Delft University of Technology

Data Assimilation
WI4204

Data Assimilation

Authors:
Julian Sanders (4675045)
Alvedian Mauditra Aulia Matin (5689252)

June 6, 2023

1 Question 1

Assignment:
Show that the linearized shallow water equations are equivalent to a wave equation, when the friction is neglected. Then compute (analytically) the propagation speed of the tidal wave. And finally substitute the values used in the model code and compare this to the propagation speed in the numerical model. How can you 'measure' the propagation speed in the numerical model?

Implementation:

The full shallow water equations in 1 dimension are:

$$\begin{aligned} \frac{\partial h}{\partial t} + \frac{\partial}{\partial x}((D+h)u) &= 0 \\ \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} &= -g\frac{\partial h}{\partial x} - fu + \nu\frac{\partial^2 u}{\partial x^2} \end{aligned} \tag{1}$$

In these equations, the following terms are...

- D : Water depth, from bottom to equilibrium level 20.0 [m] on average
- h : Wave height relative to equilibrium level [m]
- u : Velocity in the X direction [m/s]
- g : Graviational acceleration, 9.81 [m/s^2]
- f : Viscous drag coefficient [1/s]
- μ : Kinematic viscosity [m^2/s]
- x : Position in x direction [m]
- t : Time [s]

When we linearize these and neglect the kinematic viscosity we get the equations:

$$\begin{aligned} \frac{\partial h}{\partial t} + D\frac{\partial u}{\partial x} &= 0 \\ \frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} + fu+ &= 0 \end{aligned} \tag{2}$$

We are allowed to neglect the kinematic viscosity as we are well into the turbulent regime when modelling the tides in the Westerschelde.

If we neglect the friction term f, we get the system of equations:

$$\begin{aligned} \frac{\partial h}{\partial t} + D\frac{\partial u}{\partial x} &= 0 \\ \frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} &= 0 \end{aligned} \tag{3}$$

When we differentiate the first equation with respect to time, and the second equation with respect to x, we get:

$$\begin{aligned} \frac{\partial^2 h}{\partial t^2} &= -D\frac{\partial^2 u}{\partial t \partial x} \\ \frac{\partial^2 u}{\partial x \partial t} &= -g\frac{\partial^2 h}{\partial x^2} \end{aligned} \tag{4}$$

Assuming that the derivative of D over time and that of g over x are both zero.

We can combine these two equation to get something of the same form as a wave equation in one dimension for h:

$$\frac{\partial^2 h}{\partial t^2} = Dg\frac{\partial^2 h}{\partial x^2} \tag{5}$$

From this we see that the wave propagation speed is $c = \sqrt{Dg}$

If we substitute $D = 20.0$ [m] and $g = 9.81$ [m/s^2] into this expression, we get $c = 14.0$ [m/s]

We can measure the propagation velocity by observing how fast the waves travel from the western boundary to the eastern boundary in the simulation.

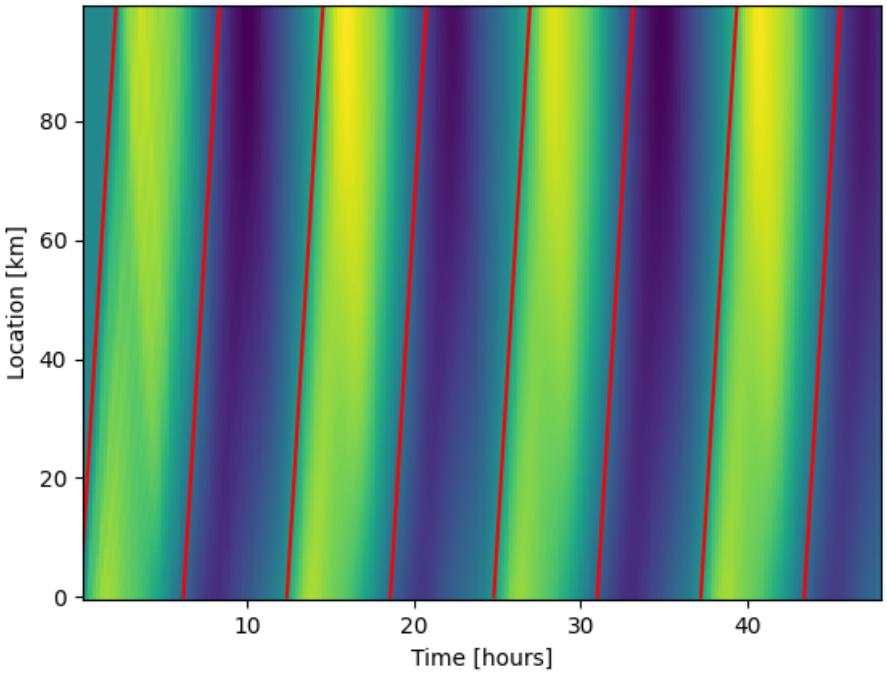


Figure 1: Characteristic lines (in red) plotted over the model solution plotted in 2D for space and time. From this we obtain that the propagation speed of the wave fronts is about 12.8 m/s

We can also indirectly measure the propagation velocity in the numerical model by finding the wavelength and period of the waves, and using the formula $c = \frac{\lambda}{T}$ where λ is the wavelength and T is the period of the oscillations.

The wavelength and the period can be found by manually inspecting the 2D solution plot of the model. From this we can easily see that $T = 12.4$ hours. This coincides with the period of the tides on earth, which is 12 hours and 25 minutes. Finding the wavelength is more complicated, as the bay is not large enough to contain one full wave.

To find the period of a full wave, we can modify the length of the wave in the simulation such that is is larger. (And we also increase the grid size accordingly). As this is a change to the domain of the wave equations, and not the parameters of the PDEs, this modification will not change the wave propagation velocity. By increasing the length from 100km to 700km, we can more easily read off the wavelength. From the solution in Figure 2, we can read off a wavelength of 575 km.

