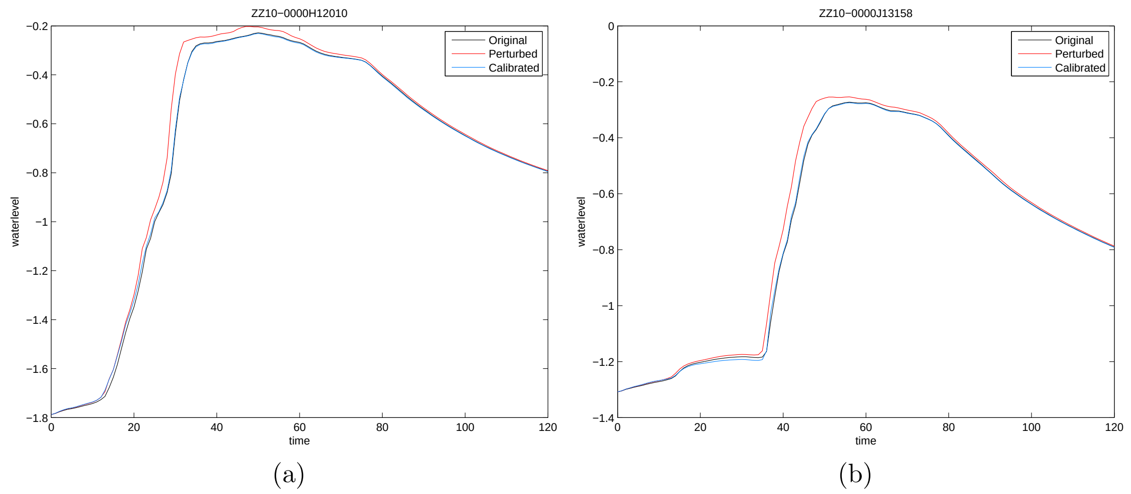


Figure 4.13: Resulting waterlevels when calibrating two surface area parameters. Eventhough the flat open paved surface area is set to 0, the waterlevel results are quite good.

Calibration by means of absolute amounts is the prefered method. Modifying parameters by a factor can be useful for, among other things, sensitivity analysis. Since multiplying a set of parameters by 1.1 or 1.05 means a change of 10% or 5% respectively, which in turn leads to an analysis of the residuals when a relative change in parameters occurs. Besides surface areas there are other parameters present that can be calibrated. Some of which are treated below.

4.2 Rainfall runoff

Since there are 12 different surface area types in Sobek, there also are 12 different runoff parameters, that is one for each defined surface area. These runoff parameters are an indication of the delay with which water ends up in the sewer system. However, at first only two runoff parameters will be calibrated and the others will be disregarded. The parameters that are to be calibrated are the ones that belong to the surface areas that are calibrated in section 4.1, for the reason that the model consists of mostly surface area is of these types (flat open paved and sloped roof). Further down the road, the calibration will take care of four different runoff parameters. These are all the runoff parameters that have any influence on the model. Since runoff parameters belong to certain surface areas, these are listed in table 4.8

Runoff parameter	Area type	Surface area m^2
rf2	flat closed paved	58240
rf5	flat open paved	396570
rf7	sloped roof	406540
rf8	flat roof	32300

Table 4.8: A list of the different runoff parameters with corresponding surface areas and their amounts.

Calibration of a single runoff parameter

Starting with the calibration of a single runoff parameter to see how well it performs. The runoff parameter to be calibrated is the one belonging to the sloped roof. The value of this parameter, as with all runoff parameters, will be between 0 and 1. The calibration is done by means of a twin experiment. First, the model is run with a certain value for the parameter which will be refered to as the original value in order to create the measurements. This parameter value will then be (slightly) modified, this is the perturbed value and initial value for the calibration, and an attempt to calibrate it with OpenDA will be made. The calibration result is the parameter value after calibration. In a perfect case, this calibration result will match the original value (twin parameter value). The results of calibrating this single parameter can be seen in tables 4.9 and 4.10.

Twin param. value	Initial param. value	Calibrated value
0.2	0	0.2
0.2	0.1	0.2
0.2	0.3	0.2
0.2	0.4	0.201

Table 4.9: Results from calibrating the flat open paved runoff parameter.

Twin param. value	Initial param. value	Calibrated value
0.5	0.3	0.4999
0.5	0.4	0.4999
0.5	0.6	0.4806
0.5	0.7	0.5007

Table 4.10: Results from calibrating the sloped roof runoff parameter.

At this point, calibration of the runoff parameters for both the flat open paved and sloped roof seems to work quite nicely. As with the surface area parameters, calibration of a single parameter does not pose a challenge. The more interesting part is the calibration of several parameters simultaneously.

Calibrating multiple runoff parameters

With the nice results above, it is time to investigate the performance of the calibration for multiple runoff parameters. To this end, both the flat open paved and the sloped roof parameter will be perturbed and attempts to calibrate these will be made. The results from the calibration of both parameters is shown in table 4.11. In this table the runoff parameter for the flat open paved is indicated by rf5 and the sloped roof parameter is indicated by rf7. These names are a result from the input file for Sobek, where they are simply the 5th and 7th number after the indicator "rf" for the series of 12 runoff parameters.

Twin param. values		Initial param. values		Calibrated values	
rf5	rf7	rf5	rf7	rf5	rf7
0.2	0.5	0.4	0.3	0.1998	0.5002
0.2	0.5	0.3	0.4	0.2003	0.4986
0.2	0.5	0.1	0.6	0.1995	0.5021
0.2	0.5	0	0.7	0.2026	0.4745
0.2	0.5	0.4	0.7	0.1302	0.4987
0.2	0.5	0.3	0.6	0.1992	0.5022
0.2	0.5	0.1	0.4	0.2006	0.4978
0.2	0.5	0	0.3	0.1990	0.5080

Table 4.11: Results from calibrating the flat open paved and sloped roof runoff parameters.

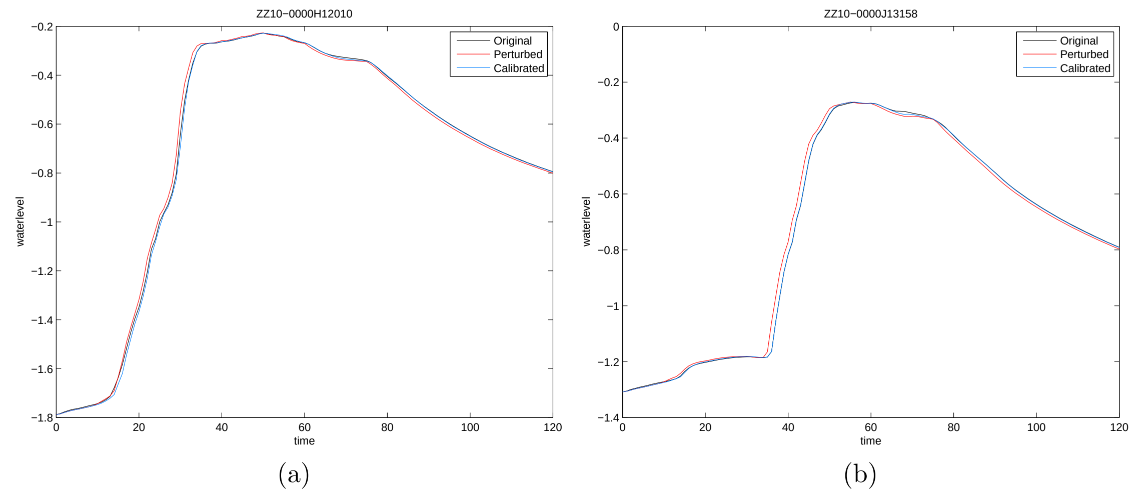


Figure 4.14: Calibration of two runoff parameters (rf5 and rf7). Both were perturbed by an amount -0.2, yet the perturbation does not seem to have a significant result in the waterlevel. For the calibration result, the waterlevel pretty much matches the original values.

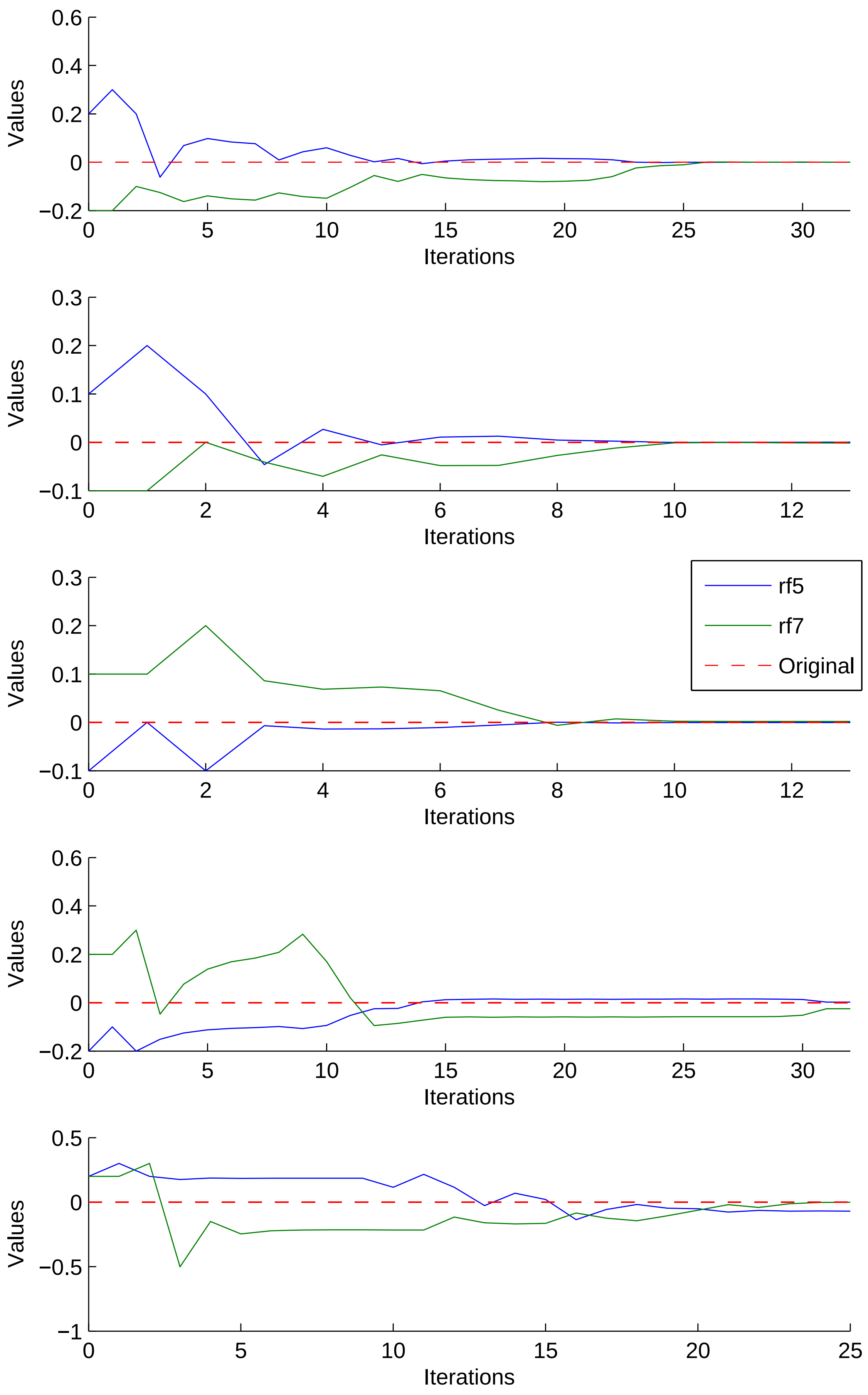


Figure 4.15: Figure showing the calibration of two runoff parameters, rf5 and rf7.

The results displayed in table 4.11 show that the calibration of both parameters performs quite well. Compared the the results for two differens surface types in the previous section, the results for the rainfall runoff are much better. Because these results are quite promising, the choice can be made to include all present runoff parameters in order to test the calibration of all of them simultaneously.

Calibration of all runoff parameters

The runoff parameters that have the most impact on the system have been calibrated above. Below, an attempt is made to include the other two runoff parameters that are influencing the model as well. Because the total amount of different surfaces that are defined in the model is four, the number of runoff parameters is equal to this as well. Figure 4.16 shows the calibration of three different starting vectors for these four runoff parameters. Again, the deviation (or offset) from the original parameters is used.

Twin param. values				Initial param. values				Calibrated values			
rf2	rf5	rf7	rf8	rf2	rf5	rf7	rf8	rf2	rf5	rf7	rf8
0.2	0.2	0.5	0.2	0.3	0.1	0.6	0.1	0.2077	0.1974	0.5090	0.2080
0.2	0.2	0.5	0.2	0.3	0.3	0.4	0.3	0.1782	0.2056	0.4845	0.2514
0.2	0.2	0.5	0.2	0.4	0.4	0.3	0.4	0.1256	0.2545	0.3818	0.8768

Table 4.12: Results from the calibration of four runoff parameters.

The starting vectors used in the calibration were chosen randomly and yield some interesting results. The first row shows that the calibration works quite well. In this row, the calibrated parameter vector is very close to what it should be. The calibrated parameter vector in the second row deviates a bit more from the twin vector. However, the elements 'rf5' and 'rf7' are quite well calibrated. The third row shows the most interesting behaviour. The calibrated vector, and in particular the element 'rf8', shows a large deviation from the twin vector. At this moment, a clear explanation for this very large deviation cannot be given, however, it would be closely related to individual parameters not being perfectly identifiable and there being some correlation between the parameters.

Figure 4.16 shows the calibration for the three parameter vectors as the algorithm progresses. It shows that for the first two parameter vectors, the calibration attains some value for all parameters which is then more or less constant over the next few iterations and then the algorithm is stopped. The third row, with the eventually large deviation in 'rf8', shows that this parameter is only adjusted to this 'extreme' value somewhat at the end of the calibration.

To gain more insight in the parameters, some identifiability analysis can be performed, which is done in the next section.

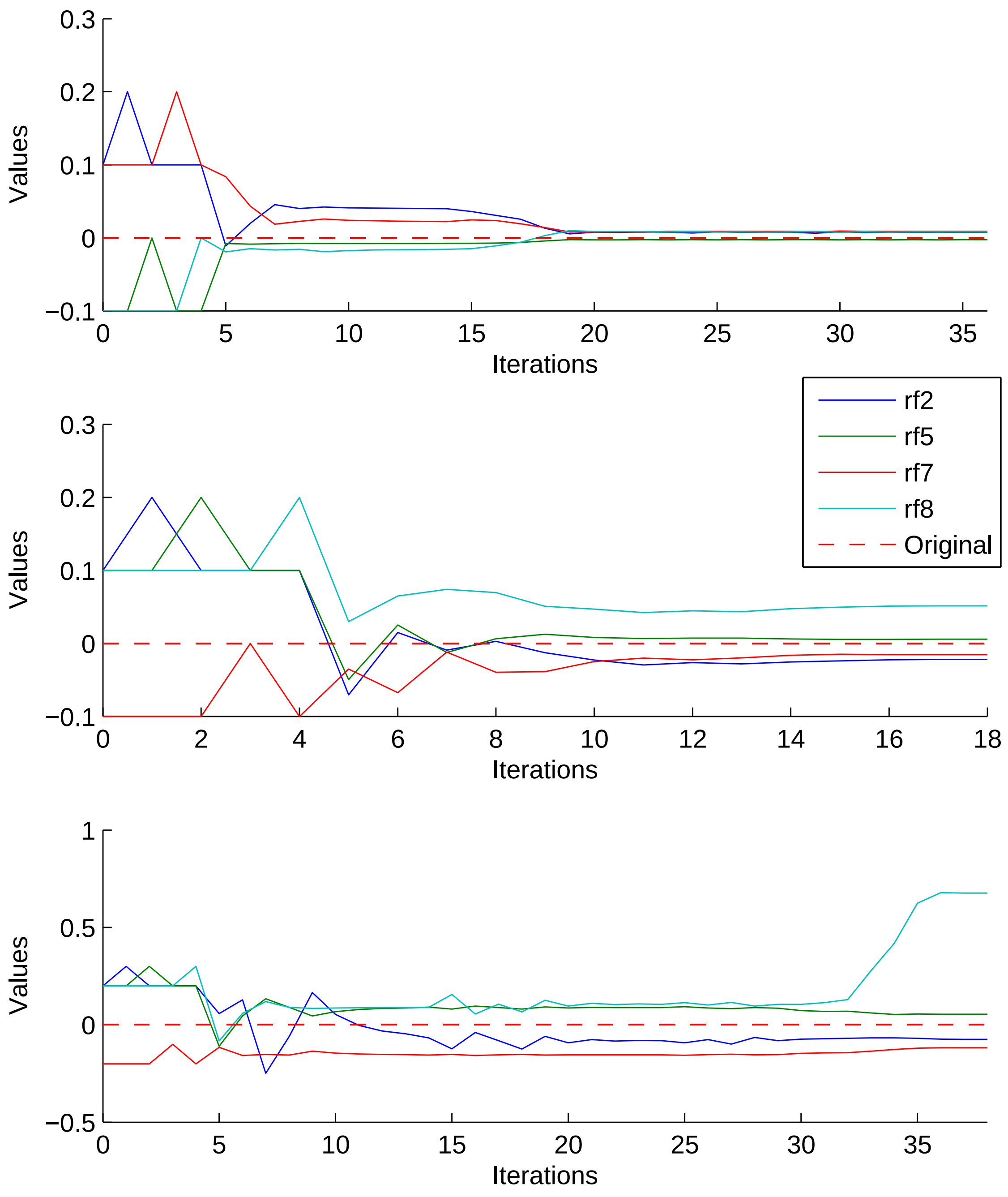


Figure 4.16: Figure showing the calibration of four runoff parameters (rf2, rf5, rf7 and rf8)

4.3 Parameter analysis

As mentioned in section 2.5 the analysis of estimated parameters is an area of interest. Once the parameter vector has been succesfully calibrated, the method described in section 2.5 can be used to visualise the correlation between different parameters and the identifiability of parameters or parameter sets.

Four runoff parameters

The first parameter vector that will be analysed is the one resulting from the calibration of four runoff parameters. Specifically the one corresponding to the first row in table 4.12 and the top graph in figure 4.16. The corresponding calibrated parameter vector is $\theta_0 = [0.2077 \quad 0.1974 \quad 0.5090 \quad 0.2080]$. In order to calculate the Jacobian, the model is run four times, each time perturbing just one parameter by about 5%. Singular value decomposition is applied to the Jacobian which results in the matrix Σ with the singular values on the diagonal and the matrix V whose column vectors consist of the eigenvectors of $J^{\mathrm{T}}J$. The matrices Σ and V can be seen in equation (4.1)

$$\begin{aligned} \Sigma &= \begin{bmatrix} 60.4342 & 0 & 0 & 0 \\ 0 & 33.3956 & 0 & 0 \\ 0 & 0 & 7.3339 & 0 \\ 0 & 0 & 0 & 3.2144 \end{bmatrix} \\ V &= \begin{bmatrix} 0.1819 & 0.6650 & 0.0024 & -0.7244 \\ 0.9418 & -0.2159 & 0.2547 & 0.0391 \\ 0.2423 & -0.0864 & -0.9661 & -0.0216 \\ 0.1456 & 0.7097 & -0.0423 & 0.6880 \end{bmatrix} \end{aligned} \tag{4.1}$$

As can be seen from the matrix V, the first eigenvector is a combination of all four parameters yet it is dominated by the second entry, corresponding to the parameter 'rf5'. Because this is the eigenvector belonging to the largest singular value, this is the vector that indicates the parameter direction which has the most influence on the model. The second most influential vector is mainly a combination of the first and fourth element, which correspond to the parameters 'rf2' and 'rf8'. From the matrix V follows that, though the first column vector is dominated by the parameter 'rf5', no single parameter can be perfectly identified. Instead each column vector is still a combination of parameters.

Using the Jacobian, information regarding parameter relations can be determined through calculation of the correlation matrix according to equations (2.25) and (2.26). The resulting correlations between parameters can be seen in figure 4.17. From this figure, a clear separation between two parameter groups can be seen. There are the parameters 'rf2' and 'rf8' which are highly negatively correlated and exhibit a lower correlation with the parameters 'rf5' and 'rf7'. For the latter two parameters the same holds. These are also highly negatively correlated and in turn show a substantially lower correlation with the parameters 'rf2' and 'rf8'.

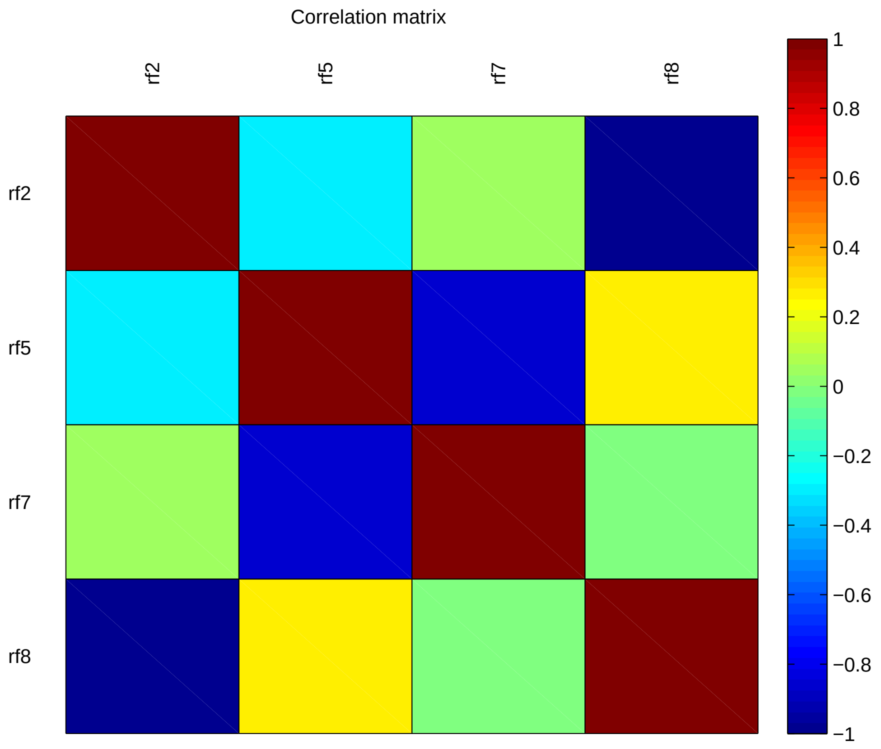


Figure 4.17: Correlation matrix of the four runoff parameters.

The results from the above parameter analysis is not surprising and seems in agreement with intuitive reasoning about the model. Since the surface areas listed in table 4.8 for 'rf2' and 'rf8' are of the same order, it follows that any increase in the runoff parameter 'rf2' could be negated by about the same decrease in the parameter 'rf8'. The fact that these parameters are less correlated with the other two, 'rf5' and 'rf7', also follows from the amounts of surface area. The amount belonging to 'rf5' and 'rf7' is about a factor 10 larger than the area belonging to 'rf2' and 'rf8'.

Surface area plus runoff parameters

Expanding the parameter vector to include the surface area belonging to the different runoff parameters, the same analysis as above is applied. The parameters that are dealt with in this case are the runoff parameters listed above and the surface area parameters 'surface2', 'surface5', 'surface7' and 'surface8'. Each of the surface area parameters is a collection of the nodes listed in figure 4.8.

In order to perform the analysis, the starting point for the parameter vector is the optimal model parameter vector. This optimal vector is not the result from a calibration but rather the input parameter vector for the model with which the measurements are created. Again the Jacobian is created by perturbing every single parameter in a separate model run and calculating the residuals. The matrix V resulting from the singular value decomposition of the Jacobian will shed some light on the identifiability of the parameters or groups of parameters. This matrix is visualised in figure 4.18. From this figure quickly follows that the parameters can at least be identied within two separate groups, namely the group of runoff parameters and the group of surface parameters. Another observation is that there are two parameters that are quite well identifiable in this case. These are the runoff parameters 'rf2' and 'rf5'. Other than that, the pair of runoff parameters 'rf7' and 'rf8' can only be identified as a combination.

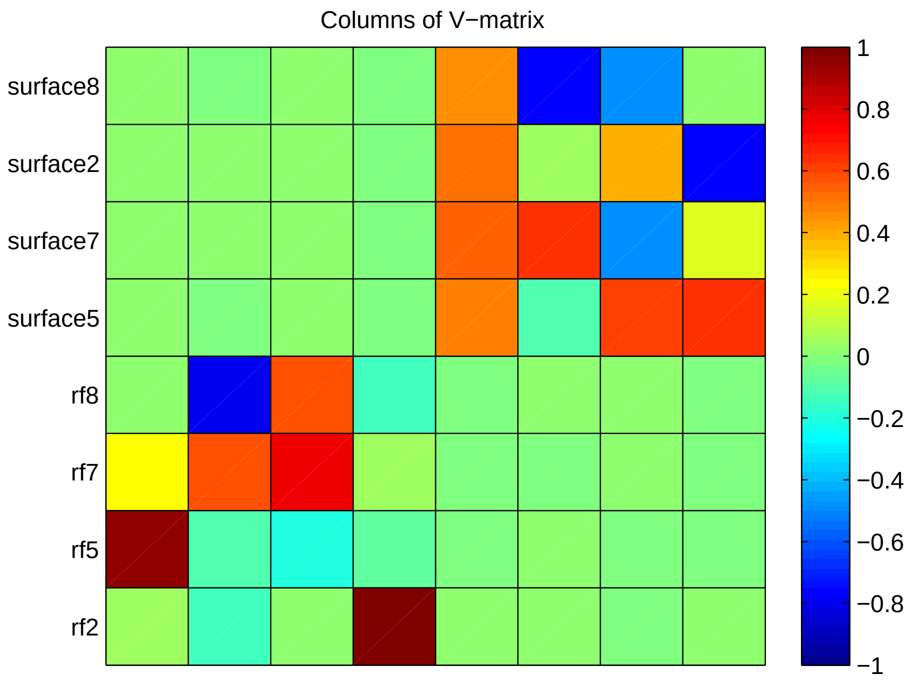


Figure 4.18: Columns of the matrix V. This gives an indication of the identifiability of parameters.

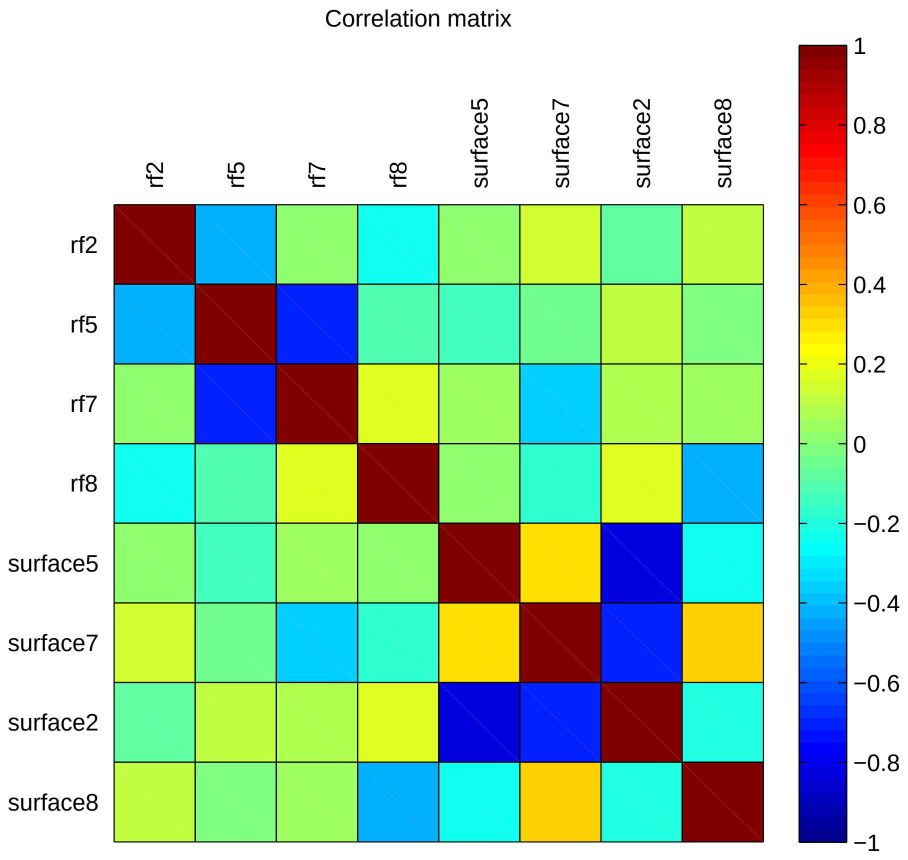


Figure 4.19: Correlation of the 8 parameters.

The singular values that belong to the columns of the V-matrix are

$$[55.8709 \quad 7.8821 \quad 5.9117 \quad 3.3476 \quad 0.1656 \quad 0.0191 \quad 0.0091 \quad 0.0036]$$

From the singular values, it is very clear that the runoff parameters have a much larger impact on the model than the surface area parameters. The fifth singular value is already a factor 20 smaller than the fourth. The corresponding column vector for this singular value indicates identifiability only for a combination of all surface area parameters. For the lower singular values, not a lot of useful information about the identification of the (combinations of) parameters can be given. These singular values indicate that the model is rather insensitive to changes in the particular parameters.

Using the Jacobian, the correlation matrix can again be calculated. This matrix is visualised in figure 4.19. It can be seen that some things are the same compared to figure 4.17. The correlation between the parameters 'rf5' and 'rf7' is significant, as it was in figure 4.17. A difference is that the correlation of 'rf2' and 'rf8' that was present in figure 4.17 is not present in the current case. Instead, 'rf2' shows a significant negative correlation with 'rf5' and in turn 'rf8' shows some negative correlation with the 'surface8'.

Explanations for the correlation are again somewhat intuitive. The correlation between 'rf8' and 'surface8' can be explained because an increase in runoff parameter together with a decrease in surface area can yield the same amount of water entering the sewer system. Though the total amount of water entering the sewer system will eventually be lower due to less surface area being present.

Correlation between different runoff parameters are again explained by the fact that the total amount of water entering the system will remain more or less constant when perturbing these parameters according to their (negative) correlation. The same holds for parameters for the surface areas. Enlarging one type of surface area and diminishing another can yield the same total amount of surface area and thus the same amount of water that is caught and enters the system. What is somewhat harder to explain is the lack of a significant negative correlation in figure 4.19 compared to figure 4.17.

This section gave a brief oversight of a method to determine if parameters can be identified for model calibration and in what way parameters are correlated with each other. It can be used to group certain parameters that cannot be identified separately, thus reducing the total number of parameters that have to be calibrated. This method has also been applied in e.g. [Post, 2012] in order to group parameters concerning sewer overflow at internal weir locations where these could not be identified separately. The method of calculating the covariance matrix by using the Jacobian can be used by certain calibration methods to provide the user with confidence intervals for the calibrated parameter values. Such applications can be found in e.g. [Stegeman, 2012], [Clemens, 2001] and [Stolze, 2008].

Other than the parameters described here, there are more that can be included in the calibration and analysis. Some examples of these parameters are friction and orifice discharge coefficients [Stegeman, 2012] which can be used to simulate blockages present in the system and internal weir coefficients [Post, 2012] to influence the sewer overflow.

Chapter 5

Assimilation

5.1 Model uncertainty

Model uncertainties are a very important factor when applying the ensemble Kalman filter. Several methods of modelling these uncertainties have been used. Basically, uncertainties are modelled by imposing noise on the state of the system and propagating the state including this noise in time. The noise that is used is generated by an AR(1) process.

5.1.1 Model state

The model state of a system are the results from a simulation. For a sewer system model this could be the waterlevels and also the discharge values in the system. For the sewer system model in Sobek, the model state consists of merely the waterlevels, meaning that these values can or will be adjusted by OpenDA whenever observations are available. This creates some discrepancy between the discharges and the waterlevels in the system, but this is quickly fixed when Sobek performs the next timesteps in the simulation where the discharges are corrected using the new waterlevels.

An extension of the state can be made by making certain parameters time dependent. These parameters can then be added to the state vector in OpenDA and then go through the filtering process. This is particularly useful when dealing with parameters that are assumed constant in time but might very well vary depending on, for example, the amount of rainfall.

5.1.2 AR(1) process

The first-order autoregressive process or AR(1) process is used to provide a sequence of dependent random variables $\gamma(t_0), \gamma(t_1), \gamma(t_2), \ldots$ with mean 0 and limiting standard deviation σ_γ, where the subscript indicates that this is the standard deviation belonging to the random variable $\gamma(t_i)$ as $i \to \infty$.

$$\gamma(t_i) = \alpha\gamma(t_{i-1}) + \mu(t_{i-1}) \quad \text{for} \quad i = 1, 2, \ldots \tag{5.1}$$

In this equation, the parameter α depends on the noise model timestep and a chosen parameter τ, which is the time correlation scale, in the following way: $\alpha = \exp\left(-|t_{i+1} - t_i|/\tau\right)$ if $i \geq 1$. From equation 5.1 also follows that if $\alpha = 0$, this process will generate white noise only because nothing of the previous timestep is carried over to the next timestep and then the $\gamma(t_0), \gamma(t_1), \gamma(t_2), \ldots$ are independent of each other. $\mu(t_i)$ is normally distributed with mean 0 and standard deviation equal to $\sigma_\mu := \sigma_\gamma\sqrt{1-\alpha^2}$. The first realisation, $\gamma(t_0)$, comes from a normal distribution with standard deviation σ_γ. It follows that

$$\begin{aligned}\sigma^2_{\gamma(t_1)} = \text{var}(\gamma(t_1)) &= \alpha^2\text{var}(\gamma(t_0)) + \text{var}(\mu(t_0)) \\ &= \alpha^2\sigma^2_\gamma + \sigma^2_\gamma(1-\alpha^2) = \sigma^2_\gamma\end{aligned}$$

Assuming that $\sigma_{\gamma(t_i)} = \sigma_\gamma$ holds for i, it is found that

$$\begin{aligned}\sigma^2_{\gamma(t_{i+1})} &= \alpha^2\text{var}(\gamma(t_i)) + \text{var}(\mu(t_i)) \\ &= \alpha^2\sigma^2_\gamma + \sigma^2_\gamma(1-\alpha^2) = \sigma^2_\gamma\end{aligned}$$

Thus by induction it is seen that $\sigma_{\gamma(t_i)} = \sigma_\gamma$ for all i. Using an AR(1) process, noise realisations are made and these are then used to augment the model state to represent the uncertainty. The way in which this is done is described in the next section below.

5.1.3 Noise modelling methods

Modelling the uncertainty in the state is an important issue in the ensemble Kalman filter. In general this is done by imposing noise on the state vector with a particular distribution function for each ensemble member. As time progresses, the ensemble of model states will still be a good representation of the uncertainty. In this specific case, this is done by using an AR(1) process to generate noise realisations and then add these to the state. Referring back to the model equation 2.1, the uncertainty in the state is represented by the term $G_k w_k$. The vector w will be a vector with a number of realisations of an AR(1) process so w_k is the vector containing realisations of the random variable $\gamma(t_k)$. The size of this vector depends on the number of realisations that are required, which in turn depends on the specific noise modelling method applied but will not exceed the size of the state vector. G is a matrix which is created according to the noise modelling method and it determines what noise from the vector w is applied to specific nodes in the system. The matrix G will be constant over time unless stated otherwise, so $G_k = G$.

The model uncertainty in a certain point is in some way also dependent on the surrounding points. Thus, to ensure that the noise applied to the state vector is a good representative for the model uncertainty, several different ways of modelling uncertainty have been investigated. Each of the methods discussed represents the way that noise is applied to the state vector for a single ensemble member. In the ensemble Kalman filter, this method is then applied for each of the ensemble members. The first way of doing this is by generating a single noise value and putting this same value on all the nodes in the system. This means that when noise is applied, the waterlevel of all the nodes is increased (or decreased by the same amount). This method corresponds to defining a single noise model in the configuration file and putting all elements of the state vector in one block. When applying this method, the matrix G will actually be a vector with length n, which is the length of the state vector. The vector w_k will consist of a single realisation of the AR(1) process:

$$G = [1 \quad 1 \quad \dots \quad 1]^T, \qquad w_k = \gamma_1(t_k)$$

A different method of modelling the uncertainty is to create as many realisations of the AR(1) process as there are state vector elements. This allows for a different realisation to be put on each element of the state vector. The matrix G corresponding to this method is the n x n identity matrix. The vector w_k is a vector with length n and so contains n different realisations of the AR(1) process.

$$G = I_{nxn}, \qquad w_k = [\gamma_1(t_k) \quad \gamma_2(t_k) \quad \dots \quad \gamma_n(t_k)]^T$$

The first method described above has the advantage that only one generation of the noise parameter is necessary and this is put on the entire state as a whole. The disadvantage is that this way of imposing noise will have a very low information value. The other option to impose a new realisation of the noise parameter on every node, would theoretically give a lot of information. The disadvantage of this method, however, is that the noise will be "flushed out" rather quickly. The mean value of the noise imposed on the system is (would be) 0, which is the sum of the increase in waterlevels due to this noise.

The idea of this noise being quickly flushed out can be explained by a simple example. Given a very small system consisting of three consecutive nodes. Then three realisations of noise, with a mean of 0, are added to the waterlevel in each of the nodes. When the model continues the simulation, it will not take long for the water to flow from the increased waterlevel(s) to the decreased waterlevel(s), resulting in the same situation as just before the noise was applied.

A third option, which will be referred to as the boxing method, is to divide the system in smaller areas and, for each area, generate one noise realisations which will be used for all the nodes in this particular area. With this method, points that lie in the same area will be increased (or decreased) by the same amount, so any water added will not flush out directly. The mean of the noise put over the different areas will be 0. The number of nodes in each area is not constant, some areas may contain as little as 0 points while other areas may contain more than 40 points. This means that the mean of the noise put over all the points is not necessarily 0. A visual representation of how noise is added to the state of the system can be seen in figure 5.1 for the way that different noise is generated for each point and in figure 5.2, where the system is divided in a grid of 10 x 10 and the nodes in the same block will be subject to the same noise value. For this specific 10 x 10 example, the number of areas will of course be 100. This means that, at most, 100 different realisations of the AR(1) process will have to be generated. The vector w_k will then be a vector of length 100. The matrix G will be an n x 100 matrix containing 1's and 0's. Each column of this matrix corresponds to an area and each element in this column vector will be either 1, if the corresponding state vector node is in the area, or 0 if it is not.

$$G = \begin{bmatrix} 1 & 0 & 0 & \cdots \\ 1 & 0 & 0 & \cdots \\ 1 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ 0 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \quad w_k = \begin{bmatrix} \gamma_1(t_k) \\ \gamma_2(t_k) \\ \vdots \\ \gamma_{100}(t_k) \end{bmatrix} \tag{5.2}$$

When applying the boxing method, noise values of the different blocks might still differ by a lot. To this end, some interpolation can be applied between the blocks. The center of the block will have the noise value originally imposed and on other points the noise value is determined by interpolation using the centers of the neighbouring blocks. The effect that this has on the noise imposed on the nodes can be seen in figure 5.3. For a good comparison, the noise realisations are the same as the ones used in figure 5.2. With this interpolation also spatial correlation is introduced.

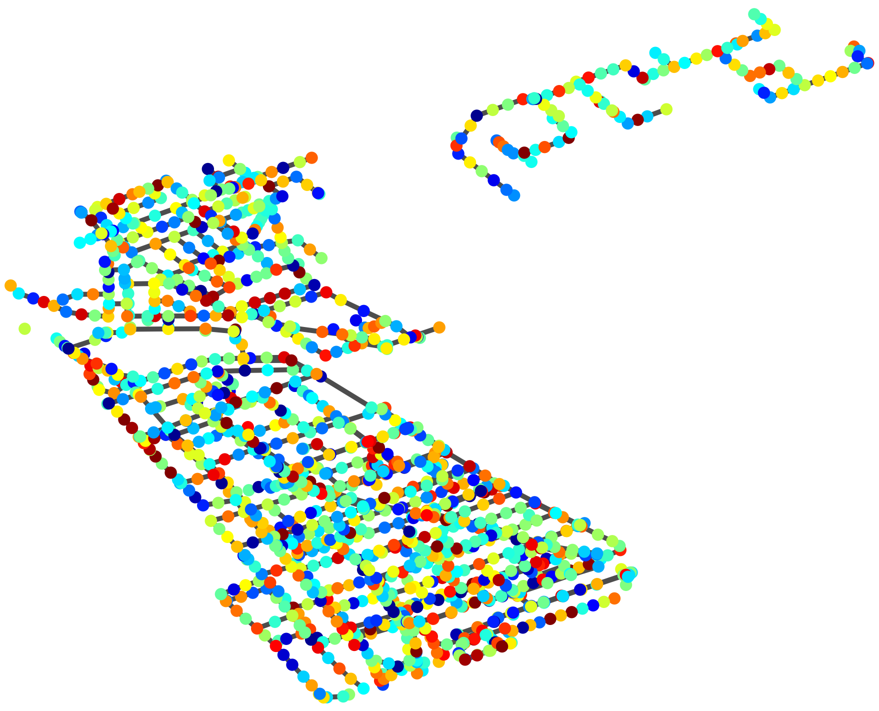


Figure 5.1: This figure represents the distribution of noise among the points in the system when each point is given a different value.

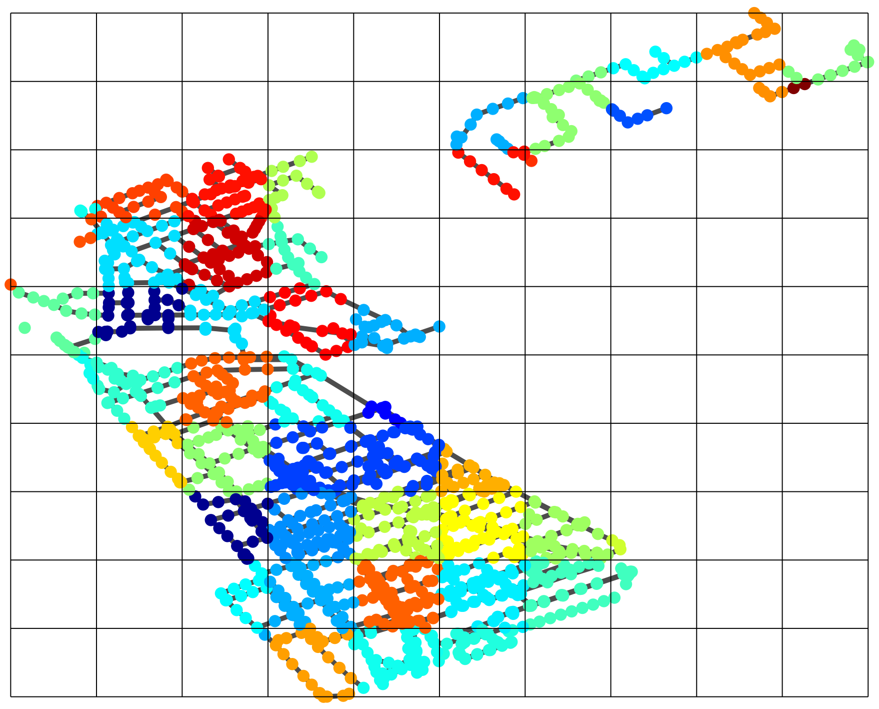


Figure 5.2: Each block in the figure has its own independent noise realisation, which means that the nodes in the same block will receive the same noise value.