Then, using $T = 12.4$ hours and $\lambda = 575$ km, we find $c = 12.9$ m/s

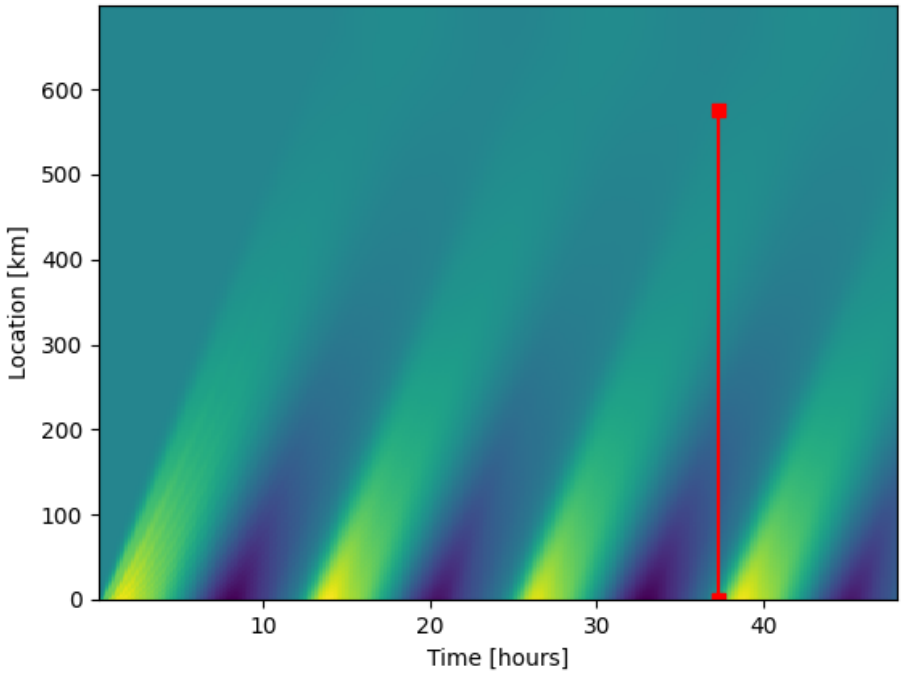


Figure 2: The solution of the model extended to a 700 km bay, with a manual measurement of a wavelength of 575 km (red).

The final (and most accurate) method for finding the propagation speed is to once again use the wavelength and the period to find the velocity. This time, the wavelength and period are obtained by a discrete Fourier transform of the solution of the model.

In order to make the discrete Fourier transform more accurate, we once again extend the length of the bay, this time to 24 000 km, or about 4 times the radius of the Earth. We can then perform a 2-dimensional DFT over the solution array to find the amplitudes of the different frequency components of the solution.

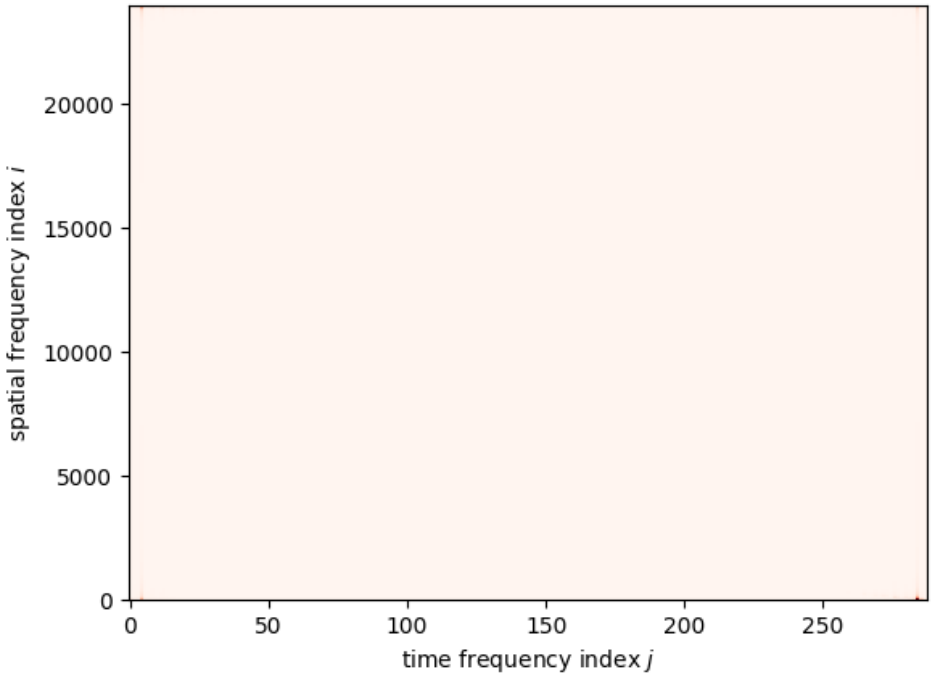
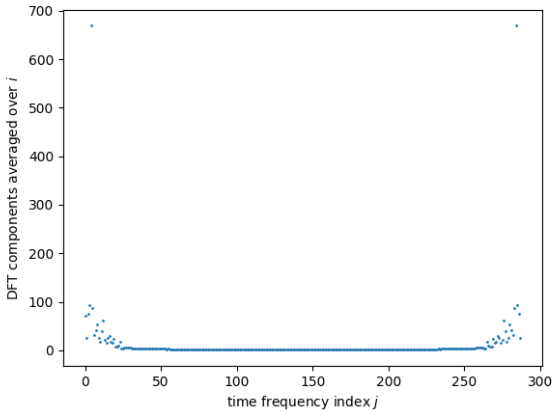
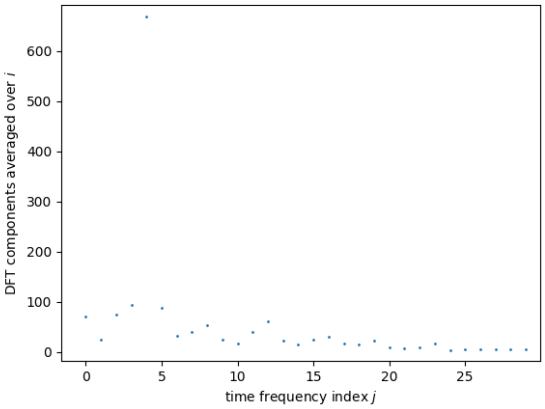


Figure 3: The 2 dimensional DFT of the solution for a bay of length 24000 km. Note the four small peaks in the corners of the figure, which correspond to the amplitudes of low frequency signals.

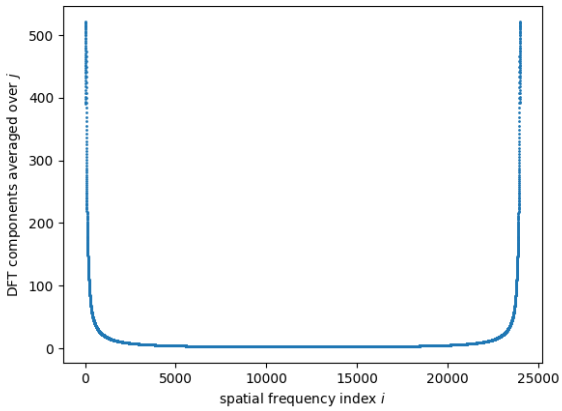


(a) The amplitudes of the time frequency components as a function of the frequency number j.

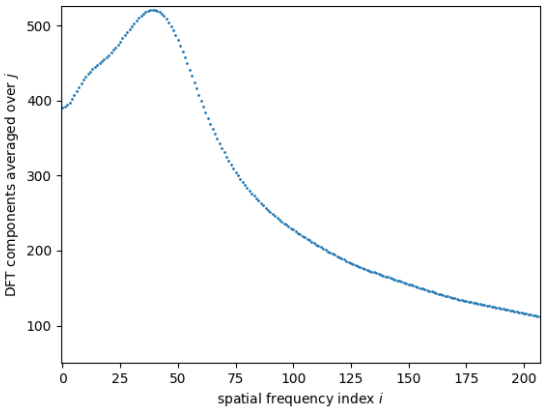


(b) A closeup of the amplitudes of the time frequency components. Note she sharp peak at $j = 4$.

Figure 4: The 2DFT amplitudes averaged over the index corresponding to space. This gives the amplitudes of the various time frequencies $\omega_t[j] = \frac{2\pi j}{MT_{tot}}$



(a) The spatial frequency amplitudes as a function of the frequency number j.



(b) A closeup of he spatial frequency amplitudes. Note the peak at $i = 39$.

Figure 5: The 2DFT amplitudes averaged over the index corresponding to time. This gives the amplitudes of the various spatial frequencies $\omega_x[i] = \frac{2\pi i}{NL_{tot}}$

In the above figures it can be observed that the predominant spatial frequency is that corresponding to the 39 ± 0.5 index, and that of the predominant time frequency is that of the 4 ± 0.5 index. These correspond to wavelength and period $\lambda = 615 \pm 16$ km and $T = 12.0 \pm 1.5$ hours. Since we already have a more accurate estimate of the period, as we know the period of the tides on earth, we shall use $T = 6.4$.

Using $\lambda = 615 \pm 16$ and $T = 6.4$ we obtain $c = 13.76 \pm 0.36$ m/s

This last estimate is by far the most accurate, as a manual measurement of the wavelength or the propagation speed is more prone to errors than an more exact method such as a DFT.

Note that for all three methods, the propagation speed of the model solution is less than the theoretical 14.0 m/s obtained when the friction term f is neglected. This is due to the fact that friction will decrease the propagation velocity.

2 Question 2

Assignment
Let's consider the accuracy of the model or the model error. This very simple model is definitely not perfect, so there should be room for improvement. A common approach to improve the output of a model in comparison to observations is to improve the model itself (physics, numerics, input data). Write down three aspects how you think this specific model could be improved.

Implementation

Improvements related to the physics of the model:

- Include a non-constant depth over the estuary
- Add a Coriolis force coefficient depending on position in the model
- Model the problem using two spatial dimensions

Mathematical improvements to the model:

- Model the problem including the nonlinear terms
- Model the problem with smaller time steps and a finer spatial discretization grid
- Model the problem with a higher order numerical integration method that the second order scheme that is used currently

Other improvements could be taking the gravitational acceleration as non-constant including the kinematic viscosity term, but we judge these to have a negligible effect on the numerical solution compared to the aforementioned factors.

3 Question 3

Assignment
As a reference, we want to see how well the model is doing without data-assimilation. We will first look at calm weather conditions, for this purpose we can apply 'tidal-analysis' as a filter to keep only the tidal frequencies. The model files provided contain a altered boundary condition and altered observations (with 'tide' in filename). Run the model and quantify the accuracy of the model for this case in terms of the bias and RMSE. This gives us a first quantitative idea about the accuracy of the model. Name a few alternative statistics that you could have used? Do these have benefits over bias and RMSE?

Implementation
The plots of $h_{num} - h_{data}$ are shown in Figure 6. We define the bias as

$$\frac{1}{N}\sum_{n=1}^{N}\left(E[h(n)] - h_{data}(n)\right) \tag{6}$$

with $h(n)$ being the numerically-computed value of h at the n-th time-step, $h_{data}(n)$ the observed value of h at the time corresponding to the n-th time step, and N being the total number of time steps in the simulation. The RMSE is defined as

$$\sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(h(n) - h_{data}(n)\right)^2}. \tag{7}$$

The bias and RMSE values are shown in Table 1. A few alternative metrics that could be used to assess the model are the infinity norm and the one norm:

- $\max_{n\leq N}|h(n) - h_{data}(n)|$
- $\frac{1}{N}\sum_{n=1}^{N}|h(n) - h_{data}(n)|$

The infinity norm offers the advantage that it is more sensitive to the highest deviation from the observed data, so this norm shows the maximum deviation. This has the advantage that the maximal deviation becomes immediately apparent from the norm value.

The one norm is the average of the absolute value of the difference between the model and the observation. This gives a more averaged picture of the deviation, as it is less sensitive to peak deviations than the infinity norm and the 2-norm.

Table 1: Bias and RMSE values for each location

Locations:	**Cadzand**	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.0	-0.02509191912	-0.1254636804	-0.16081320230	-0.2391504913
RMSE:	0.0	0.3796929936	0.5937107861	0.5926021925	0.4164716026
InfNorm:	0.0	0.6165739112	1.137642993	1.379958373	1.144116927
OneNorm:	0.0	0.3481288434	0.5334815168	0.5190146961	0.3674213087