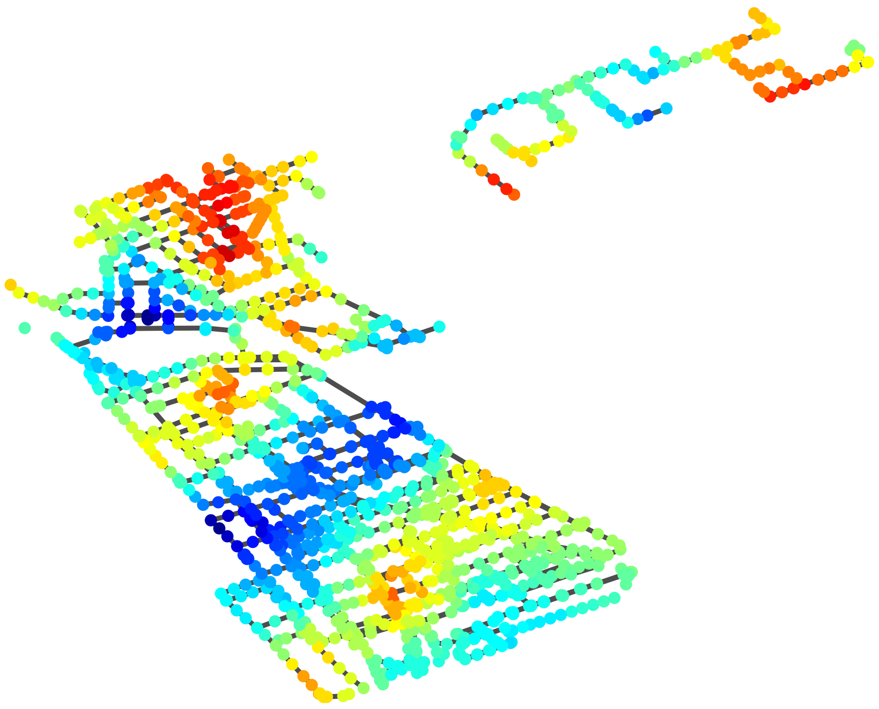


Figure 5.3: The same noise realisations are used as in figure 5.2 but in this case a spatial correlation is created by interpolating noise values that lie between block centres

The implementation of the different noise methods, using the configuration types that are discussed in appendix A, is straightforward, except for the interpolated boxing method. When a method is applied in which different newly generated noise values are put on one or several nodes, then these nodes can be put in different configuration blocks. For the interpolated boxing method however, there is some manipulation after the generation of the noise and as such, it is easier to generate noise from outside the OpenDA environment, store it in a file and import the noise from this file into OpenDA.

5.2 Precipitation

An important factor in simulating urban drainage systems is the precipitation that is used. The same precipitation is used in most, if not all, simulations. This is a two hour storm event used to simulate heavy rainfall in a short amount of time in which most of the rainfall takes place in the first hour. The storm event is visualised in figure 5.4. The average intensity over the first hour is 10.05 mm/hr and the average intensity over the complete event (two hours) is 5.25 mm/hr. From a report of the KNMI, [Buishand and Wijngaard, 2007], follows that the 60 minute event occurs twice every year. So the event is not incredibly rare yet still presents a decent storm event to simulate heavy weather conditions with and thus a good event to run simulations with.

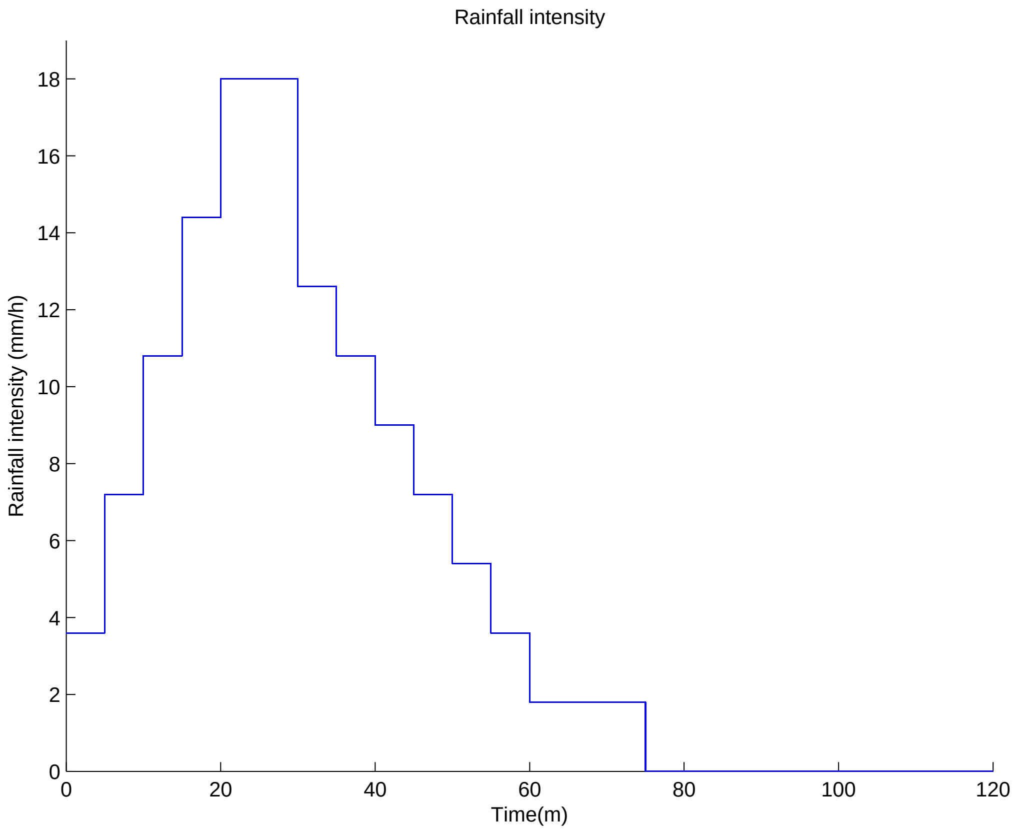


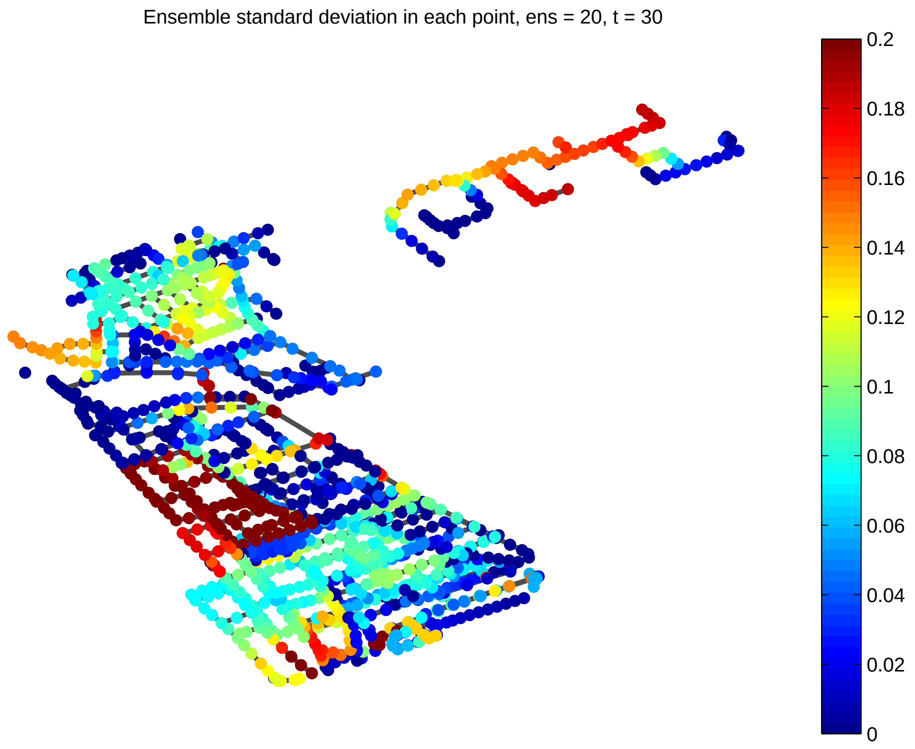
Figure 5.4: The precipitation used in most (all) simulations. The amount of rainfall per timestep can be seen, where the timestep is five minutes. The total time of the event is two hours.

5.3 Number of ensemble members

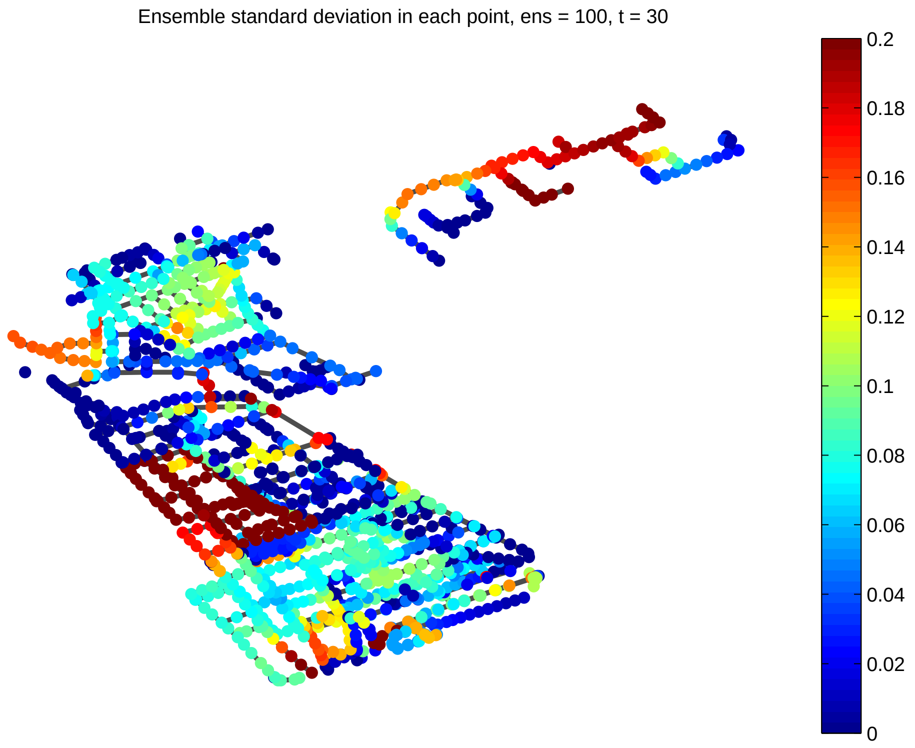
The number of ensemble members is a key factor in determing the error statistics for the assimilation. More ensemble members would yield a better and more accurate representation of these statistics. However, since the model must be run for each ensemble member, this can be very time consuming for a high number of ensemble members. A choice is made to run the assimilations with an ensemble size of 20. To examine if this is a good choice, a comparison is made with a simulation of 100 ensemble members. Specifically the propagation of the ensemble spread, which can be characterized by the standard deviation, is of interest. If the ensemble spread for an ensemble of 20 members is close to the ensemble spread for an ensemble of 100 members, then the ensemble of 20 members is considered well enough for the assimilation. The ensemble spread in the system for an ensemble size of 100 and an ensemble size of 20 are compared at different times for all locations.

In order to accomplish this, two stochastic simulations are set up. These simulations are performed without any measurements and forward the model in time in order to get a better view on the model uncertainty. For the first simulation an ensemble size of 20 is used and for the second simulation an ensemble size of 100. Other aspects of the simulations are kept the same. The noise that is added to the state vector will be done using the boxing method without the interpolation. Generating noise realisations is done using an AR(1) process with standard deviation $\sigma_\gamma = 0.1$ and time correlation scale parameter $\tau = 60$ minutes and noise model period is $t_{i+1} - t_i = 10$ minutes.

In section 3.3.3 it is argued that Sobek restarts will be performed at time intervals of 10 minutes to avoid a flaw in the Sobek restarts. This restart interval is the reason that the noise model period is set to 10 minutes. Every time the simulation is restarted, it is accompanied by a new noise model realisation. Figures 5.5 and 5.6 show the standard deviation for each point at simulation times 30 and 60 minutes respectively. Figure 5.7 shows the difference in standard deviation between the case where 100 ensemble members are used and where 20 ensemble members are used.

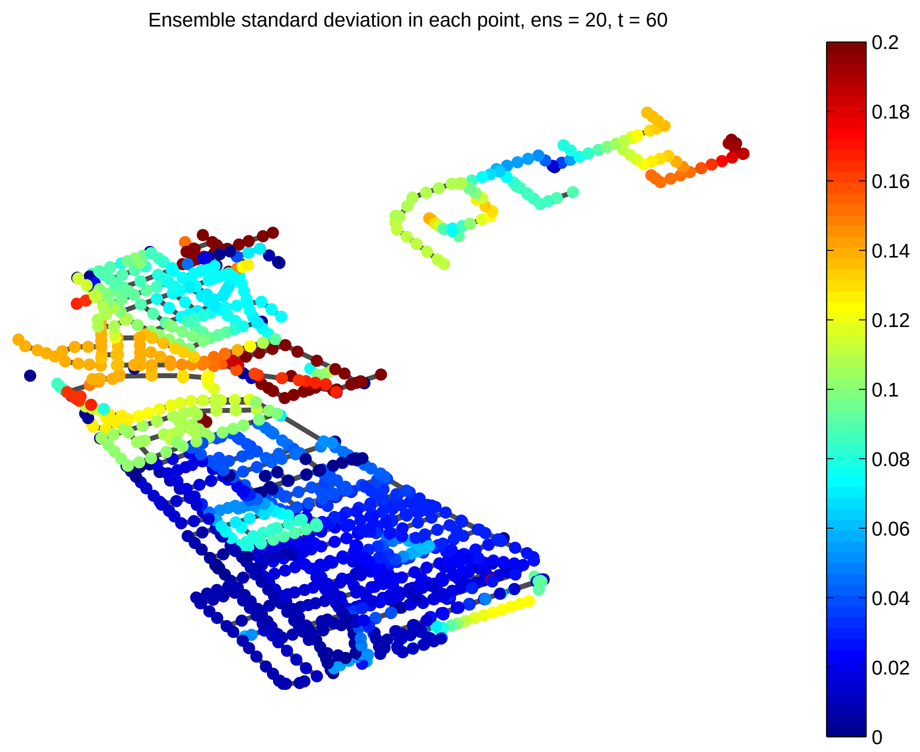


(a) Ensemble size = 20

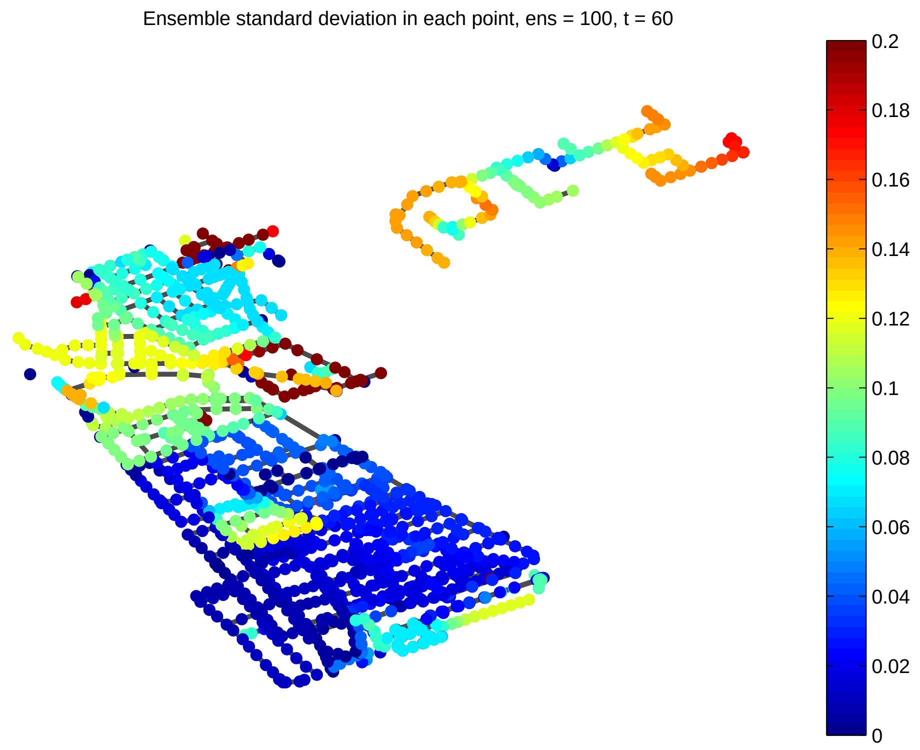


(b) Ensemble size = 100

Figure 5.5: Standard deviation for two different ensemble sizes at time = 30 mintes

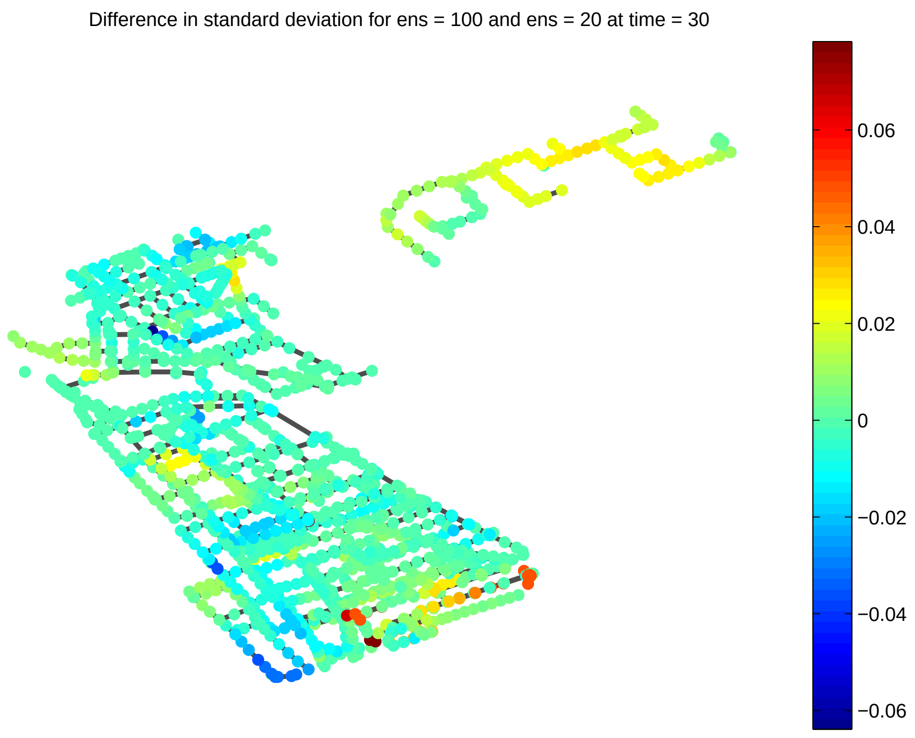


(a) Ensemble size = 20

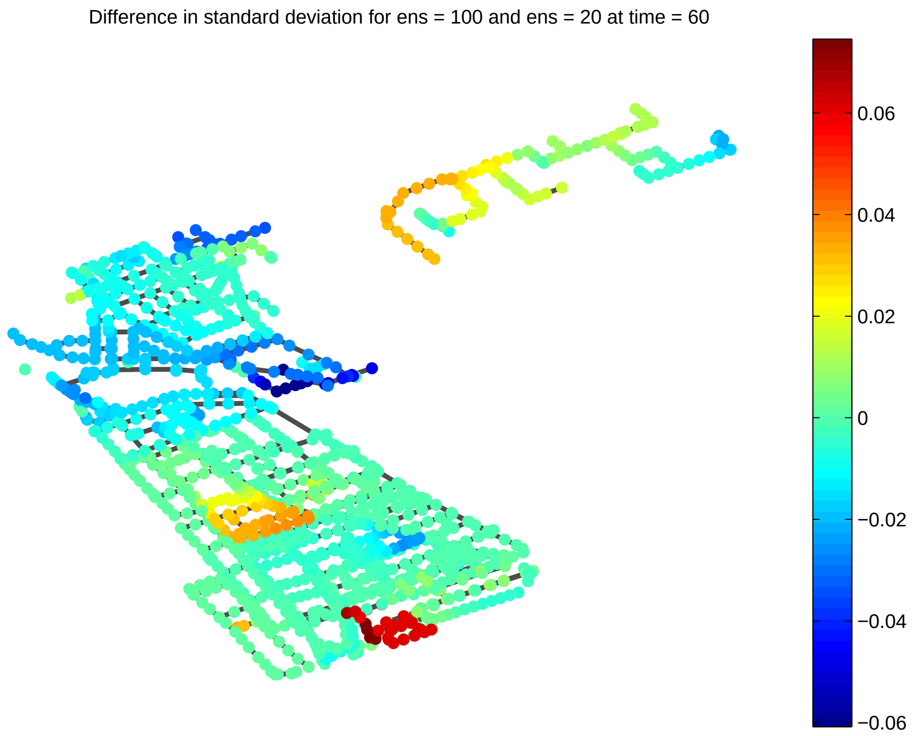


(b) Ensemble size = 100

Figure 5.6: Standard deviation for two different ensemble sizes at time = 60 mintes



(a) Difference at time = 30



(b) Difference at time = 60

Figure 5.7: The difference in standard deviation between the stochastic simulation with 100 ensemble members and the simulation with 20 ensemble members

The differences in the standard deviation between both ensemble sizes is small. At time $t = 30$ minutes, the mean of the difference is 0.0013 with a standard deviation of 0.0123. At time $t = 60$ minutes, the mean of the difference is -0.0009 and the standard deviation is 0.0159. These numbers indicate that the overall difference in the two ensemble sizes is small in the entire system and in most of the system this difference is very close to 0. A few areas catch the eye where the difference seems to be relatively large; the dark blue area just above the middle, where a large negative difference appears indicating a larger standard deviation for an ensemble size of 20 than for 100. The orange area just below with a medium positive difference and the red area below that with a large difference. The difference in the blue area is not as important as the standard deviation itself. For both 20 and 100 ensemble members, this standard deviation (figure 5.6) is rather large (> 0.23) which means that, as far as representing error statistics, both ensemble sizes will yield a rather large uncertainty.

For the orange area, the same story goes as for the blue area. Again from figure 5.6 it can be seen that this area has a larger standard deviation than the surrounding points (around 0.1 for the ensemble size of 100 and around 0.08 for the ensemble size of 20). It can be seen that this area is more or less isolated from the rest of the system and is only connected by two reaches.

The red area is even more isolated than the orange area, in that it is only connected with the rest of the system by a few internal weirs and no regular pipes. Therefore the difference in standard deviation for this small area is disregarded.

Based on the ensemble standard deviations for 20 and 100 ensemble members, the conclusion is made that using an ensemble size of 20 will provide a well enough representation of the error statistics. Which is why most simulations with assimilation will be done using an ensemble size of 20.

5.4 Observation locations

For the assimilation of the system, a set of assimilation points is used (also referred to as measurements or observations). The objective for these points is that they provide as much information about the entire system as possible. The simplest way of achieving this is to take measurements in every single point of the system. However, this is not realistic for a number of reasons; firstly, it might not be possible to take measurements in every single point in the system for structural reasons e.g. manholes might not be reachable. Secondly, the cost of realizing a full monitoring network is simply too great.

The next best thing is to construct a small set of assimilation points that still provide a lot of information about the entire system. This can be done by considering correlation between points. A strong correlation between one point and its neighbouring points indicates that any change in this particular point will likely mean a change in the correlated neighbouring points as well. Another consequence of a high correlation is that when a measurement is available that leads to an update of the state in this point, the neighbouring points that are highly correlated will be adjusted as well.