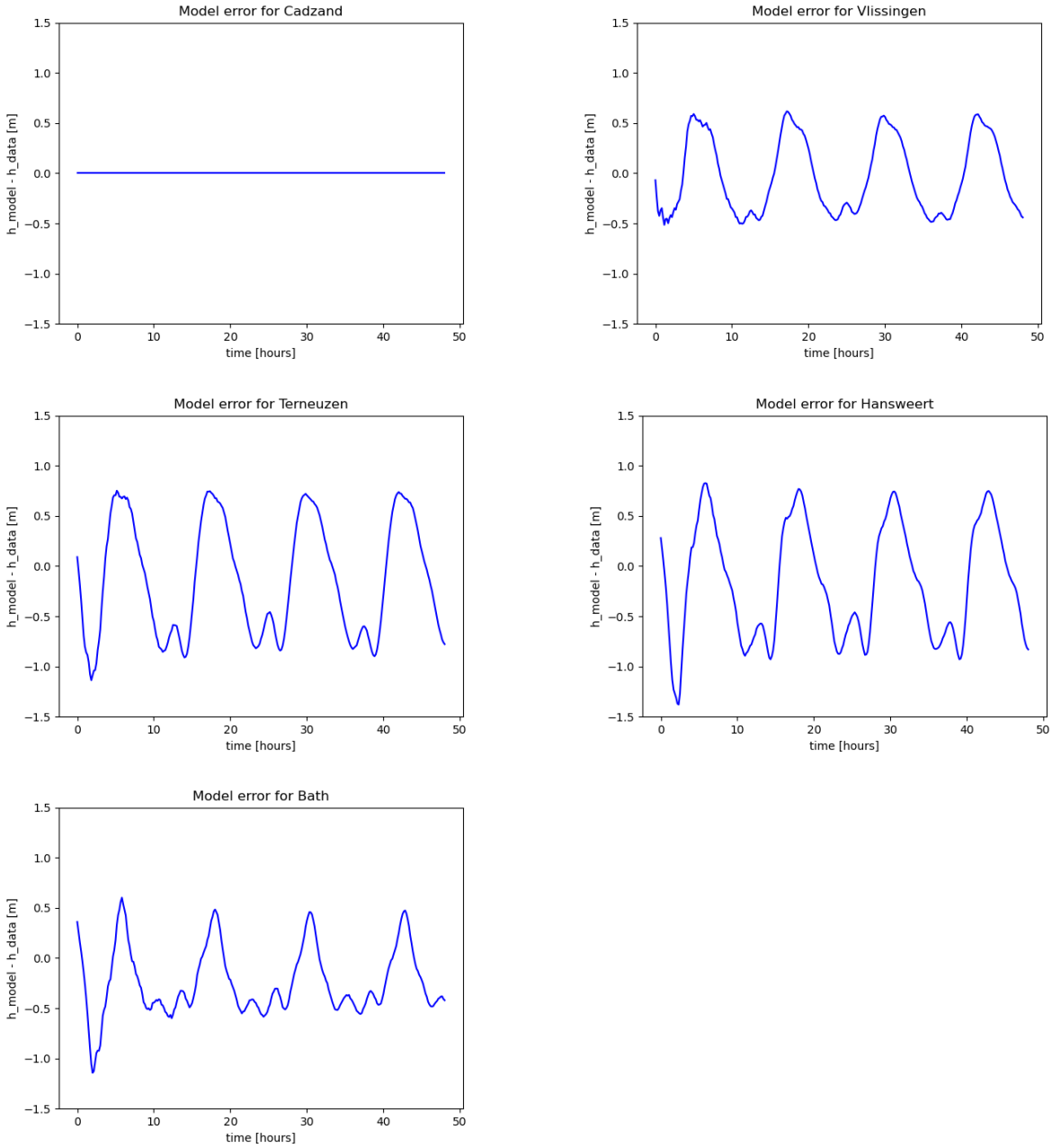


Figure 6: Model errors without stochastic forcing or data assimilation

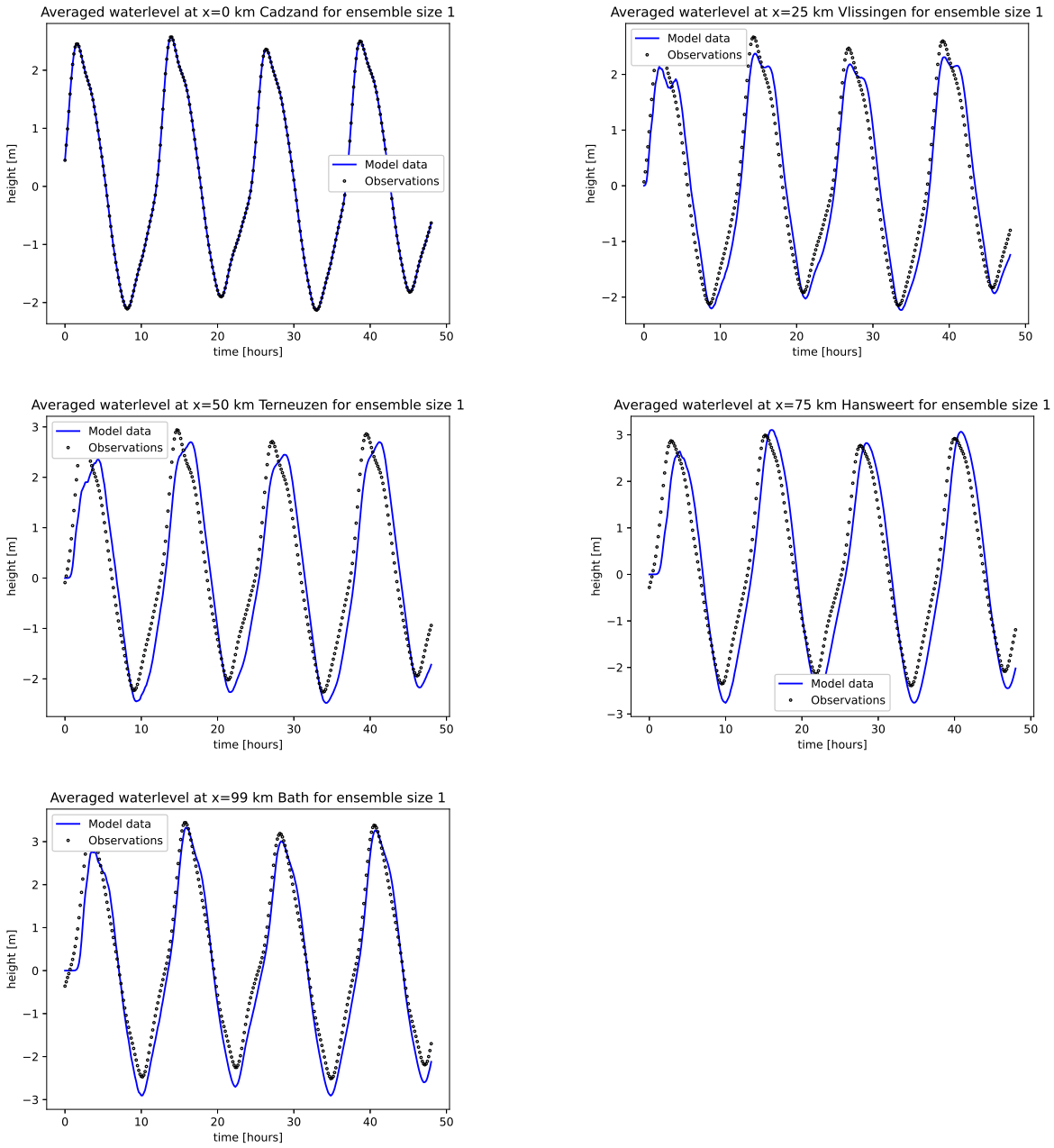


Figure 7: Plots of water level with no forcing for the model without stochastic forcing or data assimilation. The observations plotted are those using the 'tidal-analysis' filter

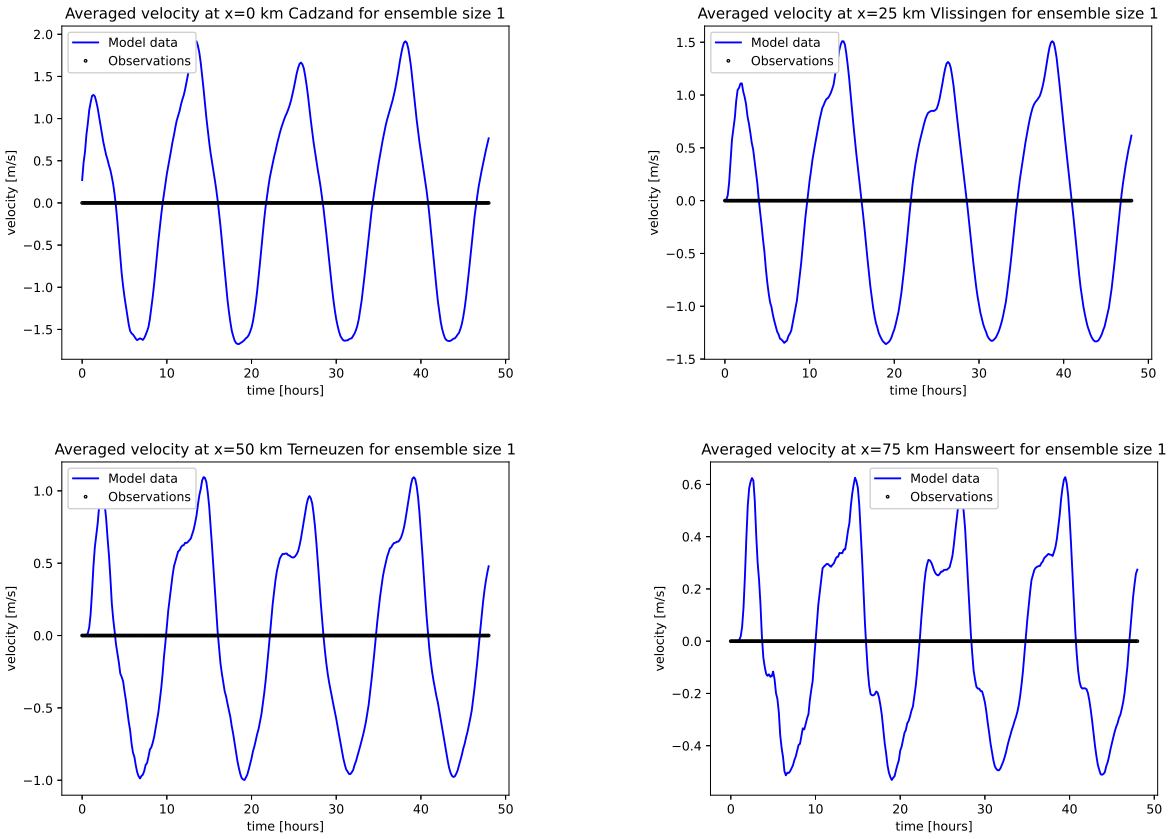


Figure 8: Plots of the horizontal velocity of the water with no forcing for the model without stochastic forcing or data assimilation. The observations plotted are those using the 'tidal-analysis' filter

4 Question 4

Assignment
Before applying an Ensemble Kalman Filter, we first need to model the uncertainty with a stochastic extension of the model. In this project, you'll add an AR(1) type stochastic forcing to the western boundary.

$$N(k+1) = \alpha N(k) + W(k) \tag{8}$$

Use $\alpha = \exp\left(-\frac{dt}{T}\right)$ with dt the timestep of the model and $T = 6[hours]$. How can one select the standard deviation σ_W for $W(K) \sim N(0, \sigma_W^2)$ so that $\sigma_N = \lim_{k\to\infty} \sqrt{E(N(k)^2)}$ equals $0.2[m]$? Implement this additional forcing term in the model, run for an ensemble of size 50 and plot some relevant results. Is the uncertainty as modelled with the ensemble comparable to your results of question 3? Compute some statistics (for the ensemble spread) and explain the results.