A set of observation locations had been provided by [Post, 2012], however, this set was not very well suited for the use in the ensemble Kalman filter and so it was improved by developing a new set of locations specifically for the ensemble Kalman filter. The old set had been developed specifically for calibration purposes. For calibration, the correlation of observation locations with other points in the system is of less importance than when applying data-assimilation with the ensemble Kalman filter. This set of observation locations is used in the calibration in section 4. A short discussion on this set of observation locations can be found in appendix B.

The development of a new set is discussed below.

Determining the correlations between points in the system is done by performing a stochastic ensemble simulation using an ensemble size of 100. This simulation needs to be done only once, so computation time for this stochastic simulation is of less importance. Also, the assumption is that the ensemble size of 100 will give correlation estimates that are better (be it slightly better) than for an ensemble size of 20. In this stochastic simulation, noise is added to the state at 10 minute time intervals, according to the boxing method as illustrated in figure 5.2, to mimic uncertainty. The informative value in every point will be based on the correlation of this point with other points in the system. The correlation of points with eachother will diminish as the distance between these points gets larger. Correlations that do appear over longer distances are called spurious correlations and will not be considered. This is done by considering a circle around each point with a fixed radius and only considering the correlation with points that are within this circle. This radius is set to 500 metres. This is a somewhat arbitrary number, though it has a base in the fact that the velocity of water in the system will be limited. So a change in a point will not influence points over a distance that simply can't be reached. Velocities greater than 1 m/s are already considered rare and with this velocity and in this case, water will travel at most 600 metres in 10 minutes.

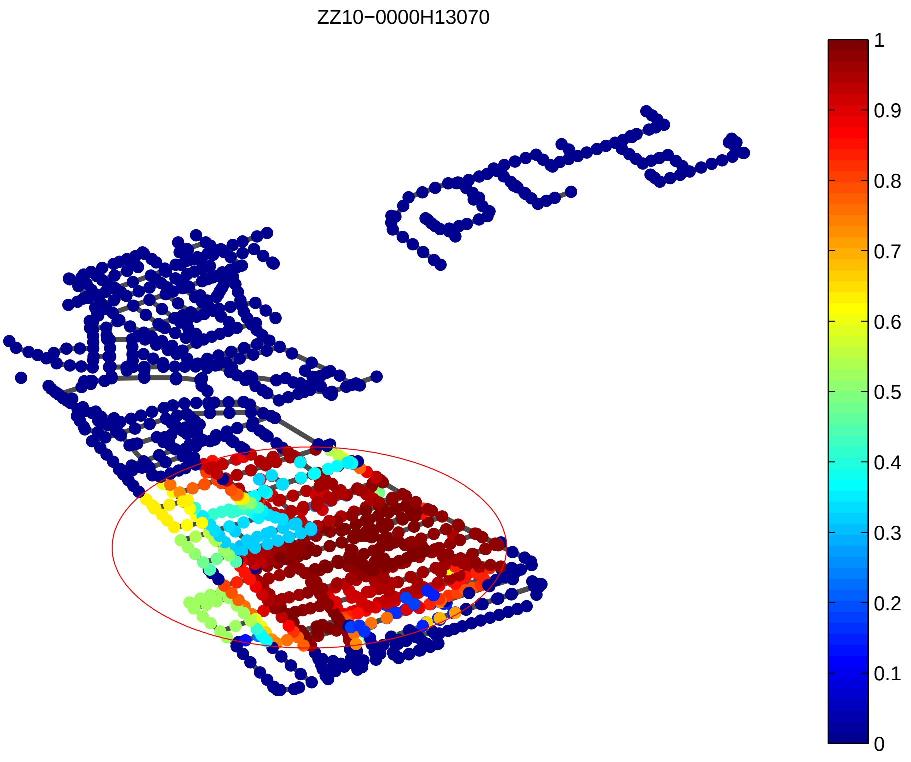


Figure 5.8: This figure shows the correlation of all the nodes within a 500 metre radius with the center node. The correlation of nodes outside the radius is set to 0.

Figure 5.8 gives an example of a node and its correlation with other nodes within a radius of 500 m. With only correlations considered within this circle, there are several different ways of defining an informative value for the points in the system:

- Number of points that are considered to have a correlation, i.e. number of points in the circle.

- Average correlation of the particular point with all other points within the radius. This can be applied to either absolute values of correlations or the real correlations.
- A combination of the above. An obvious criterion for informative value would be a reasonable number of correlated points within the 500 metre radius while at the same time having a rather high average correlation.

The last option is the obvious choice to determine the informative value in each node. Specifically, a point is informative if it satifies at least the following criteria:

1. The average correlation with other points within the radius of 500 metres is at least 0.5.
2. The number of correlated points times the square of the average correlation with other points is at least 120.

The first criterion is a rather obvious one. A minimal requirement is imposed on the mean correlation with surrounding points for a point to provide some worthwhile information. The number 0.5 in the first condition is chosen arbitrarily. The second condition defines a relationship between the number of correlated points and the mean correlation of these same points. Taking the square of the mean correlation will lead to a relationship between the mean correlation and number of points where when the mean correlation doubles, the required number of correlated points is divided by four. The number 120 in the condition is again chosen somewhat arbitrarily. This number leads to a requirement of 480 nodes when the mean correlation is 0.5.

The listed criteria provide a minimal condition for points in the system to be informative. Both criteria can be tightened by increasing the number for the required average correlation in the first criterion or the number of correlated points times the square of the average correlation with other points in the second criterion.

The points that satisfy the minimal conditions can be seen in figure 5.9, as well as points which satisfy stronger versions of the second criterion. All the points which satisfy the minimal criteria can be considered as possible observation locations. The information value in the figure is scaled from 0 to 1. Where 0 indicates that these points do not even satisfy the minimal criteria as listed above. Higher values for the information indicate stricter versions of the second criterion. This means that points with an information value close to one are highly suited to be used as observation locations for the ensemble Kalman filter.

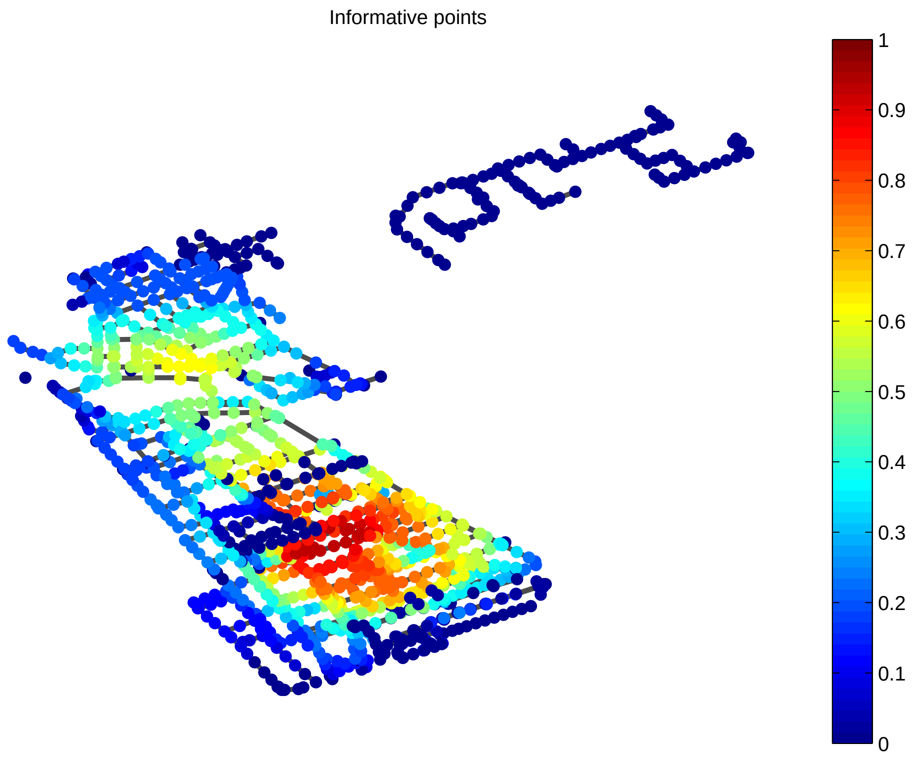


Figure 5.9: This figure shows the information value provided by each node in the system.

From these points, a set of measurement points is chosen. The final condition for measurement points is that the chosen points should not be too close to each other, since choosing neighbouring points will barely increase the information that the set gives about the system. The set of observation locations chosen can be seen in figure 5.10. A comparison of the new set of observation locations is made and can be seen in appendix B.
For use in the assimilation, an uncertainty will be defined on these observations. This uncertainty depends on for example the accuracy of the measuring equipment used or the event that measured values are off due to sediment blocking the equipment or comparable causes.

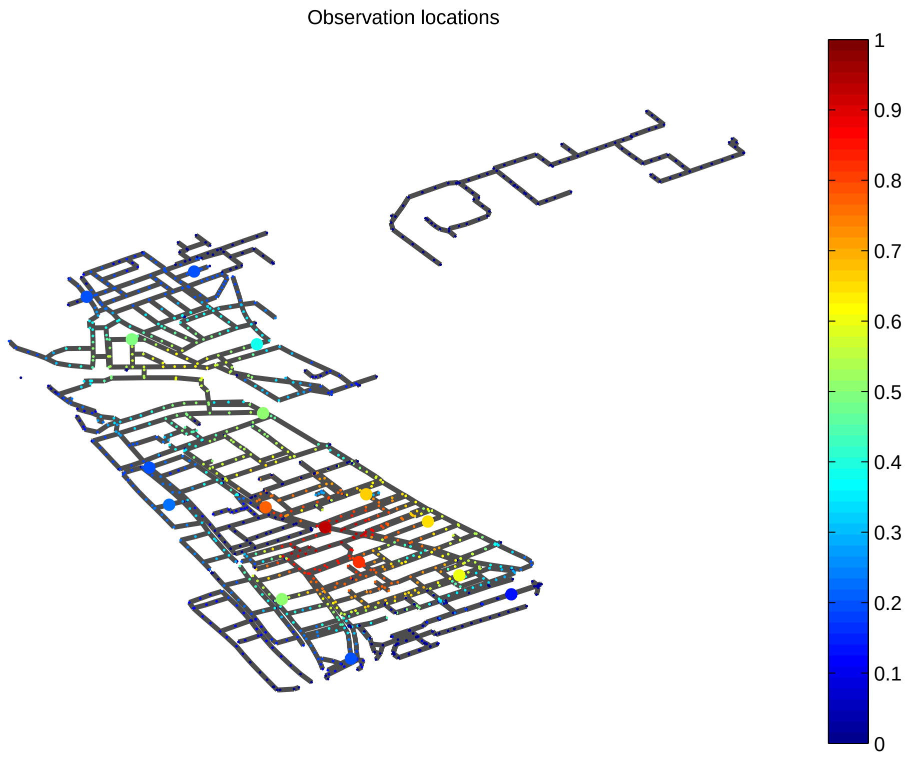


Figure 5.10: The 16 coloured points are the ones chosen to be the set of measurement or assimilation points.

5.5 Assimilation results

For the assimilation of observations, the ensemble Kalman filter is used. Initial attempts at assimilating observations gave bad results. This was due to a bug in the part of OpenDA that handled the observations. More information about this can be found in appendix C.

With the set of optimal observation locations some assimilation experiments can be done. This will be done by performing a twin-experiment. When setting up a twin-experiment, observations are created by running a stochastic simulation with a certain set of initial conditions. Next, a perturbation is applied to the set of initial conditions and the new set of initial conditions is then used in the assimilation of the observations created earlier. The perturbation that is applied can be a modification of initial waterlevels or a change in certain parameters that will be taken into account in the assimilation.

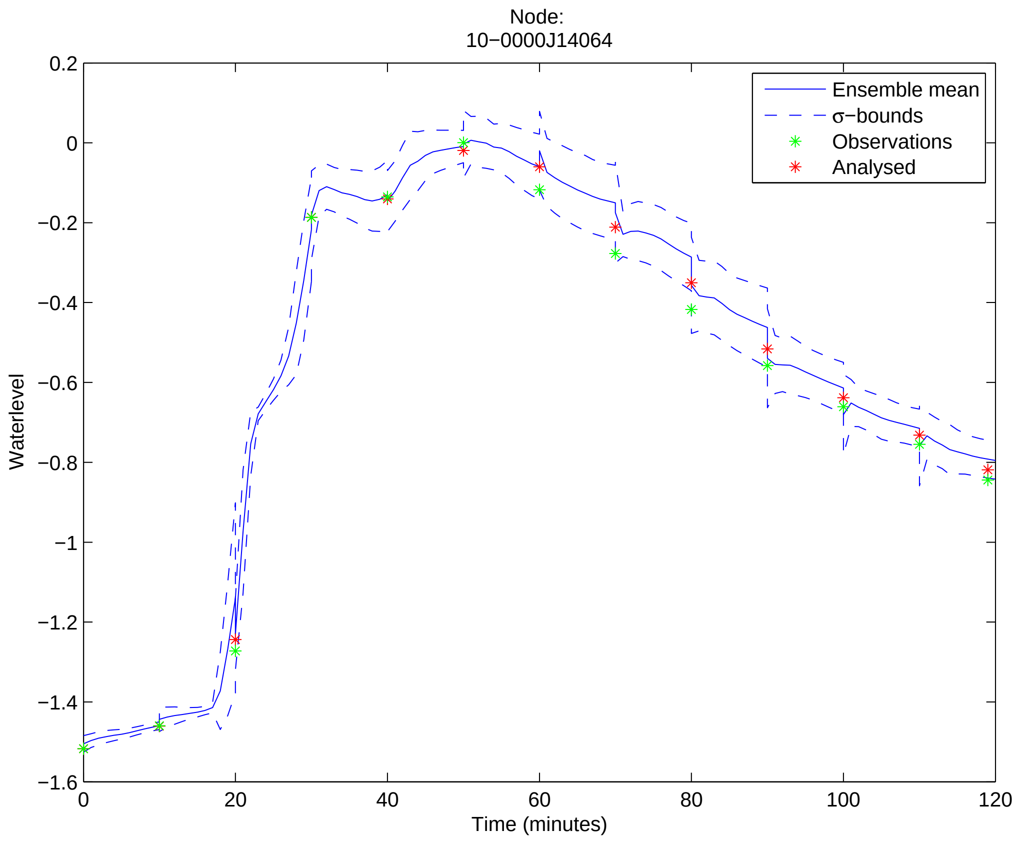
5.5.1 Non-perturbed initial assimilation

The first experiment is a test of the blackbox model and observation locations. In this case, no perturbation will be applied to the initial conditions or any of the parameters present in the model. Details of the setup of the simulation can be found in table 5.1. The observational data that is used in the assimilation comes from a stochastic run performed with the same model uncertainty and noise method as used in the assimilation. This stochastic run is done with an ensemble of model states and one ensemble member is picked to serve as observational data. From the results of this ensemble member, time series are created for the waterlevels in the observation locations which can then be used in the assimilation.

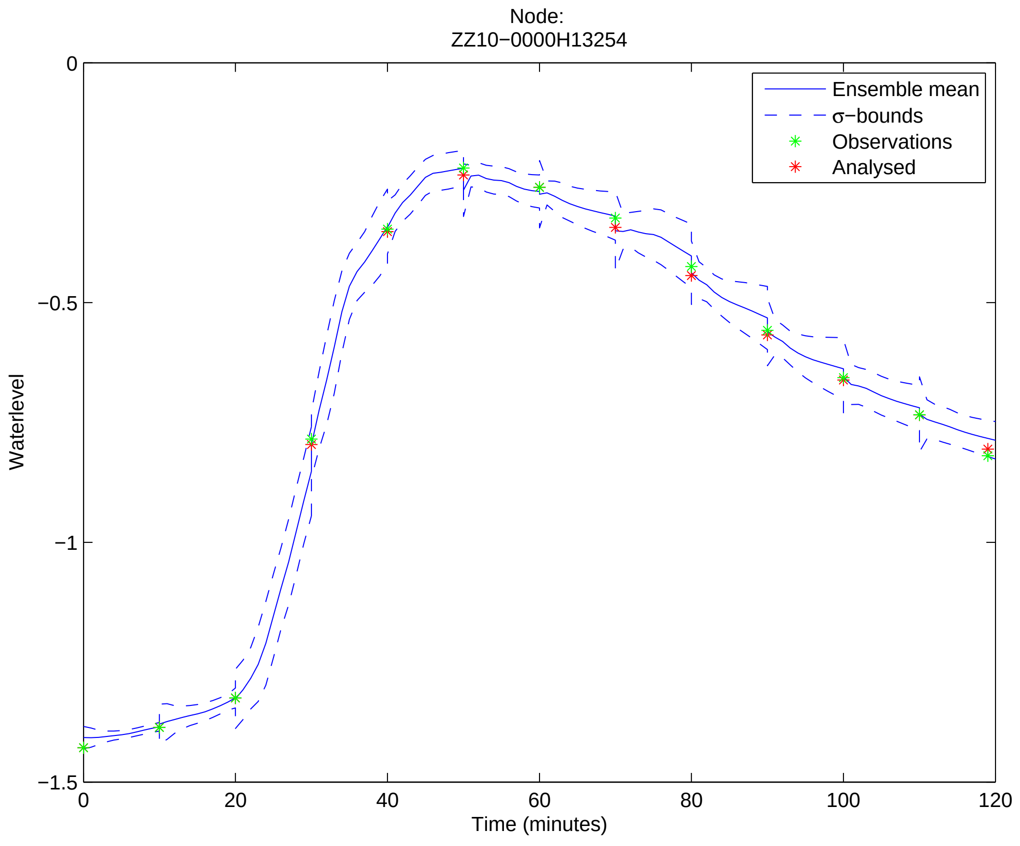
Ensemble members	20
Model uncertainty	0.1 m
Measurement uncertainty	0.1 m
Used noise method	boxing (eq. 5.2)

Table 5.1: Setup of the assimilation for the first twin experiment

Resulting waterlevels in two of the observation points can be seen in figure 5.11. The ensemble mean is used as the best guess for the model forecast at assimilation times, which is the solid blue line in the figure. The imposed model uncertainty is simulated by creating an ensemble of model states with this predescribed standard deviation. The dashed blue lines in the figure indicate the ensemble spread (ensemble standard deviation) for the model state as it is propagated in time. The observations that are used are shown in green at assimilation times. The resulting analysed waterlevels are shown in red.



(a)



(b)

Figure 5.11: The observed waterlevel, ensemble mean and its one standard deviation (σ) bounds and the analysed waterlevel in two of the measurement locations.

The figure shows no drastic changes in these two nodes as observations are assimilated, which is what would be expected from an assimilation with no perturbations applied. The (small) updates that do occur are a result of the way that observational data is created. Creating this observational data from a single ensemble member from a stochastic run means that there is no guarantee that the observational data will match the ensemble mean. On the other hand, observational data will not be very far off from the ensemble mean thanks to the standard deviation of the noise that is put on the state.

For each point in the system, the average of the absolute values of the difference between ensemble mean and the analysed state over the entire assimilation timespan can be seen in figure 5.12. That is, let $\bar{x}_k^f$ be the ensemble mean at time k and x_k^a the analysed state at time k, the values are calculated as follows

$$\text{Average absolute difference} = \frac{1}{n} \sum_{k=t_1}^{t_n} |\bar{x}_k^f - x_k^a|$$

These values give a good indication for the behaviour of the filter. Since no perturbation is applied to initial conditions, the expectation is that no significant updates will be done when assimilating the observations. The figure in fact shows that the analysis updates in the entire system do not exhibit any extreme behaviour. The average analysis update for all point is below 0.055 with 85% of all the points even below 0.025. The reason that these numbers are not closer to 0 is due to the fact that the observational data was created with the stochastic simulation with a standard deviation of 0.1 on the state vector.

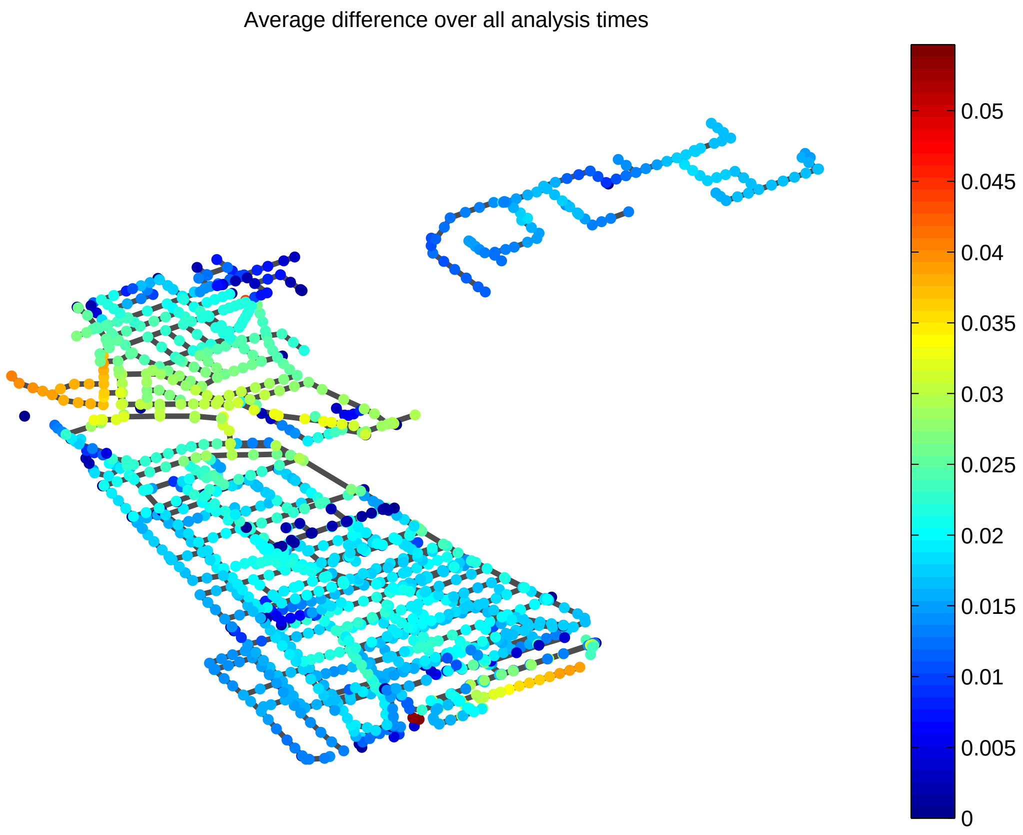


Figure 5.12: This figure gives an overview of the average absolute difference between the ensemble mean and the analysed state for each point.