Implementation

Assuming $W(k)$ is normally-distributed white noise, we can write down the expectation of $N(k+1)^2$ as follows:

$$\begin{aligned} E[N(k+1)^2] &= E[(\alpha N(k) + W(k))^2] \\ &= \alpha^2 E[N(k)^2] + 2\alpha E[N(k)W(k)] + E[W(k)^2]. \end{aligned} \tag{9}$$

Since $N(k) = N(k-1) + W(k-1)$, $N(k)$ and $W(k)$ are uncorrelated and therefore $E[N(k)W(k)] = 0$. $W(k) \sim N(0, \sigma_W^2)$, so $E[W(k)^2] = \sigma_W^2$. Then

$$\begin{aligned} E[N(k+1)^2] &= \alpha^2 E[N(k)^2] + \sigma_W^2 \\ &= \alpha^{2k} E[N(0)^2] + \sigma_W^2 \sum_{j=0}^{k-1} \alpha^{2j} \\ &= \exp\left(-k\frac{dt}{3}\right) E[N(0)^2] + \sigma_W^2 \sum_{j=0}^{k-1} \exp\left(-j\frac{dt}{3}\right) \end{aligned} \tag{10}$$

and, for $E[N(0)^2] < \infty$,

$$\begin{aligned} \lim_{k\to\infty} \sqrt{E[N(k)^2]} &= \sqrt{\lim_{k\to\infty} \exp\left(-(k-1)\frac{dt}{3}\right) E[N(0)^2] + \sigma_W^2 \sum_{j=0}^{k-2} \left(\exp\left(-\frac{dt}{3}\right)\right)^j} \\ &= \sqrt{\frac{\sigma_W^2}{1 - \exp\left(-\frac{dt}{3}\right)}}. \end{aligned} \tag{11}$$

Selecting $\sigma_W = 0.2\sqrt{1 - \exp\left(-\frac{dt}{3}\right)}$ will result in $\lim_{k\to\infty} \sqrt{E[N(k)^2]} = 0.2$.