Pluvial flooding

An aspect in modelling sewer areas with heavy rainfall events, is the occurance of pluvial flooding. This occurs when the sewer system is not able to cope with the amount of water that enters the system and can cause extensive (economical) damage, see for example [Veldhuis, 2011] and [Spekkers et al., 2011].

The effect of the rainfall event on the pluvial flooding is investigated. Figure 5.13 shows the areas which are, at some point, flooded. Since this case concerns a non-perturbed simulation, this flooding is only the result of the heavy rainfall event that is used. The visualised flooding is useful in comparing results with the assimilation of certain sediment levels in reaches by using this figure as a reference.

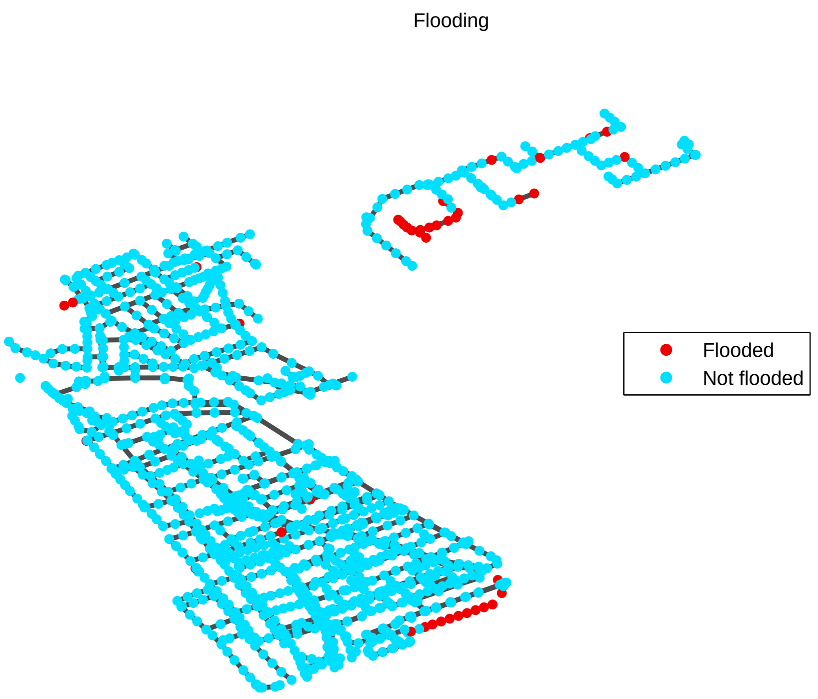


Figure 5.13: Flooding that occurs in the case of the non-perturbed twin experiment which was set up according to table 5.1.

5.5.2 Sediment levels

An experiment is done by including sediment levels in the state. These sediment levels can be used to simulate a blockage in the sewer system. By making these sediment levels time dependent and adding them to the state vector, the assimilation of measurements can lead to an indication where the sewer system is blocked.

Sediment levels in Sobek

In Sobek, reaches are defined individually. However, the shape type of these reaches can be a general one. A sewer system in Sobek can contain, for example, 10 reaches which are all round pipes with a diameter of 600 mm. The same shape type definition, that is 'Round 600 mm', will then be used for each of the 10 reaches. The sediment levels in a reach are defined in the shape type definition. This means that when sediment levels are used in simulations, these sediment levels are then applied to every reach that has this particular shape type. This obviously is not ideal when trying to simulate sediment levels in individual reaches. A solution for this is to create a separate shape type definition for each reach, so that sediment levels can be defined individually. Before this is done, however, a twin-experiment is performed to examine the behaviour of the assimilation of the sediment level in one shape type.

For this twin-experiment, perturbations are applied to the sediment level in the reach type 'Round 600 mm'. This means that a sediment layer is defined in every reach in the system that has this particular type. As mentioned several reaches can have the same shape type, see figure 5.14. In the sewer system there are 53 reaches that have this type. The setup of the twin-experiment in this case can be seen in table 5.2

Measurement creation	
Pipe diameter	0.6m
Initial sediment level in Round 600 mm	0.5 m
Initial sediment level in other reach types	0 m
Model noise method	Boxing method
Noise on the waterlevels	0.005 m
Noise on sediment levels	0.005 m
Assimilation	
Ensemble members	20
Initial sediment level overall	0 m
Model noise method	Boxing method
Noise (uncertainty) on state	0.05 m
Noise on sediment level in Round 600 mm	0.2 m
Measurement uncertainty	0.05 m

Table 5.2: Setup of the twin experiment when one type of reach shape is added to the state.

For the measurement creation, the results of a single ensemble member of the stochastic simulation is used. The noise on the state has been reduced when creating this, because the intention is to create measurements that are mainly the result of a perturbed sediment level in the reach with shape 'Round 600 mm'. The (only) perturbation applied in this experiment is the modification of the initial sediment level.

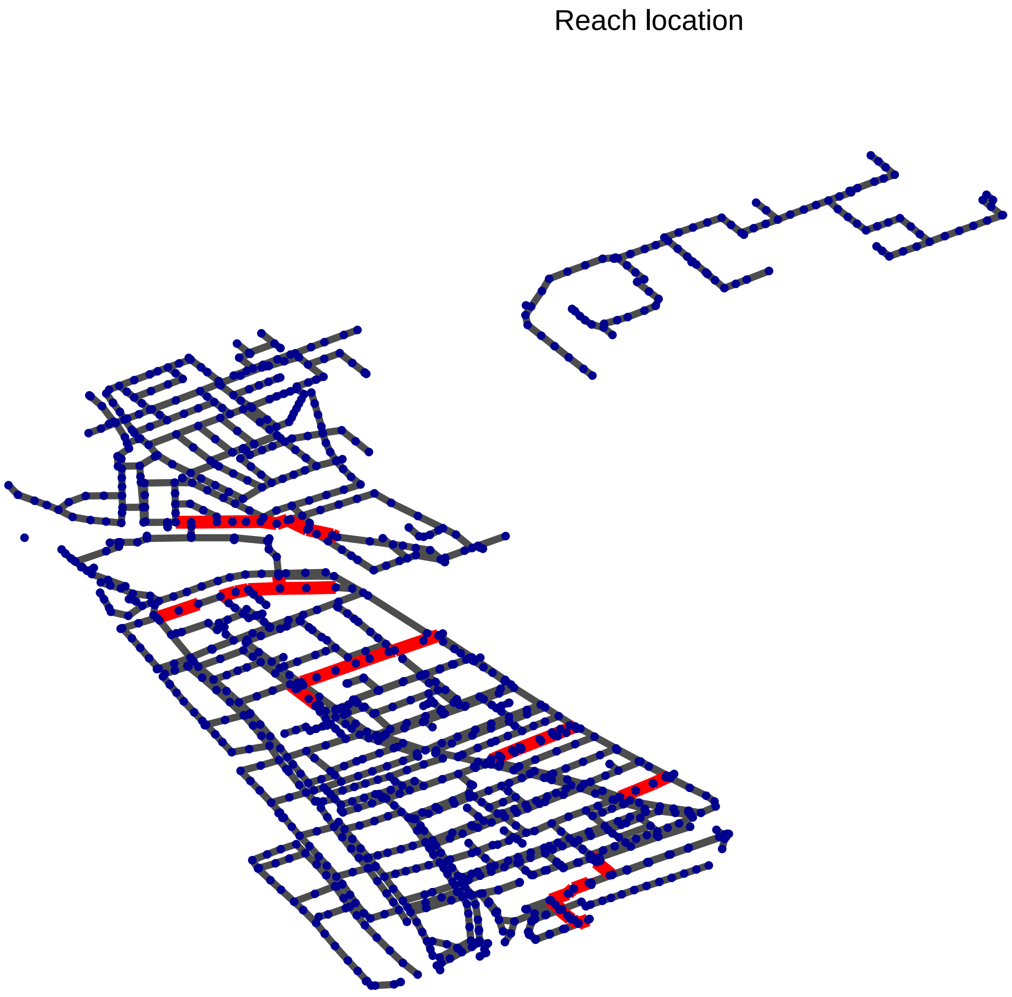


Figure 5.14: This figure shows the locations of the reach type 'Round 600 mm'. This is a general type and appears on several places in the system.

For this twin-experiment, the results of the same two measurement points of the system are shown in figure 5.16. The results of the sediment levels in the specific shape type ('Round 600 mm'), can be seen in figure 5.15.
The results of this twin-experiment are quite good. The waterlevels in the observation points, figure 5.16, are adjusted in the right direction. More importantly, the sediment level in the reach type 'Round 600 mm' is adjusted correctly as well, see figure 5.15. After the assimilation of the first five observations this sediment level reaches the value very close to 0.5, which was the value with which the observations were created.

Figure 5.17 shows the absolute difference between the ensemble mean and analysed state, similar to figure 5.12. In this case, the waterlevels at several locations are adjusted more drastically than in the non-perturbed case. This is the case specifically around locations of the perturbed reaches, which is of course no surprise since the waterlevels at these locations are directly influenced by these perturbations.

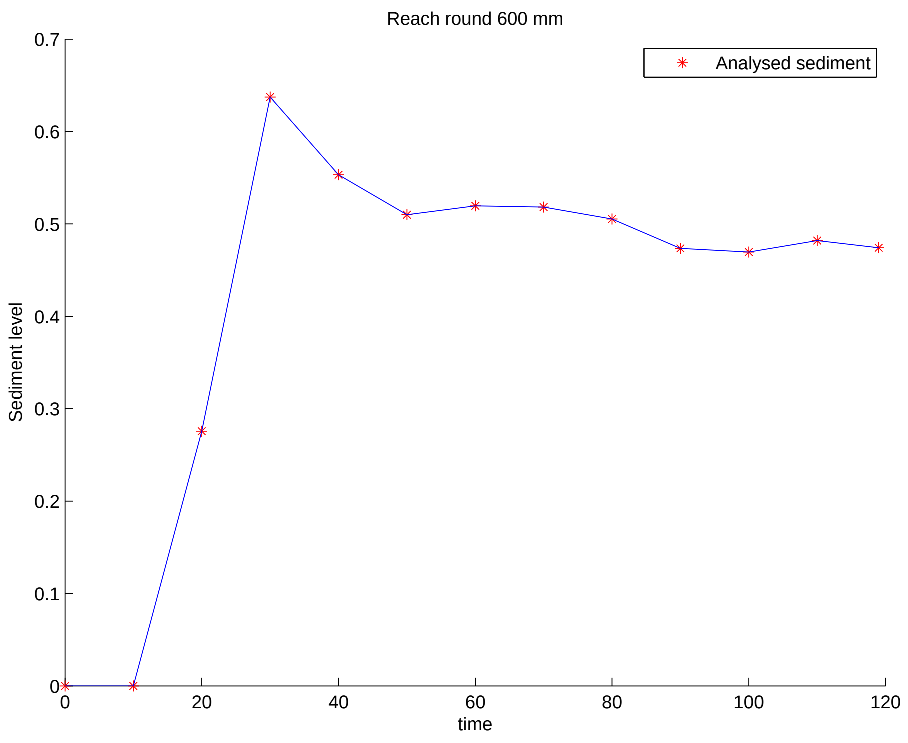
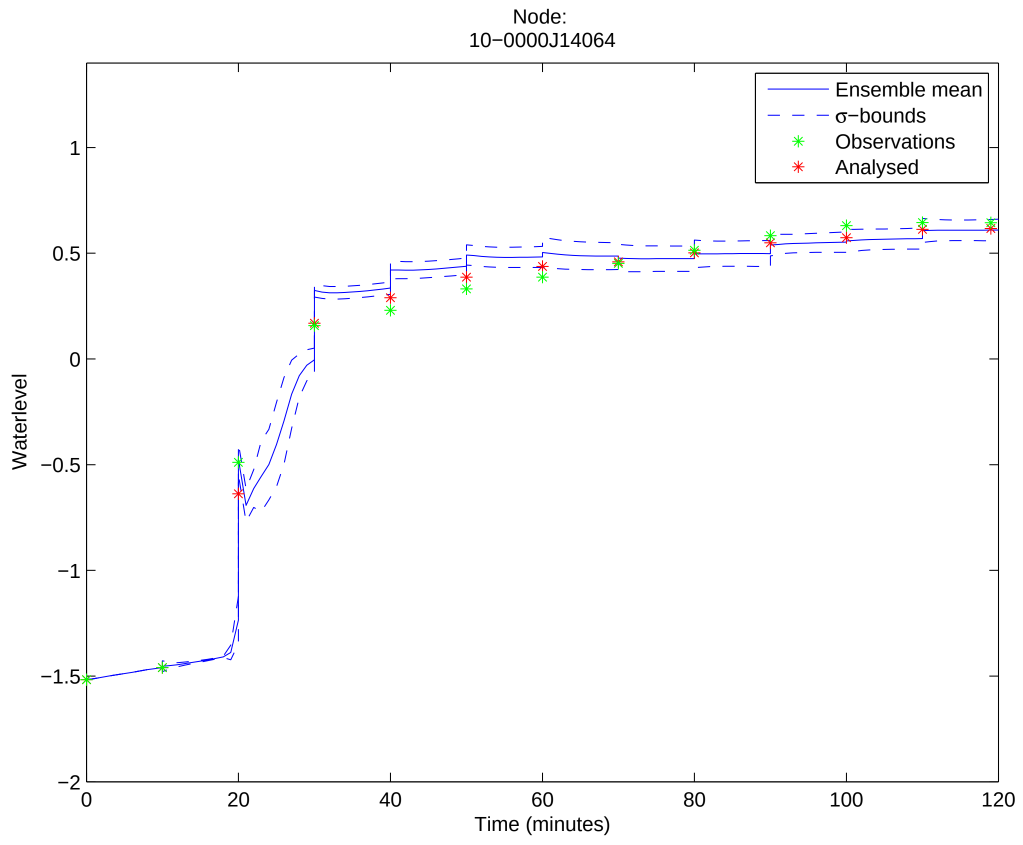
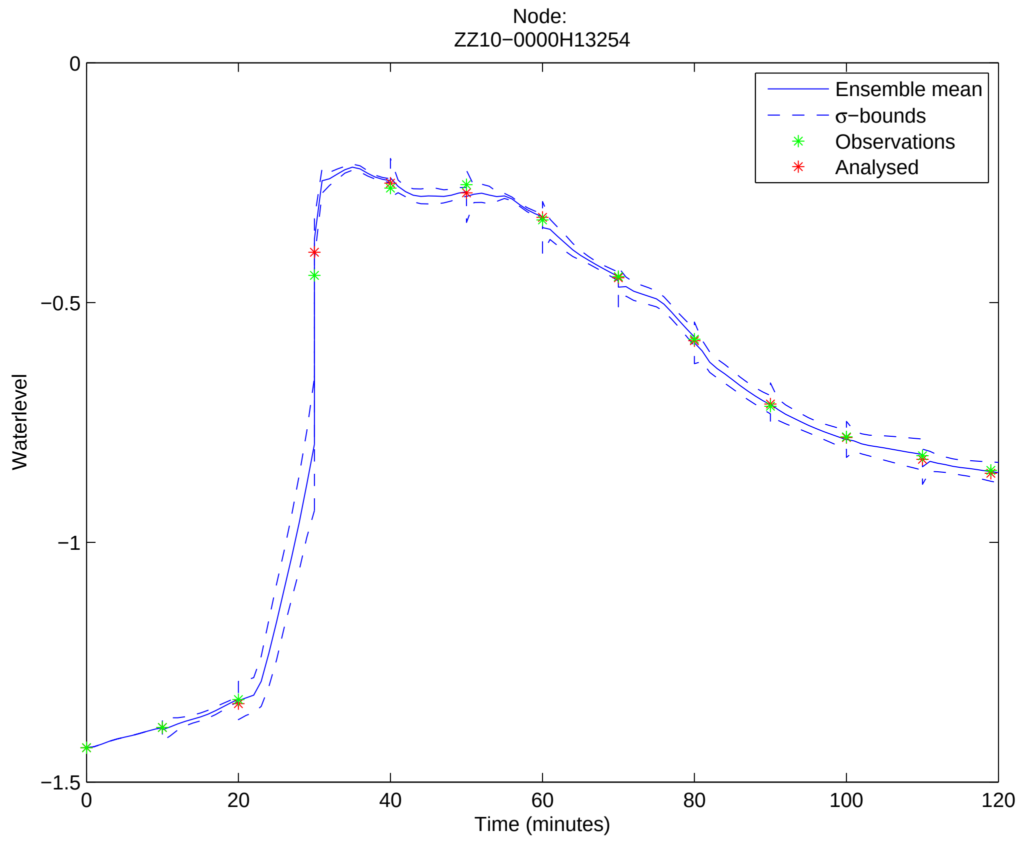


Figure 5.15: The sediment level in time as observations are assimilated.



(a)



(b)

Figure 5.16: The observed waterlevel, ensemble mean and its σ-bounds and the analysed waterlevel in two observation points.

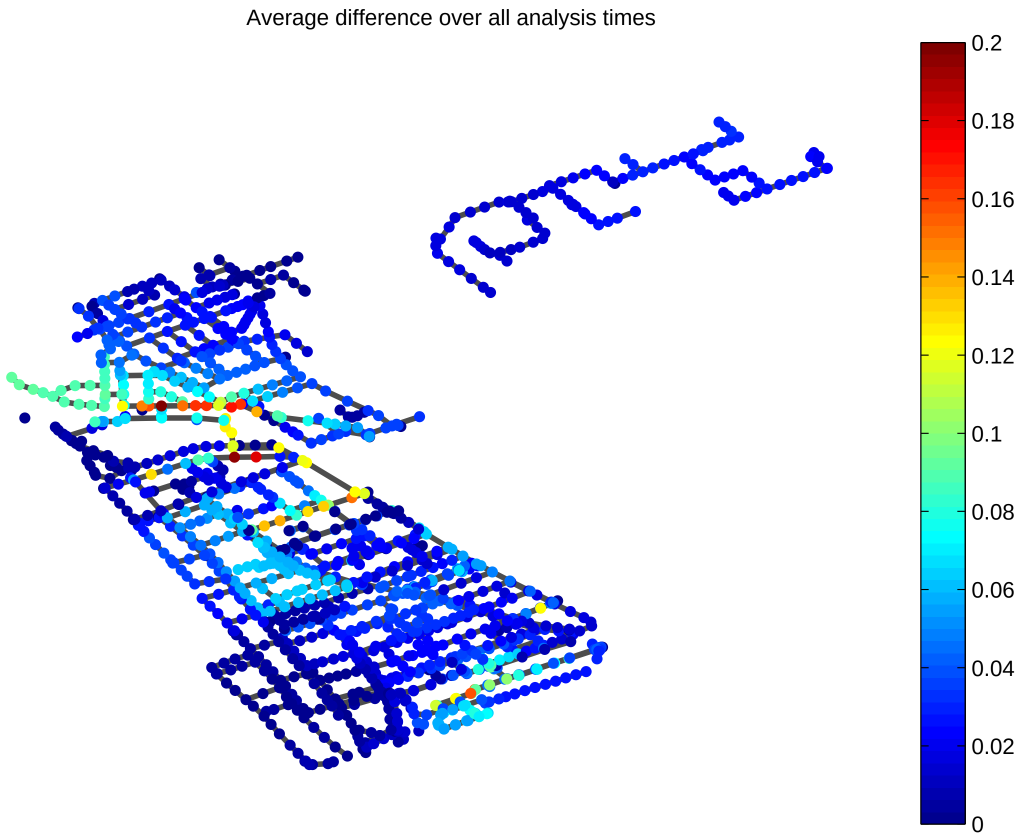


Figure 5.17: The absolute difference between the ensemble mean and analysis in each point.

The flooding that occurs in the system in this simulation can be seen in figure 5.18. There are several more locations in which flooding occurs in this case, where the more interesting locations are in the north east of the system. This area contains none of the perturbed reaches and the only connection to the rest of the system is by a pump which has the suction side in this particular part of the system. This means that there should be no flooding differences due to model attributes in this simulation compared to those of the non-perturbed one. This means that the differences in flooding is due to spurious correlations that induce an update in the waterlevels at these locations.

In the rest of the system, there are several more locations in which flooding occurs. This is the result of the pertubed reach being adjusted in combination with waterlevel adjustements from the assimilation.

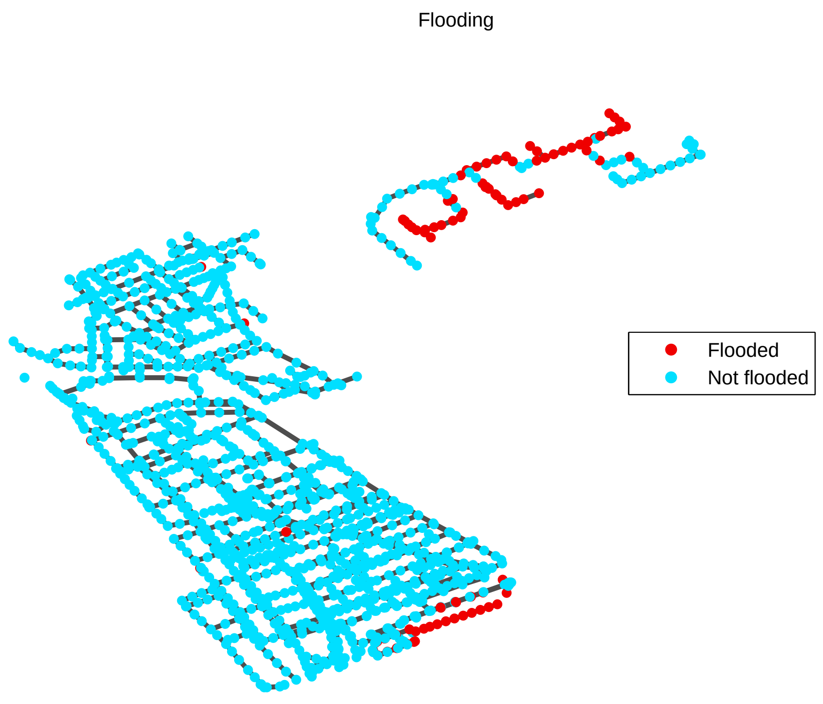


Figure 5.18: Flooding that occurs when the sediment level in reach type 'Round 600 mm' is perturbed (setup according to table 5.2).

One single reach in the state

The model can be made a bit more complex by defining individual shapes for each of the reaches in the system. This leads to the possibility for the sediment level in single reaches to be adjusted at assimilation times instead of, as seen in the above example, adjusting a type of shape that is applied to multiple reaches.