Forcing is implemented by generating noise using equation (8) and adding it to the western boundary condition. We can use the data from the westernmost location (Cadzand) Both are used to generate the results shown in Figure 9, which show the averaged results of running the model 50 times with forcing.

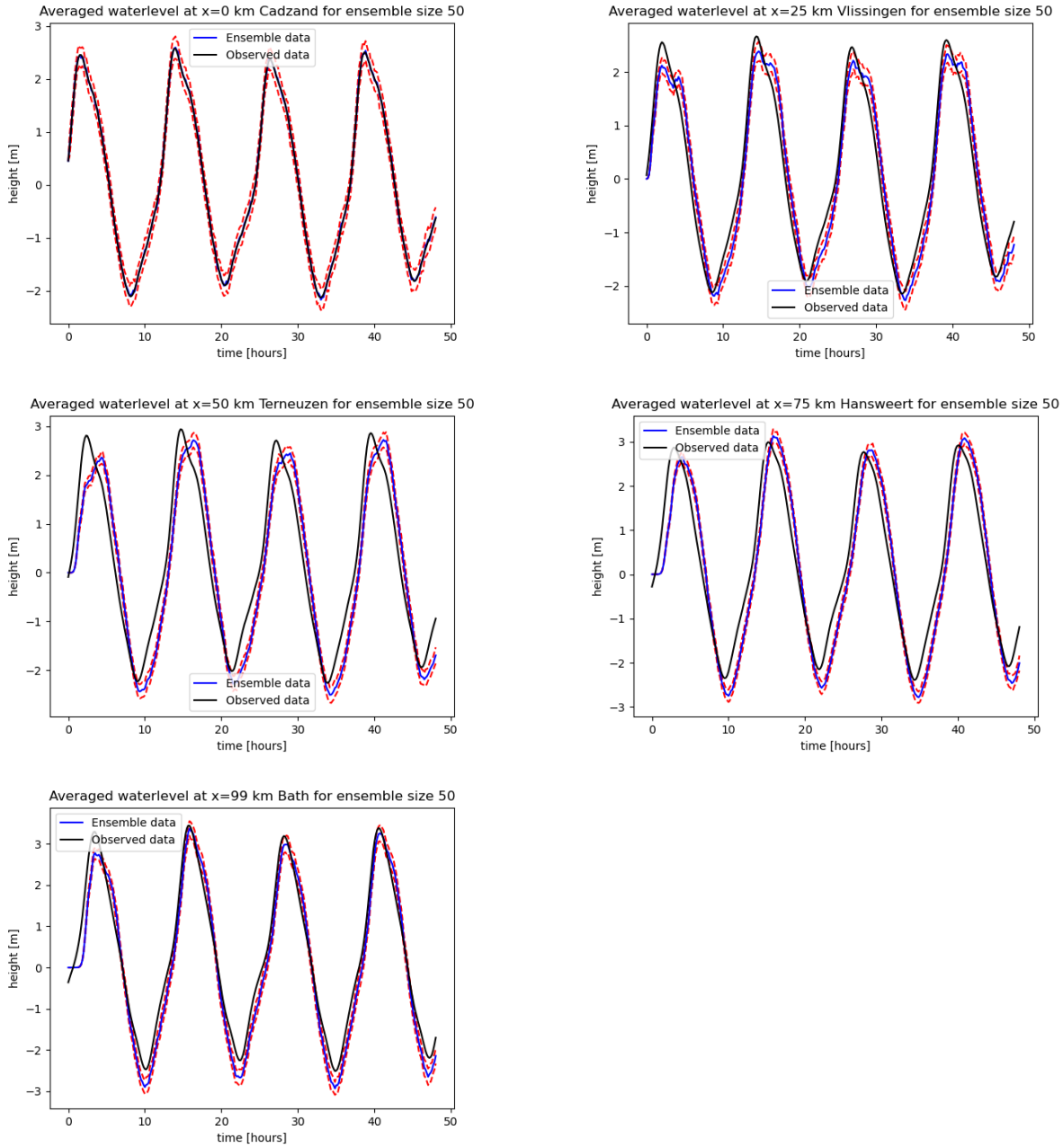


Figure 9: Plots of averaged water level with the standard deviation resulting from the with stochastic forcing and data from Cadzand as the western boundary condition

The RMSE and bias values are displayed in Table 2.

Table 2: Bias and RMSE values for each location with forcing and boundary values from data

Locations:	**Cadzand**	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.0092487708115	-0.01607570089	-0.1155342984	-0.15068897366	-0.2298915598
RMSE:	0.1965537522	0.3987456657	0.5926791912	0.5891492356	0.4367342395
InfNorm:	0.7127033321	0.8147135071	1.110492787	1.361067344	1.139326009
OneNorm:	0.1550691060	0.3484461940	0.5277881383	0.5151729525	0.3735012903

5 Question 5

Assignment
One key requirement for the use of the Kalman filter is the Markov property. Show how you fit the model in the standard system notation and explain that your implementation satisfies the Markov property.

Implementation
For each time-step k, the state $\mathbf{x}_k$ can be written as $\mathbf{x}_k = (x_{k,0}, x_{k,1}, \ldots, x_{k,2n-1})^T$. Let ξ_k be the western boundary condition at time-step k with u_k as the deterministic western boundary condition, $w_k \sim N(0, \sigma_w)$, and α a constant as in Question 4. Then $\xi_k = u_k + \alpha N_k + w_k$.

Each forward step of the model is given by the `timestep` function in the code with $\mathbf{A}$ and $\mathbf{B}$ being sparse matrices defined in `initialize`. One full forward run `timestep` is written out in Algorithm 1.