Measurement creation	
Pipe diameter Initial sediment level in ZZ10-0000J12466-10-0000J13010 Initial sediment level in other reach types	0.3m 0.2 m 0 m
Model noise method	Boxing method
Noise on the waterlevels	0.05 m
Noise on sediment levels	none

Assimilation	
Ensemble members	20
Initial sediment level overall	0 m
Model noise method	Boxing method
Noise (uncertainty) on nodes in state	0.05 m
Noise on sediment level in ZZ10-0000J12466-10-0000J13010	0.05 m
Measurement uncertainty	0.05 m

Table 5.3: Setup of the twin experiment when a single reach type is added to the state.

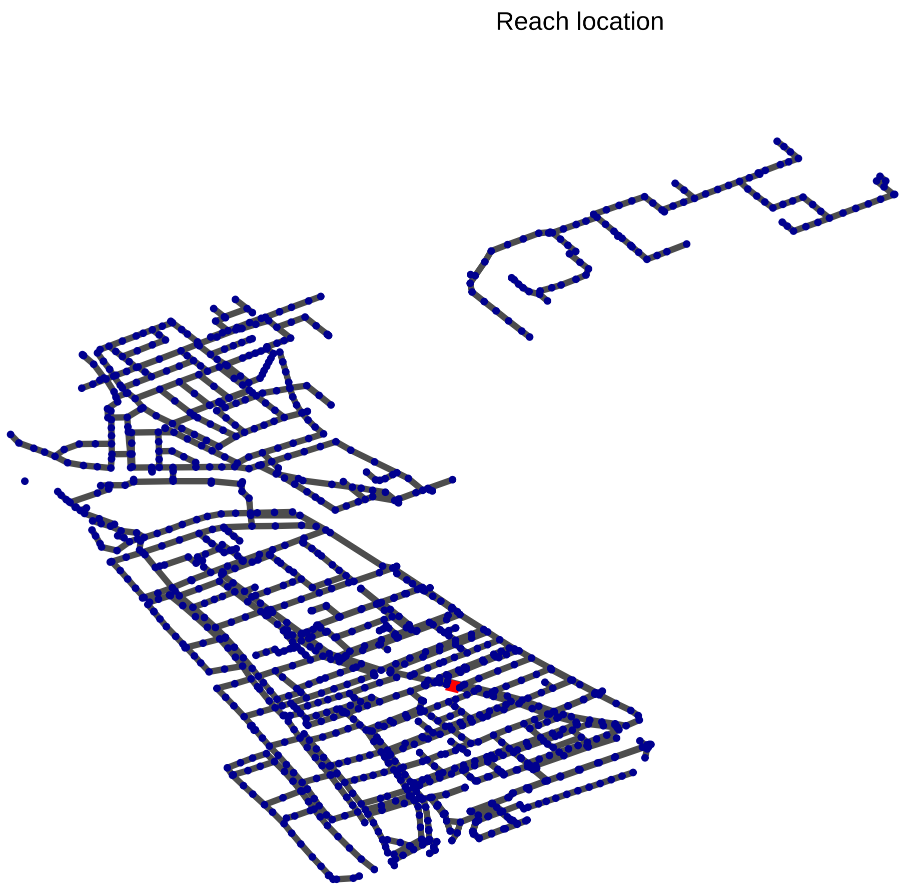


Figure 5.19: The reach that is added to the state vector is indicated in red.

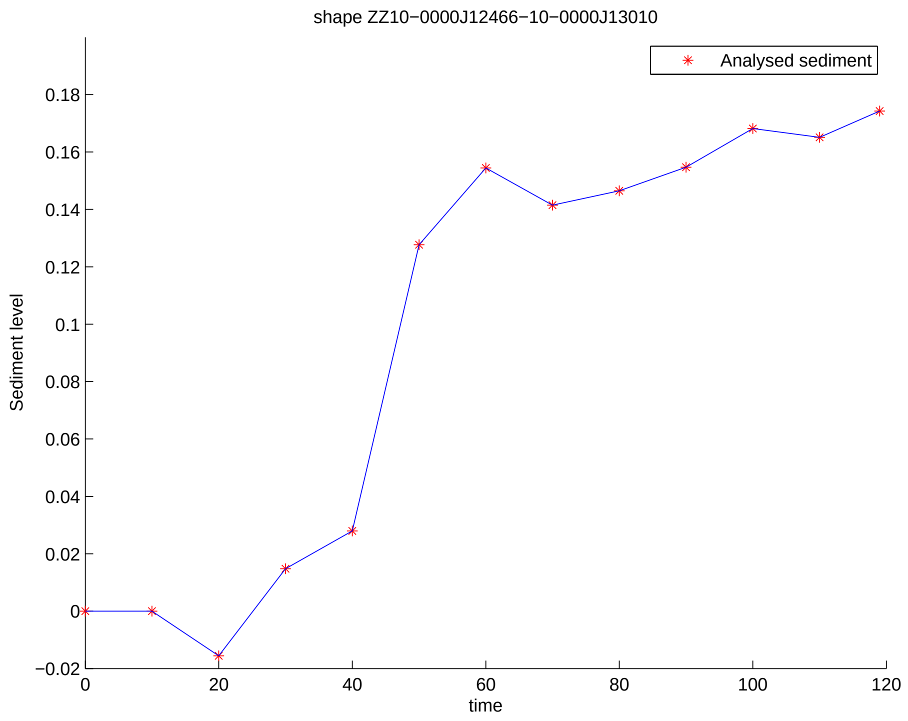
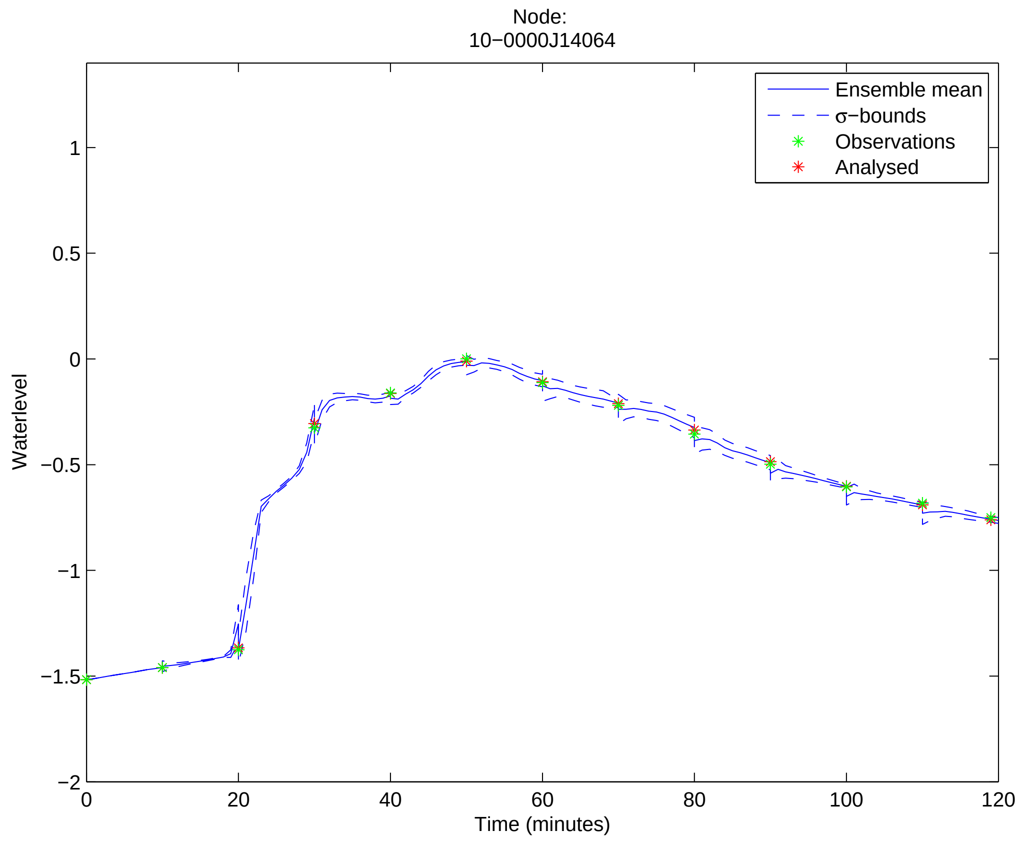
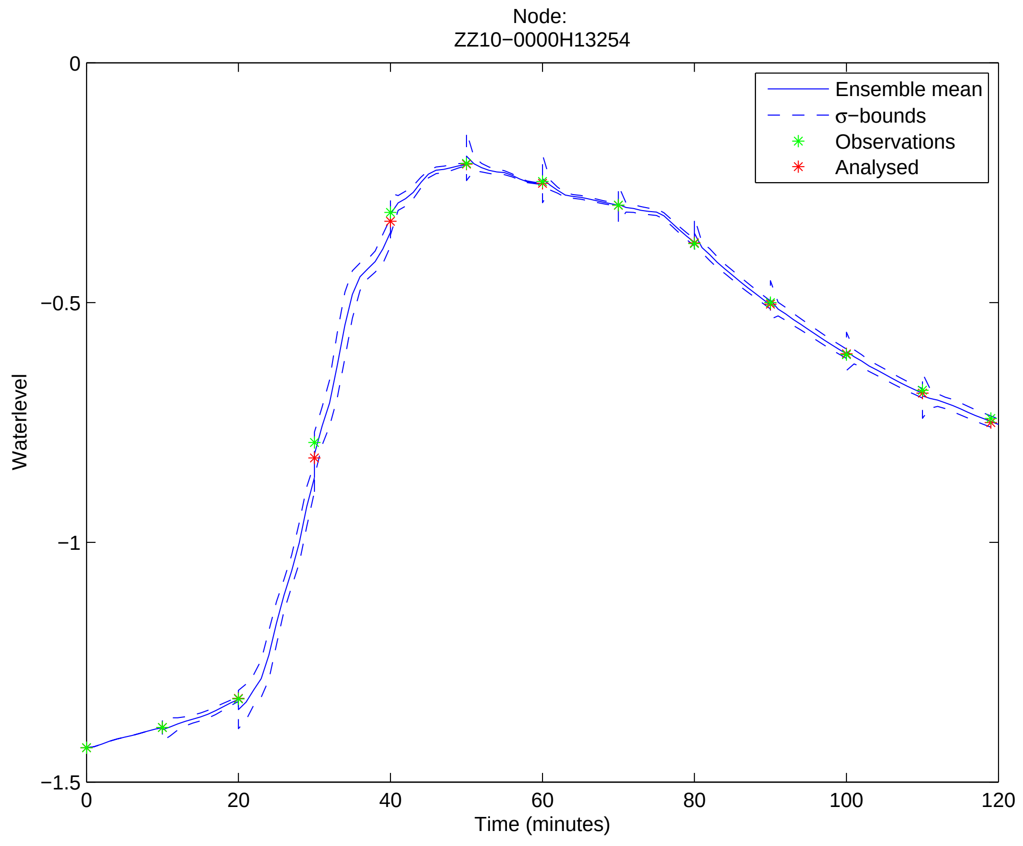


Figure 5.20: Resulting sediment levels when a single reach shape is perturbed.

The assimilation results for this single reach can be seen in figure 5.20. This figure shows that after several assimilation steps, the sediment level in this reach is adjusted upwards. At the end of the simulation, the sediment level is close the the level with which the observations are created, just under 0.18 at the end of the assimilation whereas the observations were created with a value of 0.2.



(a)



(b)

Figure 5.21: The observed waterlevel, ensemble mean and its σ-bounds and the analysed waterlevel in two observation points.

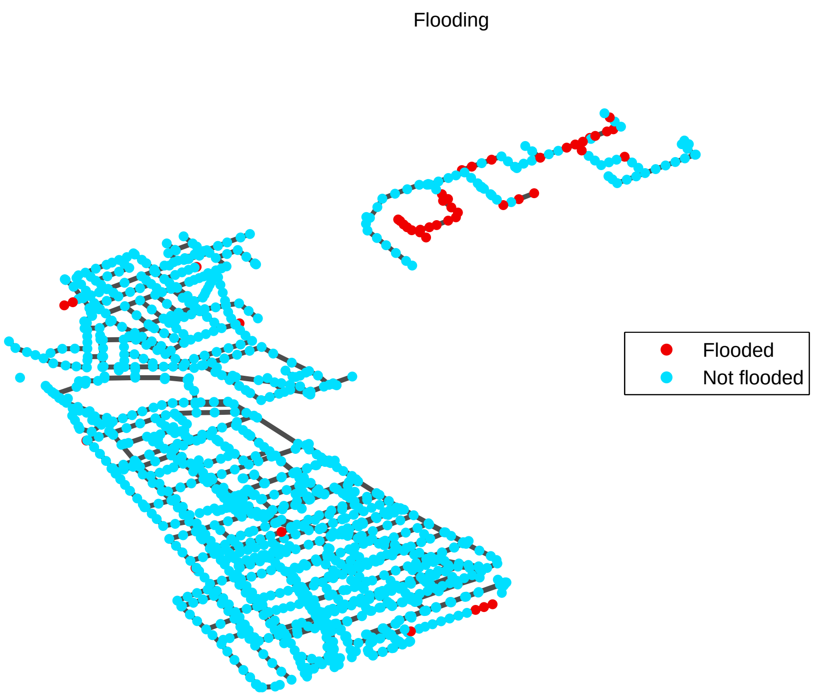


Figure 5.22: Flooding that occurs when the sediment level in a single reach is perturbed (setup according to table 5.3).

The flooding that occurs in the system for this simulation can be seen in figure 5.22. This figure shows a high resemblance with figure 5.13 which is not surprising. Since there is only one reach that is perturbed, the simulations should be very much alike. This means that most of the flooding that occurs in this case is also mostly due to the rainfall event that is used.

Chapter 6

Conclusions and future research

In this report, the blackbox coupling for Sobek and OpenDA is improved and tested for performance. In the development of this blackbox coupling, certain (small) issues have been detected and have either been resolved or an acceptable work-around has been devised for these.

6.1 Summary of resolved Sobek blackbox issues

Two of the issues that came into play when developing the blackbox coupling for Sobek were the restart errors that are discussed in section 3.3.3. Both of these flaws came to light when trying to use Sobek restarts from within OpenDA.

The first concerned restart timing with Sobek. The problem was that restarting at some intervals would induce a discrepancy between restarted runs and full simulations while restarting at some other intervals would lead to perfect matches between these results. The proposed workaround for this is to restart at 10 minute intervals to avoid the discrepancy in results.

The second issue in the restarts was the one where pumps in the system would not retain their status from the end of the previous simulation interval. Instead these pumps were set to 'off' whenever a simulation was (re)started and their status would only depend on the current state, i.e. waterlevel, at the suction side of these pumps. This lead to discrepancies in resulting waterlevels around these pumps, for example figure 3.6, which shows the waterlevel at the suction side of a pump in the system of Delft. The workaround that has been applied to counter this flaw, is to not take observations from a full simulation but rather from a restarted run using OpenDA.

It should be noted that both workarounds are devised to avoid dealing with these issues. The issues themselves, however, are still present in Sobek.

Another issue that caused assimilation with OpenDA to give faulty results came from the way observations were used in this particular blackbox coupling together with a bug in the OpenDA software part that was used for this. In appendix C, results of the assimilation of a non-perturbed twin-experiment can be seen. With the waterlevels being updated to far away from the ensemble (and observations) at assimilation times, it is clear that these results are far from acceptable.

Eventhough this flaw has already been fixed in the OpenDA software, this way of using observations was already discarded for the blackbox coupling. This was because together with the workaround method to avoid the pump error, a different way of using observations in OpenDA was implemented and as such, the part of the OpenDA software containing the bug was no longer used.

6.2 Summary of blackbox results and conclusions

In testing the blackbox coupling, two main functionalities are tested. Namely the calibration of parameters and the assimilation of observational data in time. It turns out that both functionalities show great promise for further use in modelling urban drainage systems.

Calibration

Calibration of several parameters using OpenDA is promising. This can be seen for the surface areas in tables 4.3 and 4.6 and also for rainfall runoff parameters in tables 4.11 and 4.12. Though the calibration of some parameters can be quite far off at times, this is what happened when performing a calibration which led to the result in the last row of table 4.12. In this case, the parameter 'rf8' was "calibrated" to have a value of 0.87 while it should have been calibrated to 0.2. Obviously there are some improvements to be made here.

Besides the calibration of parameters, some identifiability analysis has been applied. The method described in section 2.5 and applied in section 4.3 of parameter analysis can be used to determine if and which parameters are highly correlated. It can also be used to determine if parameters can be identified seperately for the calibration. When dealing with a high number of different parameters the situation can occur that some parameters cannot be identified seperately but in a group they can be. It can be advantageous to collapse these parameters into one group and calibrate them as one.

There are more parameters that can be used in the calibration. Think for example of friction parameters, several discharge coefficients. These parameters can also be used to simulate blockages in sewer systems. Including these in the calibration could bring certain blockages to light in the system as has been shown in [Stegeman, 2012].

Assimilation

Using the blackbox wrapper to assimilate observations as they become available is the most important part of the work. In section 5, the assimilation using the blackbox wrapper is discussed. Starting by determining a good set of observation locations that can be used in combination with the ensemble Kalman filter. A short discussion on a previously used set of observation locations and the reason these were discarded can be found in appendix B. The new set, shown in figure 5.10, is chosen in such a way that it provides a good amount of information for the entire system. The criteria used in order to determine the set of observation locations can possibly be refined. Any new or refined criteria, however, will still be based on the correlations of observation locations with surrounding nodes.

The assimilation of observations in time, using the new observations is tested. A twin-experiment in which no perturbations are applied yields good results. Because there are multiple observation locations and there is no localisation applied, spurious correlations can still occur. The waterlevel results, however, are very well adjusted throughout the assimilation timespan. In order to simulate blockages in a system, some experiments have been done by making the sediment levels in the reaches a time dependent parameter and taking these along in the assimilation. This results in an alternative way of 'calibrating' parameters, because these parameters will then be taken along in the state vector and will be adjusted in time as observations become available. The results from this assimilation method are very good, as can be seen in figures 5.16 and 5.21 show the update for sediment levels in time in two different cases and figures 5.15 and 5.20 show the waterlevels at some observation locations that belong to these cases. The latter figures show that the differences between observed waterlevels and calculated waterlevels in the model are very small. In fact, most observations are still within the σ-bounds of the ensemble. The ensemble spread is also quite small. However, the correlation with the parameter of interest, the sediment level in this case, is very well represented. This follows from the fact that the sediment levels are quickly adjusted in the right direction.

Defining uncertainties when applying data-assimilation is very important. When assimilation is done with certain time dependent parameters in the state, the greatest uncertainty would the probably be in these parameters and the uncertainty on the waterlevels in the state would be smaller. Basically, this leads to the assumption that with 'perfect' parameter values, the model output will represent waterlevels in a perfect way. Of course, this an exaggeration but it does illustrate the need for well defined uncertainties on the entire state vector.

6.3 Conclusions

Several conclusions can be drawn from this work and regarding the blackbox coupling, calibration and data-assimilation for urban drainage models.

- The basic blackbox coupling works and is easy to set up, starting from a model in Sobek. It is possible to expand it with the implementation of several more exchange items, if necessary. This does, however, require some insight in the set up of the OpenDA package since some classes previously used in making exchange items should no longer be used.

- Observation locations for the ensemble Kalman filter have been determined by running simulations with OpenDA with a reasonable amount of ensemble members (100 ensemble members were used in this work). From these simulations, correlations between points can be determined which is a good indication for the information that these points provide about the rest of the system. Though the sewer system of Delft is rather flat, several locations show a significant lack of correlation with the surrounding points. This can influence the performance of the ensemble Kalman filter. For this reason, the method of finding new observation locations is based on distance between observation locations and moreover based on correlation with the surrounding network.

- Two different ways of parameter estimation have been investigated. The first is the calibration by means of the DuD method. In this method, parameters are assumed to be constant in time and will be estimated by running a full simulation several time, trying to minimise a cost function. This cost function is usually a weighted least squares problem. Results are promising but require some more research in order to try to perfect the method.

- Another way of estimating parameters involves the possibility of making parameters time dependent and taking these along in the assimilation with the (ensemble) Kalman filter. This will update parameter values in time as the simulation continues. This has been shown for the sediment parameter and works quite well over the (small) time period of two hours. This method can also be applied for several different parameters. For example rainfall runoff or friction parameters. This is a useful method of estimating parameter values since some parameters, such as the rainfall runoff delay, are assumed to be constant while in practice these can vary. The variation of these parameters is usually dependent on events like precipitation in the model.
- Parameter analysis is a useful tool to show the relation between different parameters. It can be used to show if parameters can be identified individually or rather only in a group and the analysis can be used to gain some insight in the correlation of parameters.
- Including model uncertainty by means of imposing noise on the nodes in the simulations has been investigated. The noise in a specific node will have a high correlation with the noise on the connected nodes. The method used in this work is to put the same values on the nodes in certain blocks in the system, i.e. the method applied in figure 5.2. This method has a high advantage over putting uncorrelated noise on the nodes, since the latter tends to dissipate rather quickly. However, there is room still for some improvement in the noise modelling method. The best way to impose correlated noise on the system would be to base this on the reaches between them. Moreover, the total length of the reaches between two points would determine the correlated noise.

6.4 Future research

This report presents a good beginning of data-assimilation applied to urban drainage systems. Results are promising and show that with a relative small set of observations, 15 observation locations for a system of 1300+ nodes, the blackbox coupling performs quite well. However, there are several things that can be improved or that can still be done. These can be divided into two groups. Several suggestions are concerned with specifics in urban drainage modelling:

- **Surface waterlevel uncertainty.** In the case of Delft, the surface water is an interesting aspect. The canals in Delft have a very limited freeboard, which makes sewer overflow an interesting issue. Sewer overflow crest levels should obviously be above the surface waterlevel to prevent any surface water from flowing into the sewer system. Should the surface waterlevels rise then it is likely that there is an inflow from the surface water into the sewer system. For future research it is a good idea to take the surface waterlevels (these are implemented as boundary nodes in Sobek) into account by defining some uncertainty on these. This would lead to better information on the flooding of areas in Delft as a consequence of increased surface waterlevels.

- **Pipe friction parameter calibration, orifice discharge** Blockages in the sewer system have been simulated by playing around with the sediment levels in various reaches in the system. Another way to simulate and possibly identify locations of blockages in the system would be to simulate these using the discharge coefficients in orifices. The latter method has been applied by [Stegeman, 2012]. However, the challenge remains to calibrate or perhaps assimilate blockages anywhere in a system while the measurement locations are fixed, i.e. not in any specific optimal locations for the particular blockages to be calibrated.