Algorithm 1 Forward model

Require: $\mathbf{A}, \mathbf{B}, \mathbf{x}_0, \mathbf{u}, \mathbf{w}$
  for $i = 0 \ldots n_t - 1$ **do**
    $x_{k,0} \leftarrow u_k + \alpha N_k + w_k$
    $\mathbf{x}_{k+1} \leftarrow \mathbf{A}^{-1}\mathbf{B}\mathbf{x}_k$
  end for

Let $M_{i,j}$ denote each element of $\mathbf{A}^{-1}\mathbf{B}$ with $0 \leq i, j \leq 2n-1$. Then

$$\begin{aligned}\mathbf{x}_{k+1} &= \begin{pmatrix} M_{0,0} & \cdots & M_{0,2n-1} \\ M_{1,0} & \cdots & M_{1,2n-1} \\ \vdots & \ddots & \vdots \\ M_{2n-1,0} & \cdots & M_{2n-1,2n-1} \end{pmatrix} \begin{pmatrix} \xi_k \\ x_{k,1} \\ \vdots \\ x_{k,2n-1} \end{pmatrix} \\ &= \begin{pmatrix} M_{0,0} & 0 & \cdots & 0 \\ M_{1,0} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ M_{2n-1,0} & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} u_k + \alpha N_k + w_k \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & M_{0,1} & \cdots & M_{0,2n-1} \\ 0 & M_{1,1} & \cdots & M_{1,2n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & M_{2n-1,1} & \cdots & M_{2n-1,2n-1} \end{pmatrix} \begin{pmatrix} x_{k,0} \\ x_{k,1} \\ \vdots \\ x_{k,2n-1} \end{pmatrix} \end{aligned} \tag{12}$$

We can 'augment' the state $\mathbf{x}_k$ by appending N_k to the state vector, so we can write $\mathbf{x}_k = (\xi_k, x_{k,1}, \ldots, x_{k,2n-1}, N_k)^T$. Then we have

$$\mathbf{x}_{k+1} = \begin{pmatrix} M_{0,0}(u_k + \alpha N_k + w_k) + M_{0,1}x_{k,1} + \ldots + M_{0,2n-1}x_{k,2n-1} \\ M_{1,0}(u_k + \alpha N_k + w_k) + M_{1,1}x_{k,1} + \ldots + M_{1,2n-1}x_{k,2n-1} \\ \vdots \\ M_{2n-1,0}(u_k + \alpha N_k + w_k) + M_{2n-1,1}x_{k,1} + \ldots + M_{2n-1,2n-1}x_{k,2n-1} \\ \alpha N_k + w_k \end{pmatrix} \tag{13}$$

$$= \mathbf{M} \begin{pmatrix} x_{k,0} \\ x_{k,1} \\ \vdots \\ x_{k,2n-1} \\ N_k \end{pmatrix} + \mathbf{B} \begin{pmatrix} u_k \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} + \mathbf{F} \begin{pmatrix} w_k \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \tag{14}$$

where

$$\mathbf{M} = \begin{pmatrix} 0 & M_{0,1} & \cdots & M_{0,2n-1} & \alpha M_{0,0} \\ 0 & M_{1,1} & \cdots & M_{1,2n-1} & \alpha M_{1,0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & M_{2n-1,1} & \cdots & M_{2n-1,2n-1} & \alpha M_{2n-1,0} \\ 0 & 0 & \cdots & 0 & \alpha \end{pmatrix}, \tag{15}$$

$$\mathbf{B} = \begin{pmatrix} M_{0,0} & 0 & \cdots & 0 \\ M_{1,0} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ M_{2n-1,0} & 0 & \cdots & 0 \\ 0 & \cdots & \cdots & 0 \end{pmatrix}, \tag{16}$$

$$\mathbf{F} = \begin{pmatrix} M_{0,0} & 0 & \cdots & 0 \\ M_{1,0} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ M_{2n-1,0} & 0 & \cdots & 0 \\ 1 & \cdots & \cdots & 0 \end{pmatrix} \tag{17}$$

Let

$$\mathbf{u}_k = \begin{pmatrix} u_k \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \mathbf{w}_k = \begin{pmatrix} w_k \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

then $\mathbf{x}_{k+1} = \mathbf{M}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k + \mathbf{F}\mathbf{w}_k$ in standard system notation. This means the state $\mathbf{x}_{k+1}$ at the time-step $k+1$ given $\mathbf{x}_k, \ldots, \mathbf{x}_0$ is the same as the state $\mathbf{x}_{k+1}$ at the time-step $k+1$ given $\mathbf{x}_k$ only, since the deterministic boundary condition value and added noise u_{k+1} and w_{k+1} do not depend on the previous time-steps. Therefore $\mathbb{P}(\mathbf{x}_{k+1}|\mathbf{x}_k, \ldots, \mathbf{x}_0) = \mathbb{P}(\mathbf{x}_{k+1}|\mathbf{x}_k)$ and so this implementation satisfies the Markov property.

Algorithm 2 Simulation with EnKF

Require: $\mathbf{M}, \mathbf{A}, \mathbf{B}, \mathbf{u}, \mathbf{w}, \mathbf{H}, \mathbf{R}, \mathbf{x}_{j,0}, \mathbf{u}_{j,:}, \mathbf{w}_{j,:}, \mathbf{v}_{j,:}$
  for $k = 0 \dots n_t - 1$ **do**
    for $j = 1 \dots N$ **do**
      $\mathbf{x}_{j,k+1} \leftarrow$ forward timestep from $\mathbf{x}_{j,k}, \mathbf{A}, \mathbf{B}, \mathbf{u}_j, \mathbf{w}_j$ using Algorithm 1
    end for
    $\mathbf{C} \leftarrow$ sample covariance of $(\mathbf{x}_{1,k+1}, \dots, \mathbf{x}_{N,k+1})$
    $\mathbf{K} \leftarrow$ Kalman gain matrix $\mathbf{C}\mathbf{H}^T(\mathbf{H}\mathbf{C}\mathbf{H}^T + \mathbf{R})^{-1}$
    for $j = 1 \dots N$ **do**
      $\mathbf{x}_{j,k+1} \leftarrow \mathbf{x}_{j,k+1} + \mathbf{K}(\mathbf{z}_{k+1} + \mathbf{v}_{j,k+1} - \mathbf{H}\mathbf{x}_{j,k+1})$
    end for
    $\mathbf{x}_{a,k+1} \leftarrow$ mean of $(\mathbf{x}_{1,k+1}, \dots, \mathbf{x}_{N,k+1})$
  end for

6 Question 6

Assignment
Implement an Ensemble Kalman filter and set up an identical twin experiment to test the implementation. Check some relevant statistics to verify your implementation. Why should the EnKF work perfectly (for a large ensemble) in this experiment? How can you verify this for your output?