  A start with this method using Sobek and OpenDA has been made and some results can be seen in appendix D. In this method, the opening height of the orifice is used as the parameter value since there is no discharge coefficient present for these orifices. The results briefly discuss the identifiability of these types of parameters. However, there is yet much to do in this area.

- **Extra observational data.** Observational data is provided by a measuring network inside the sewer system. This will provide the waterlevel inside manholes for several locations in the sewer system. It can happen that, at a certain time, pluvial flooding occurs without an obvious explanation in case of an intense storm event. This could be an indication of some sort of obstruction inside the sewer system. It is undeniable that the flooded areas provide worthwhile information regarding the location of possible blockages. It poses a challenge to then include the observed flooding in these locations into the model calibration to better determine the location of the blockages.

- **Practical observation locations.** An aspect in determining observation locations that was also considered in [Post, 2012] is the practical implementation. It is obvious that of all the locations in the system, there are quite a few that cannot be used as observation locations for practical reasons, locations that are unreachable or are part of a very busy road for example. In order to determine practical observation locations, these things have to be taken into account and locations that will not work should be omitted from the list of possible observation locations.

  Another resulting feature from the work in [Post, 2012] was creating pairs of observation locations that have a significant correlation with eachother. This is convenient, because if in one location the measurement equipment fails there is not too much loss of information. Another use of these highly correlated observation locations is that one can be used for the assimilation and the other can be used as a form of validation. This attribute of the observation network could be combined with the creation of a network in section 5.4 to improve it even further.

The above suggestions are more concerned with urban drainage modelling. However, below there are some suggestions that can improve the predictions when applying data-assimilation.

- **Spatial correlation on waterlevel noise.** At this moment, the noise imposed on the waterlevels is done via the boxing method discussed in section 5.1.3. A better way would be to implement a method with a good spatial

correlation, like the proposed interpolated method (extension of the boxing method) in the same section.
An even more advanced method of modelling the uncertainty would involve basing the uncertainty in the nodes on connections between them, as mentioned earlier. This means that noise imposed on nodes will have a higher correlation as the distance of the connection between these nodes decreases.

- **Localisation.** When assimilating observations, spurious correlations play a role. These spurious correlations occur when observational data has a notable impact on locations that are far away. Localisation will annihilate these spurious correlations, which is especially useful in the case of a small number of ensemble members. Applying localisation in this case can be difficult because distance in a sewer model is basically defined as the sum of reach lengths between the two points.

Appendix A

Noise model implementation methods in OpenDA

As part of the research in the blackbox coupling for Sobek and OpenDA, there have been two different implementations of noise models in use. For this particular project in OpenDA, two different implementations of the noise model have been used. These implementations are not to be confused with the different methods of applying noise to the state vector discussed in section 5.1.3. These implementations refer to the way the noise model is configured in the OpenDA config files.

The first, older implementation is the one where the noise model that is used is defined in the stochmodel config file. A single entry or part of the state vector is listed within a noise model config block and together with this partial state vector, the noise model is defined. An example of such a configuration can be seen below.

```
<noiseModel>
  <vector id="ZZ10-000H14088F"/>
  ...
  <ar1Model operation="add">
    <stdDevColouredNoised value="0.05"/>
    <decorrelationTimeScale>PT60M</decorrelationTimeScale>
    <noiseModelPeriod>PT10M</noiseModelPeriod>
  </ar1Model>
</noiseModel>
```

In this configuration, the AR(1) model and its attributes like model period and decorrelation time (same as correlation time) are defined with this configuration block. The vector ids defined in this block will all get the same noise value added to them. If the same noise value were to be put on the entire state then the entire state would be defined in terms of these vector ids and in a whole be put in the block.

The new implementation of the noise model can be seen in the following example. In this implementation, the noise model definition is done in a separate file, as can be seen in the second part. The start time, noise model period and end time are deduced from the simulation time span, the standard deviation and time correlation are defined in the file. In the stochastic configuration, the modelExchangeItem id refers to the part of the state vector that is subject to the particular noise model. In the stochastic model configuration file:

```
<noiseModel id="NodesNoiseModel"
  className="org.openda.noiseModels.TimeSeriesNoiseModelFactory"
  workingDirectory=".">
  <configFile>NoiseModelForNodes.xml</configFile>
  <exchangeItems>
    <exchangeItem id="noise-for-nodes" operation="add">
      <modelExchangeItem id="ZZ10-0000G12042"/>
      ...
    </exchangeItem>
  </exchangeItems>
</noiseModel>
```

The separate configuration file for the noise model:

```
<timeSeriesNoiseModelConfig>
  <simulationTimespan timeFormat="dateTimeString">199501010000,
        199501010010,...,199501010200</simulationTimespan>
  <timeSeries id="noise-for-nodes" location="node" quantity="h"
        standardDeviation="0.05" timeCorrelationScale="60.0"
        timeCorrelationScaleUnit="minutes"/>"
</timeSeriesNoiseModelConfig>
```

The stochastic model configuration file can obtain several blocks of noise models as seen above. In each block there are several vector ids or modelExchangeItems defined which refer to (elemenst of) the state vector. The way these groups are formed has to do with the method of imposing noise on the state vector.

Appendix B

Old observations

Before determining a set of observation locations which give a decent amount of information about the system, described in section 5.4, there was already a set of observation locations used by OpenDA for this particular system. This set of observation locations was the result of the research done in [Post, 2012] and is considered an optimal set of locations for calibration. However, when dealing with the ensemble Kalman filter, the correlation of these observation locations with the rest of the system is very important. This is where the old set of observation locations was insufficient and thus it has been improved specifically for the ensemble Kalman filter. The old set of observation locations is visualised in figure B.1, as well as the corresponding information value.

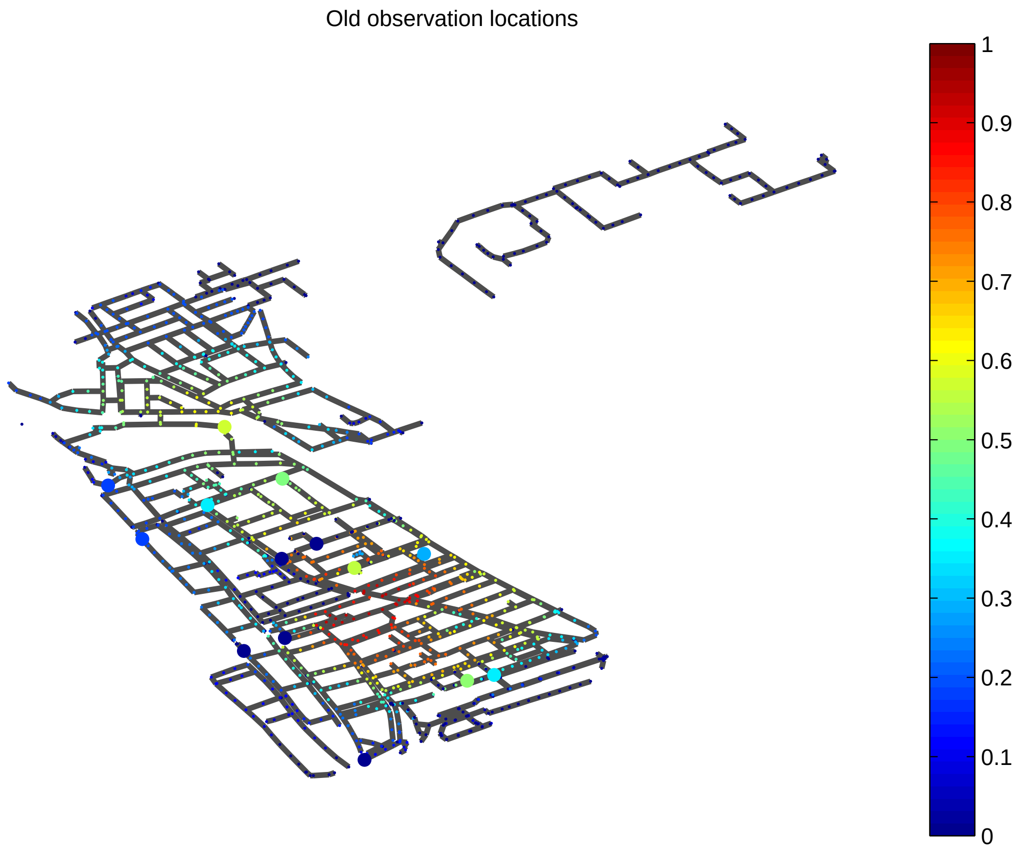


Figure B.1: This figure shows the locations of the old set of observation locations and the amount of information they give.

This figure has been created using the same criteria as in section 5.4. It becomes quite clear that there are several points with a very low information value and even some points that do not even meet the minimal criteria. This is where the old set of observation locations falls short for use with the ensemble Kalman filter.

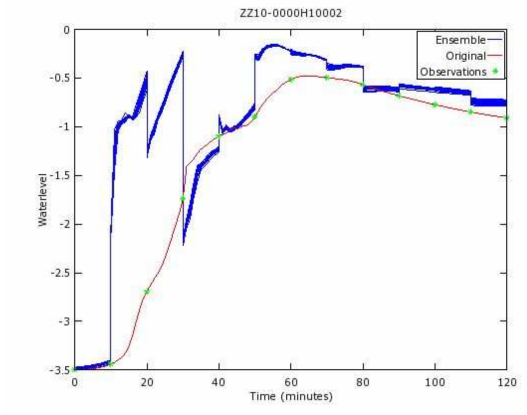
One of the reasons for the low information value in some points has been discussed in section 5.3. Here it was mentioned that there are parts of the system that are only connected to the rest of the system by a few internal weirs and as such are quite isolated from the rest.

Other reasons for the low information can include the fact that points are on the outskirts of the system or elevation differences between consecutive points. In the first case, there are simply not enough points in the vicinity for the particular node to qualify as an observation point. In the latter case, when a point is elevated above its neighbours, any water will flow away rather quickly and with this most likely also the correlation with surrounding points.

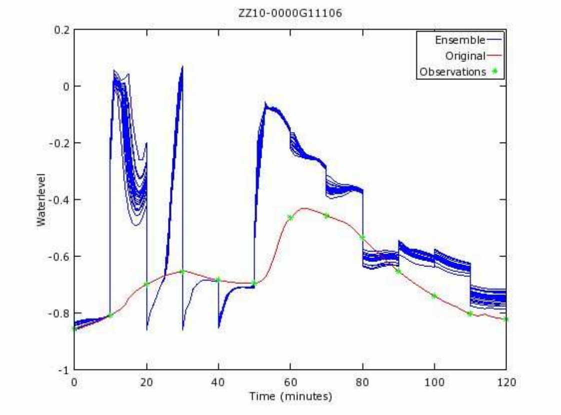
Appendix C

Bugged blackbox coupling

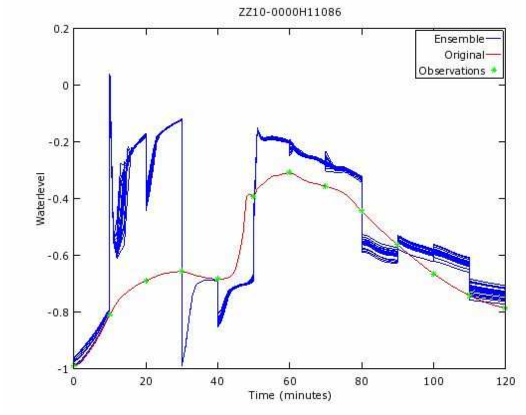
First attempts at assimilation using the blackbox coupling gave very weird results. Assimilating observations that were generated using a non-perturbed simulation would lead to drastic waterlevel updates at assimilation times at several locations. Some of these updates are visualised in figures C.1. Attempts to find the cause for this behaviour were done by, among other things, increasing the ensemble size and using different noise model implementations (see appendix A for two different implementations).



(a)



(b)



(c)

Figure C.1: The waterlevels at some observation locations while using the earlier bugged blackbox coupling.

The results in figure C.1 are created using an ensemble size of 20. The results in figure C.2 were created using an ensemble size of 100. These results still show a wrong update at analysis times.

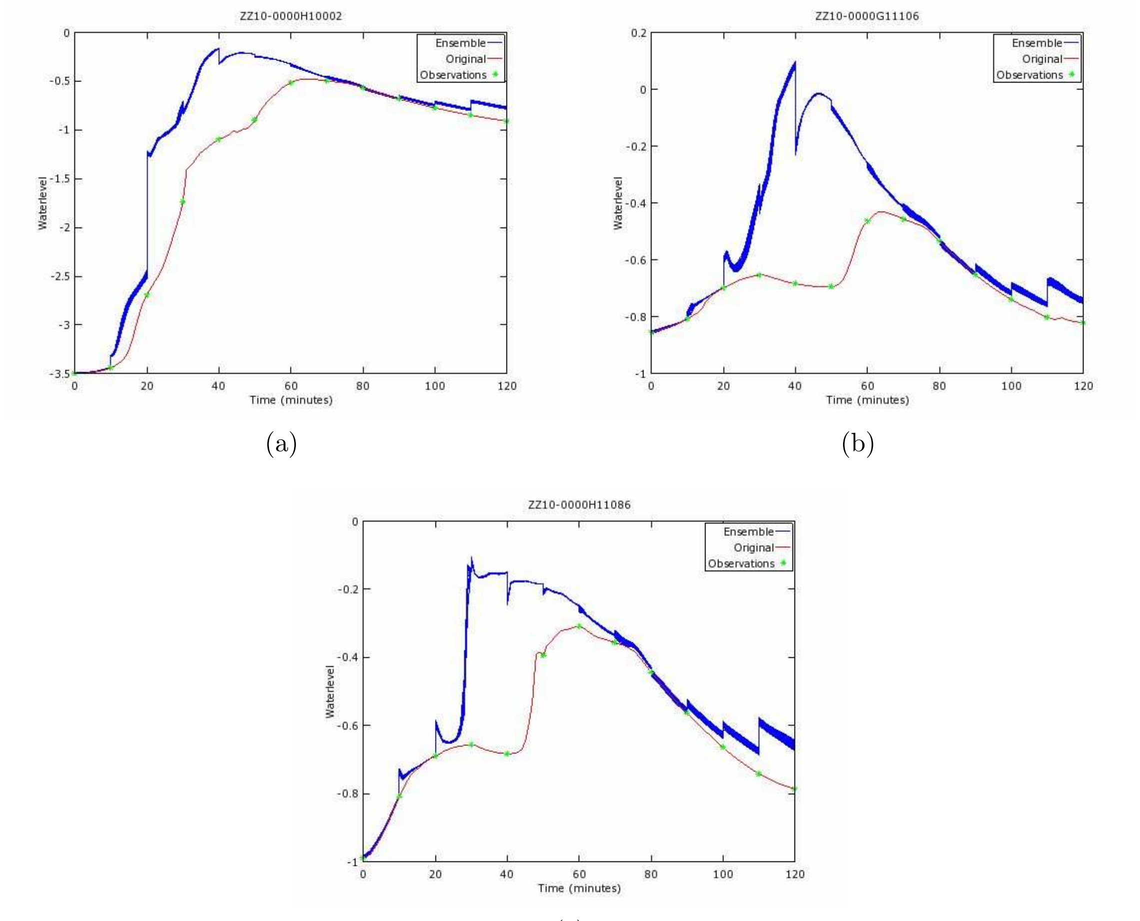
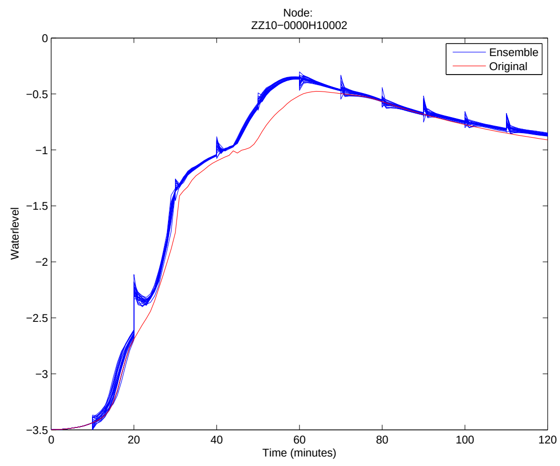


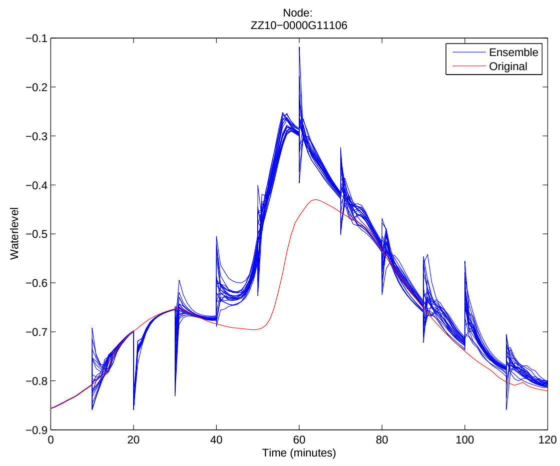
Figure C.2: Waterlevel results using the bugged blackbox coupling in a second experiment where 100 ensemble members were used.

The third set of figures, figure C.3, was created by using a different implementation of the noise model in an attempt to create plausible results. These figures show faulty behaviour as well.

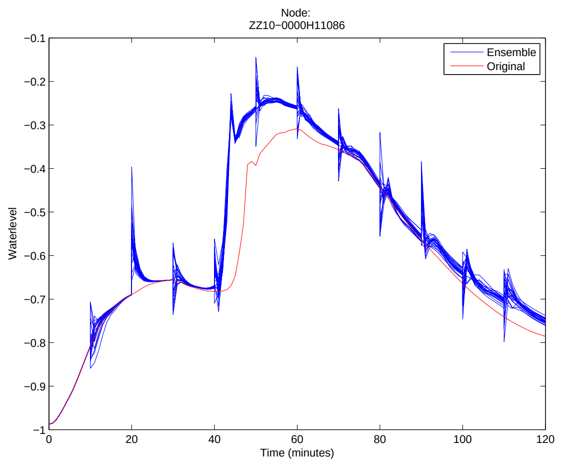
Eventually, the faulty behaviour disappeared when implementing a different way of creating and using observations. The old way of creating observations was to run a full Sobek simulation over the entire timespan. Then the `calcpnt.his` file, which contains information on waterlevels at all nodes, would be copied and used as input for the observational data. In OpenDA, this is done by the IoObjectStochObserver class.



(a)



(b)



(c)

Figure C.3: Results from yet another bugged blackbox simulation. This time using a different implementation of the noise model.

A different method of creating and using observational data was proposed. In this new method, Sobek would be used in combination with OpenDA to run a stochastic simulation which would be restarted at the same intervals as the assimilation would take place. However, the values would not be updated by an analysis step. By not choosing the uncertainties too large for this stochastic simulation, it is possible to create a set of observational data that is a good representation for the model setup used. Choosing a set of locations as observations with dummy values and observation times will allow OpenDA to run the stochastic simulation, restarting at the observation times and writing the values at these times to a resultwriter file. The values that are given to the observations for the stochastic simulation are not used, thus them being 'dummy' values. With the results file that is generated, noos-observer files can be created for every observation location with good observational data for a twin-experiment and these can then be used in the assimilation.

Implementing this way of creating and using observations actually solved the problem of getting weird results from the earlier assimilation runs. As it turns out, there was a bug present in OpenDA for the specific type of observations used and this was then avoided by changing the way observations were used.

Appendix D

Modelling blockages through orifices

The modelling of blockages is done in section 5 by modifying the sediment levels in reaches. The choice was initially made because these aspects are easily added and modified in the model. The method of choice in chapter 5 is the ensemble Kalman filter. By making the sediment levels time dependent and taking them along in the state vector.

Another method that is proposed and (partially) implemented in order to simulate blockages is to add orifices in the model. This has also been the method of choice in [Stegeman, 2012]. Sobek basically allows for two different methods of adding orifices into the system. One of which is by cutting a reach by adding an extra orifice node. The other is to change the type of a reach to an orifice, as orifices are present in the system as both nodes and reaches. Due to time constraints, only an initial analysis is presented in this section concerning the addition of orifices in the system.

As mentioned, Stegeman also used orifices as a parameter for blockages simulation. Particularly, the discharge coefficient is used. This discharge coefficient is a factor for the amount of discharge in this reach. Thus, by adjusting this, effectively limiting the amount of water that flows through this reach. However, Sobek does not allow for any discharge coefficient to be modified (Stegeman used Infoworks). Through some inspection of the orifice structure type, the choice is made to implement the orifice as a reach. Some discussion led to the opening height to be the choice of parameter in this report. The opening height is a good parameter to influence the amount of discharge accros the orifice as it functions as a gate through which the flow can occur.

Initially, the choice is made to place two orifices in the current system which are not too far from each other. For these orifices, analysis is performed on the identifiability of these structures for several slightly different sets of observation locations. The locations for the particular orifices is chosen to be in the southern part of the system, as indicated in figure D.1 and the area with orifices is enlarged in figure D.2. For the remainder of this section, the orifice parameters are called 'orifice1' and 'orifice2', where 'orifice1' is the northern one and 'orifice2' is the southern one.

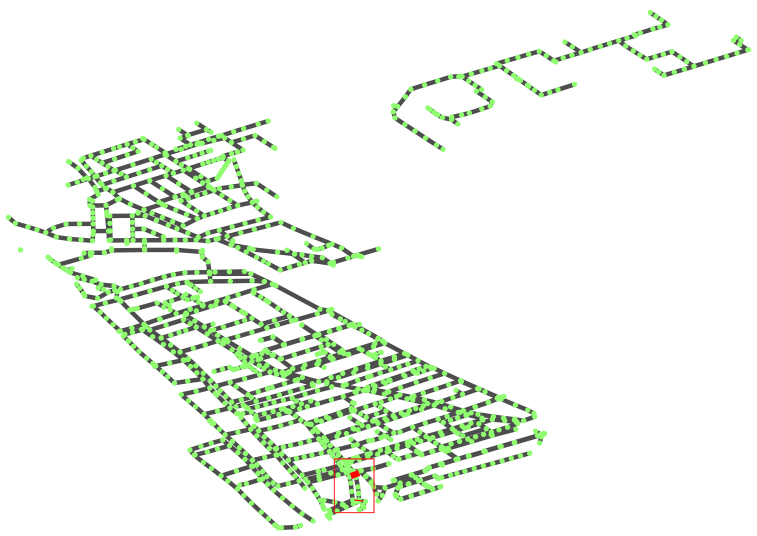


Figure D.1: Full network in which the orifices are located in the red rectangle.

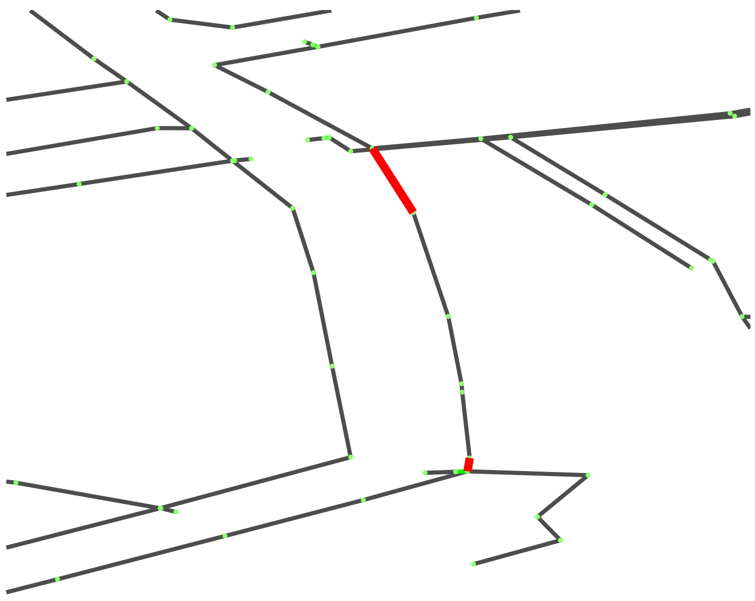


Figure D.2: Enlarged area showing orifice locations. The norther orifice is the parameter 'orifice1', the southern orifice is called 'orifice2'.

Analysis

The objective of this section is to determine the effect of observation locations on the identifiability of these orifice parameters, starting with the general set of observation locations. Other sets of observation locations include the nodes on each side of the orifices. The four present runoff parameters are also taken along in the analysis, as a means of reference. Four cases have been set up:

- The general set of observation locations, spread out over the system.
- The general set of observation locations plus two extra locations. The extra locations are the two nodes that are present at each side of 'orifice1'.
- The general set of observation locations plus four extra locations. The extra locations are the nodes that are present at each side of both orifices.

- A set of observation locations in the north eastern part of the system.

For each of these cases the singular values are calculted and the matrix V that goes with these values. There is no calibration performed since the Jacobian is calculated with the original parameter vector (i.e. optimal vector) as input.

Case 1

For the first case the singular values are

$$\text{singular values}: \; [10.8635 \quad 2.5172 \quad 1.1776 \quad 0.9297 \quad 0.3519 \quad 0.1258]$$

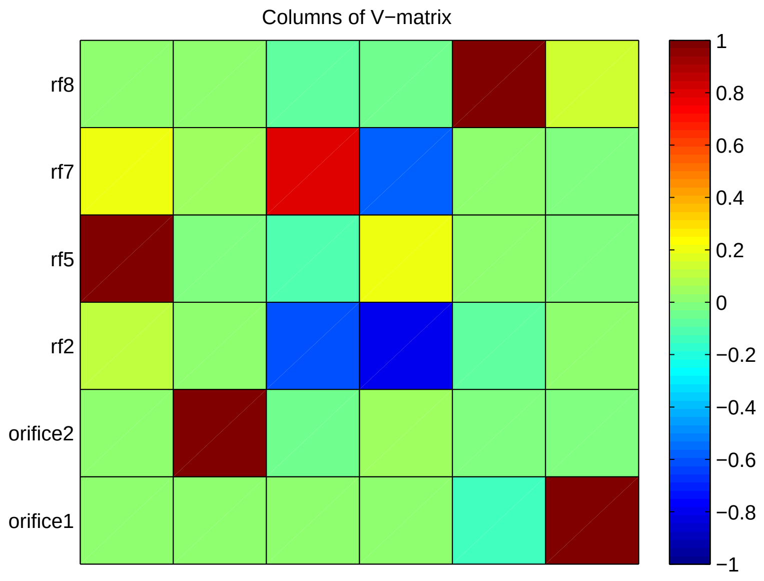


Figure D.3: Values of the V matrix for case 1.

This shows that none of the singular values take on an extremely low value, leading to the assumption that all of the column vectors of V represent reasonably identifiable parameters or parameter combinations. What is surprising in this case, looking at the matrix V, is that a lot of parameters are very identifiable. This excludes the parameters 'rf2' and 'rf7', which seem to be only identifiable in a combination.

Case 2

This case, in which the nodes next to 'orifice1' are added as observations, shows quite some improvement compared to case 1. The singular values for this case are given below as well as the columns of the V matrix in figure D.4.

$$\text{singular values}: \; [15.6799 \quad 10.9525 \quad 4.1205 \quad 1.2107 \quad 0.9637 \quad 0.3598]$$

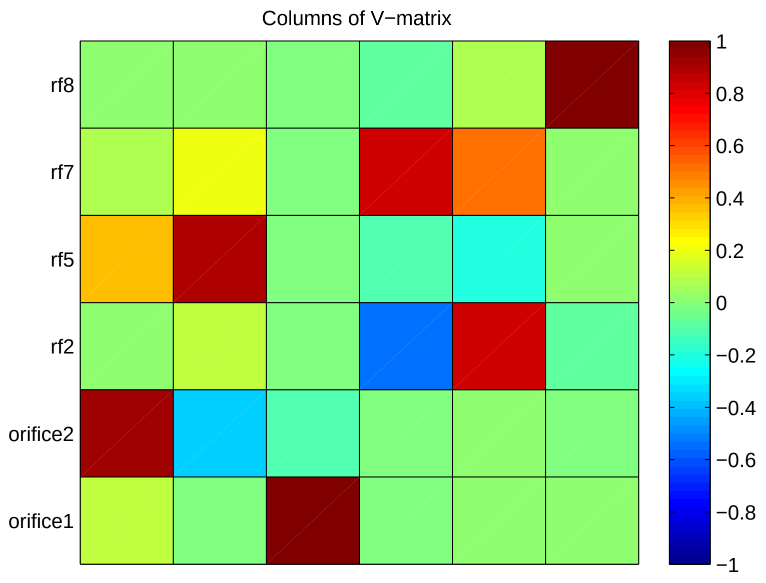


Figure D.4: Values of the V matrix for case 2.

Notice that, for the runoff parameters, in this case the singular values show a lot of similarity with the singular values in case 1. Also the identifiability of parameters is very similar. The big, and expected change, in results comes from the identification of the orifice parameters. These parameters are still very well identifiable on their own, however, the singular values for these column vectors are much larger. This is only natural since the observations include two points that are very near the orifices. As such, a change in these orifice parameters is very observable in the model output. What is a bit surprising, is that 'orifice2' is best identifiable and second best is 'rf5' while the extra observations are the nodes that surround 'orifice1', expecting better identification of this particular parameter.

Case 3

This is very similar as cases 1 and 2. Compared to case 2, two extra observation locations are included. These extra locations are the nodes that surround 'orifice2'. The singular values in this case are

$$\text{singular values}: [78.6443 \quad 12.4217 \quad 5.7563 \quad 1.2282 \quad 0.9743 \quad 0.3690]$$

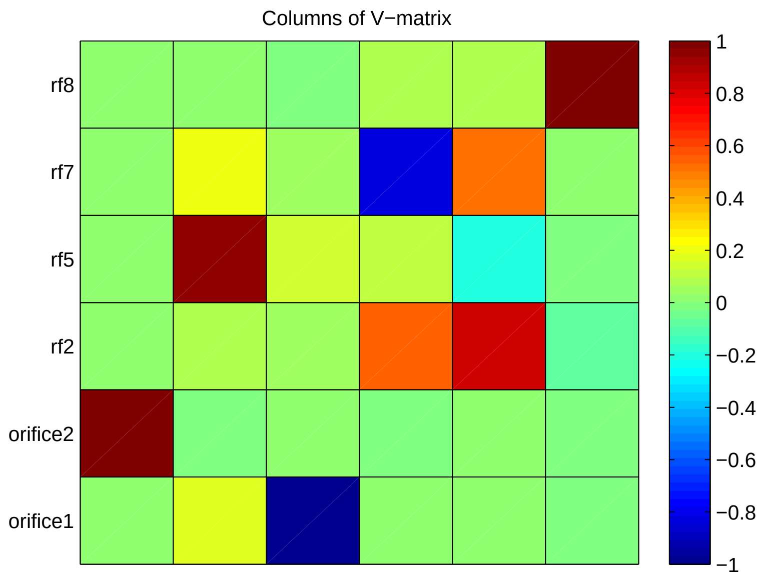


Figure D.5: Values of the V matrix for case 3.

Compared to the previous cases, no upsetting changes occur. In fact, the addition of these extra observations had no impact on the identification of the parameters. The change that did occur can be seen in the singular values. The addition of the locations surrounding 'orifice2' has greatly increased the identifiability of this particular parameter, as was somewhat expected.

Case 4

This is a special case to illustrate the influence of the chosen set of observation locations with respect to very local parameters. In this case, the set of observation locations is chosen very far from the orifice parameters with the expectation that the orifice parameters are still "identifiable" accoring to the V matrix but with very low, if not zero, valued singular values. Resulting in the conclusion that nothing useful can be said about the particular parameters. The results are in agreement with the expectations as can be seen from the singular values.

$$\text{singular values}: \; [48.1018 \quad 6.6079 \quad 2.4369 \quad 0.0960 \quad 0.0001 \quad 0]$$

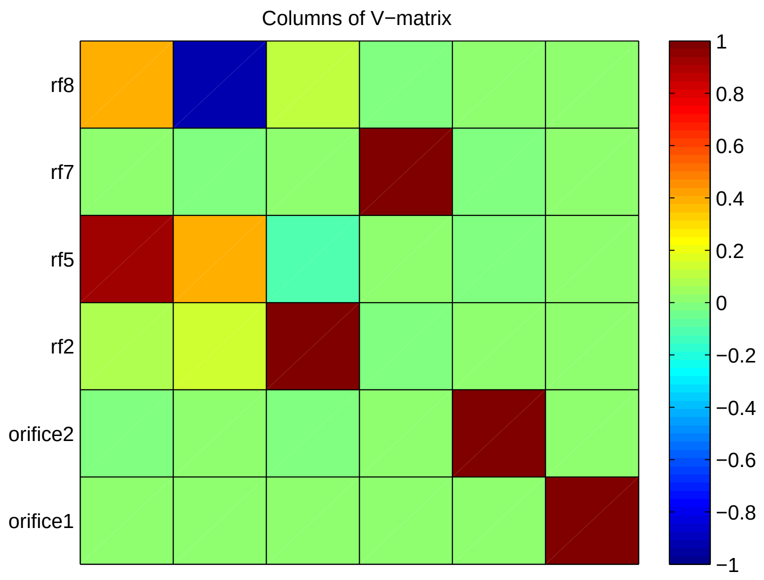


Figure D.6: Values of the V matrix for case 4.

The matrix V suggests perfect identifiability for the orifices, however, the singular values indicate that no useful information can be retrieved from the corresponding columns. The results of this last case actually show the importance of determining an optimal set of observation locations as was done by [Post, 2012] for example. It also provides some insight into the difficulty of determining exact locations of obstructions in a sewer system when every single reach is a possible candidate for these obstructions. Obviously there is still quite some work to be done in this regard.

Bibliography

[Barbu, 2010] Barbu, A. (2010). *Ensemble-based data assimilation schemes for atmospheric chemistry models.* PhD thesis, Delft University of Technology.

[Begum, 2009] Begum, N. (2009). Reservoir parameter estimation for reservoir simulation using ensemble kalman filter (enkf). Master's thesis, Norwegian University of Science and Technology.

[Bierman, 2006] Bierman, G. (2006). *Factorization Methods for Discrete Sequential Estimation.* Dover Books on Mathematics Series. Dover Publications. 1977 edition.

[Bijnen et al., 2012] Bijnen, M. v., Korving, H., and Clemens, F. (2012). Impact of sewer condition on urban flooding: an uncertainty analysis based on field observations and monte carlo simulations on full hydrodynamic models.

[Bishop and Hodyss, 2007] Bishop, C. and Hodyss, D. (2007). Flow-adaptive moderation of spurious ensemble correlations and its use in ensemble-based data assimilation. *Q.J.R. Meteorol. Soc.*, 133:2029–2044.

[Britta, 2000] Britta, P. (2000). *Calibration, identifiability and optimal experimental design of activated sludge models.* PhD thesis, Ghent Technology.

[Brown and Hwang, 1997] Brown, R. and Hwang, P. (1997). *Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions.* Wiley New York.

[Brusdal et al., 2003] Brusdal, K., Brankart, J., Halberstadt, G., Evensen, G., Brasseur, P., Leeuwen, P. v., Dombrowsky, E., and Verron, J. (2003). A demonstration of ensemble-based assimilation methods with a layered ogcm from the perspective of operational ocean forecasting systems. *Journal of marine systems*, 40-41:253–289.

[Buishand and Wijngaard, 2007] Buishand, A. and Wijngaard, J. (2007). Statistiek van extreme neerslag voor korte neerslagduren, tr-295, KNMI.

[Burgers et al., 1998] Burgers, G., Leeuwen, P. v., and Evensen, G. (1998). Analysis scheme in the ensemble kalman filter. *Monthly Weather Review*, 126:1719–1724.

[Clemens, 2001] Clemens, F. (2001). *Hydrodynamic models in urban drainage: application and calibration.* PhD thesis, Delft University of Technology.

[Cohn et al., 1998] Cohn, S., da Silva, A., Guo, J., Sienkiewicz, M., and Lamich, D. (1998). Assessing the effects of data selection with the dao physical-space statistical analysis system. *Monthly Weather Review*, 126:2913–2926.

[Cohn et al., 1994] Cohn, S., Sivakumaran, N., and R., T. (1994). A fixed-lag kalman smoother for retrospective data assimilation. *Monthly Weather Review*, 122:2838–2867.

[Deltares, 2013] Deltares (2013). Sobek user manual. version: 1.00.29255.

[Evans et al., 2002] Evans, N., Chapman, M., Chappell, M., and Godfrey, K. (2002). Identifiability of uncontrolled nonlinear rational systems. *Automatica*, pages 1799–1805.

[Evensen, 1994a] Evensen, G. (1994a). Inverse methods and data assimilation in nonlinear ocean models. *PHYSICA D*, 77:108–129.

[Evensen, 1994b] Evensen, G. (1994b). Sequential data assimilation with nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5):143–162.

[Evensen, 1997] Evensen, G. (1997). Advanced data assimilation for strongly nonlinear dynamics. *Monthly Weather Review*, 125:1342–1354.

[Evensen, 2003] Evensen, G. (2003). The ensemble kalman filter: theoretical formulation and practical implementation. *Ocean Dynamics*, 53:343–367.

[Evensen, 2004] Evensen, G. (2004). Sampling strategies and square root analysis schemes for the enkf. *Ocean dynamics*, 54:539–560.

[Evensen and Leeuwen, 2000] Evensen, G. and Leeuwen, P. v. (2000). An ensemble kalman smoother for nonlinear dynamics. *Monthly Weather Review*, 128:1852–1867.

[Gao et al., 2005] Gao, G., Zafari, M., and Reynolds, A. (2005). Quantifying uncertainty for the PUNQ-S3 problem in a Bayesian setting with RML and EnKF.

[Gillijns et al., 2006] Gillijns, S., Mendoza, B., Chandrasekar, J., de Moor, L., Bernstein, D., and Ridley, A. (2006). What is the ensemble kalman filter and how well does it work?

[Haugen et al., 2008] Haugen, V., Nævdal, G., Natvik, L., Evensen, G., Berg, A., and Flornes, K. (2008). History matching using the ensemble kalman filter on a north sea field case. *Society of Petroleum Engineers*, 13(4):382–391.

[Houtekamer and Mitchell, 1998] Houtekamer, P. and Mitchell, H. (1998). Data assimilation using an ensemble kalman filter technique. *Monthly Weather Review*, 126:796–811.

[Kalman, 1960] Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45.

[Kalnay, 2003] Kalnay, E. (2003). *Atmospheric modeling, data assimilation and predictability*. Cambridge.

[Kalnay et al., 2012] Kalnay, E., Ota, Y., Miyoshi, R., and Liu, J. (2012). A simpler formulation of forecast sensitivity to observations: application to ensemble kalman filters. *Tellus A*, 64:1–9.

[Korving, 2004] Korving, H. (2004). *Probabilistic assessment of the performance of combined sewer systems*. PhD thesis, Delft University of Technology.

[Lauvernet et al., 2009] Lauvernet, C., Brankart, J., Castruccio, F., Broquet, G., Brasseur, P., and Verron, J. (2009). A truncated gaussian filter for data assimilation with inequality constraints: Application to the hydrostatic stability condition in ocean models. *Ocean Modelling*, 27:1–17.

[Leeuwen and Evensen, 1996] Leeuwen, P. v. and Evensen, G. (1996). Data assimilation and inverse methods in terms of a probabilistic formulation. *Monthly Weather Review*, 124:2898–2913.

[Lin and Wang, 2013] Lin, C. and Wang, L. (2013). Forecasting simulations of indoor environment using data assimilation via an ensemble kalman filter. *Building and Environment*, 64:169–176.

[Lionello et al., 2006] Lionello, P., Sanna, A., Elvini, E., and Mufato, R. (2006). A data assimilation procedure for operational prediction of storm surge in the northern adriatic sea. *Continental shelf research*, 26:539–553.

[Lorentzen et al., 2008] Lorentzen, R., Shafieirad, A., and Nævdal, G. (2008). Closed loop reservoir management using the ensemble kalman filter and sequential quadratic programming.

[Maybeck, 1979] Maybeck, P. (1979). *Stochastic models, estimation and control*. Academic Press, New York.

[Morris, 1991] Morris, D. (1991). Factorial sampling plans for preliminary computational experiments. *Technometrics*, 33:161–174.

[Mulder, 2013] Mulder, D. (2013). Improving openda's blackbox wrapper for sobek. Delft University of Technology.

[Pelc et al., 2012] Pelc, J., Simon, E., Bertino, L., El Serafy, G., and A.W., H. (2012). Application of model reduced 4d-var to a 1d ecosystem model. *Ocean Modelling*, 57-58:43–58.

[Pham, 2001] Pham, D. (2001). Stochastic methods for sequential data assimilation in strongly nonlinear systems. *Monthly Weather Review*, 129:1194–1207.

[Pham et al., 1998] Pham, D., Verron, J., and Roubaud, M. (1998). A singular evolutive extended kalman filter for data assimilation in oceanography. *Journal of Marine Systems*, 16:323–340.

[Post, 2012] Post, J. (2012). Combining field observations and hydrodynamic models in urban drainage. Master's thesis, Delft University of Technology.

[Rabier et al., 1992] Rabier, F., Courtier, P., and Ehrendorfer, M. (1992). Four-dimensional data assimilation: comparison of variantional and sequential algorithms. *Quarterly Journal of the Royal Meteorological Society*, 118:673–713.

[Ralston and Jennrich, 1978] Ralston, M. L. and Jennrich, R. I. (1978). Dud, a derivative-free algorithm for nonlinear least squares. *Technometrics*, 20:7–14.

[Ravela and McLaughlin, 2007] Ravela, S. and McLaughlin, D. (2007). Fast ensemble smoothing. *Ocean dynamics*, 57:123–134.

[Sakov and Oke, 2008] Sakov, P. and Oke, P. (2008). A deterministic formulation of the ensemble kalman filter: an alternative to ensemble square root filters. *Tellus A*, 60:361–371.

[SimEnv, 2013] SimEnv (2013). Multi-run simulation environment simenv. user guide for version 3.1.

[Simon, 2001] Simon, D. (2001). Kalman filtering.

[Spekkers et al., 2011] Spekkers, M., Veldhuis, J. t., Kok, M., and Clemens, F. (2011). Analysis of pluvial flood damage based on data from insurance companies in the netherlands.

[Stanić et al., 2012] Stanić, N., Langeveld, J., and Clemens, F. (2012). Identification of the information needs for sewer asset management by assessing failure mechanisms.

[Stegeman, 2012] Stegeman, B. (2012). Model calibration as a tool to indentify sewer maintenance. Master's thesis, Delft University of Technology.

[Stolze, 2008] Stolze, J. (2008). Parameter identification in system biology. Master's thesis, Delft University of Technology.

[Stordal et al., 2011] Stordal, A., Karlsen, H., Nævdal, G., Skaug, H., and Vallés, B. (2011). Bridging the ensemble kalman filter and particle filters: the adaptive gaussian mixture filter. *Comput Geosci*, 15:293–305.

[Tippett et al., 2003] Tippett, M., Anderson, J., Bishop, C., Hamill, T., and Whitaker, J. (2003). Ensemble square root filters. *Monthly Weather Review*, 131:1485–1490.

[Veldhuis, 2011] Veldhuis, J. t. (2011). How the choice of flood damage metrics influences urban flood risk assessment. *J. Flood Risk Management*, 4:281–287.

[Veldhuis and Tait, 2011] Veldhuis, J. t. and Tait, S. (2011). Data-driven urban drainage analysis: an alternative to hydrodynamic models?

[Wall et al., 1981] Wall, J., Willsky, A., and Sandell, N. (1981). On the fixed-interval smoothing problem. *Stochastics*, 5:1–41.

[Wang et al., 2013] Wang, H., Sun, J., Zhang, X., Huang, X., and Auligné, T. (2013). Radar data assimilation with wrf 4d-var. part i: System development and preliminary testing. *Montly Weather Review*, 141:2224–2244.

[Whitaker and Compo, 2002] Whitaker, J. and Compo, G. (2002). An ensemble kalman smoother for reanalysis. In *Observation, data assimilation, and probabilistic prediction Symposium*. American Meteorological Society.

[Whitaker and Hamill, 2002] Whitaker, J. S. and Hamill, T. M. (2002). Ensemble data assimilation without perturbed observations. *Monthly Weather Review*, 130:1913–1924.

[Wilders and Heemink, 2011] Wilders, P. and Heemink, A. (2011). Environmental simulation and data assimilation. Lecture notes.

[Witteveen+Bos, 2012] Witteveen+Bos (2012). Slim meten en monitoren.

[Yaremchuk and Nechaev, 2013] Yaremchuk, M. and Nechaev, D. (2013). Covariance localization with the diffusion-based correlation models. *Monthly Weather Review*, 141:848–860.