Implementation
Let $2n$ be the dimension of the state space (using the notation in the code). Algorithm 2 gives the pseudo-code implementation of the ensemble Kalman filter for a size N ensemble [1], [2], $\mathbf{A}$ and $\mathbf{B}$ being the sparse matrices defined in the `initialize`. Additionally, $\mathbf{u}_j$ is the deterministic boundary condition, $\mathbf{w}_j$ is the noise vector discussed in Question 4, $\mathbf{R}$ is the observation error covariance matrix, and the observation operator $\mathbf{H}$ is defined as a $5 \times 2n$ matrix with ones at the indices corresponding with water level measurements at the five locations (Cadzand, Vlissingen, Terneuzen, Hansweert, and Bath) and zero everywhere else. Forcing at the boundary is implemented as explained in Question 4 and notation from Question 5 is used for the forward model. In the filtering step, $\mathbf{v}_{j,k+1} \sim N(0, \sigma_R \mathbf{I})$ is the same size as the observations.

The ensemble Kalman filter should work well for this model, as it is a linear model. Each timestep is computed linearly using matrix vector multiplications. Even though the ensemble Kalman filter can also work for nonlinear models, the fact that this model is linear ensures that the ensemble Kalman filter will work well given sufficiently large ensemble sizes.

The reason behind this perfect performance is that linear models can be fully characterized by the mean and covariance of the state variables. Furthermore, the filter preserves the linearity of the model even after the assimilation of observations. The Kalman gain $\mathbf{K}$, which combines the model and observation errors, is calculated using matrix operations linear in $\mathbf{x}$. As a result, the EnKF updates the ensemble members in a way that maintains the linearity of the model, ensuring accurate assimilation of observations.

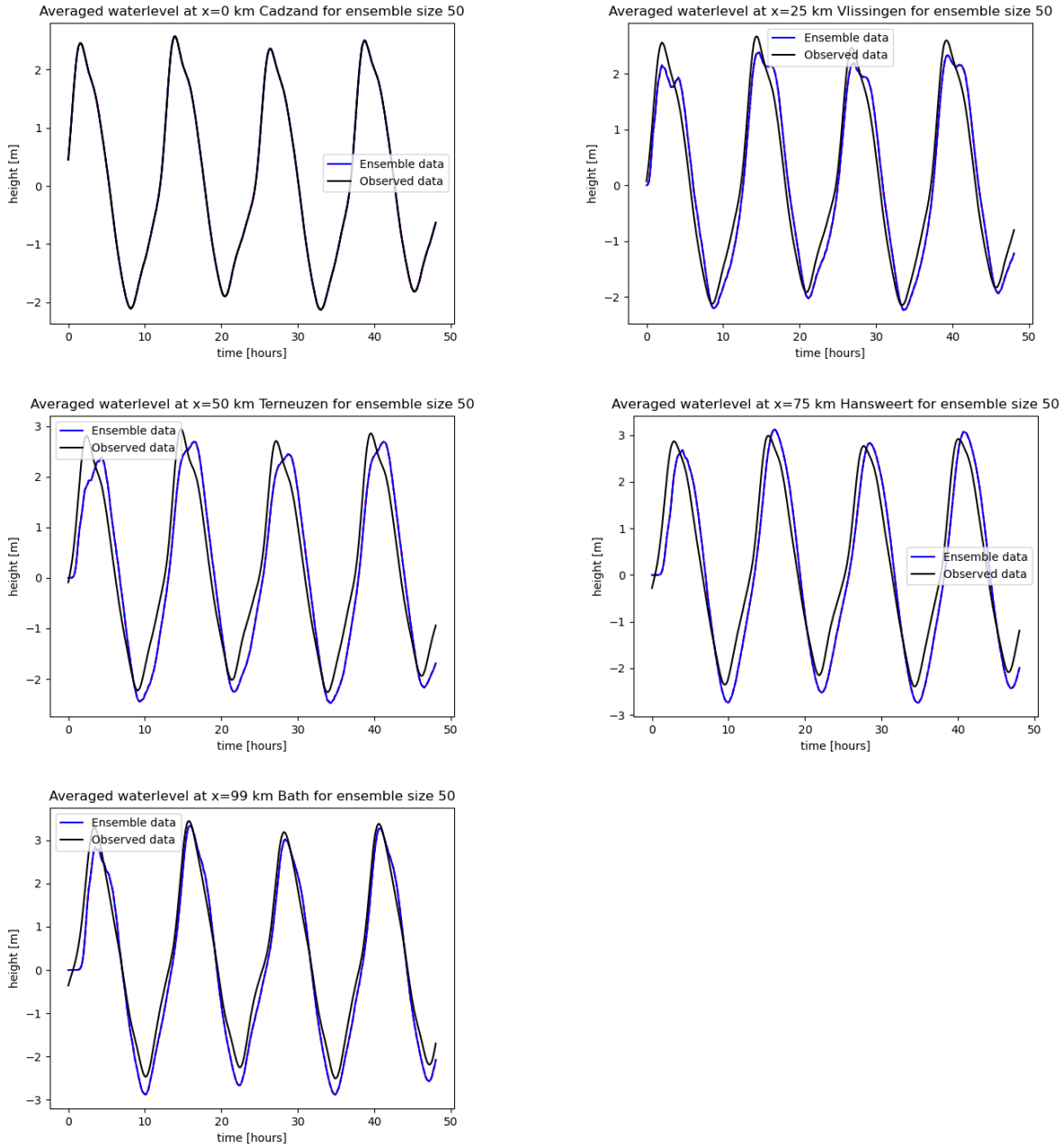


Figure 10: Plots of water level obtained using EnKF for $N = 50$, $R = 0.01\mathbf{I}$. The mean and standard deviation are plotted of the model water level. Note that the standard deviation is so small that these plots overlap with the mean waterlevel plot. For the observed data the 'tidal-analysis' filter was used.

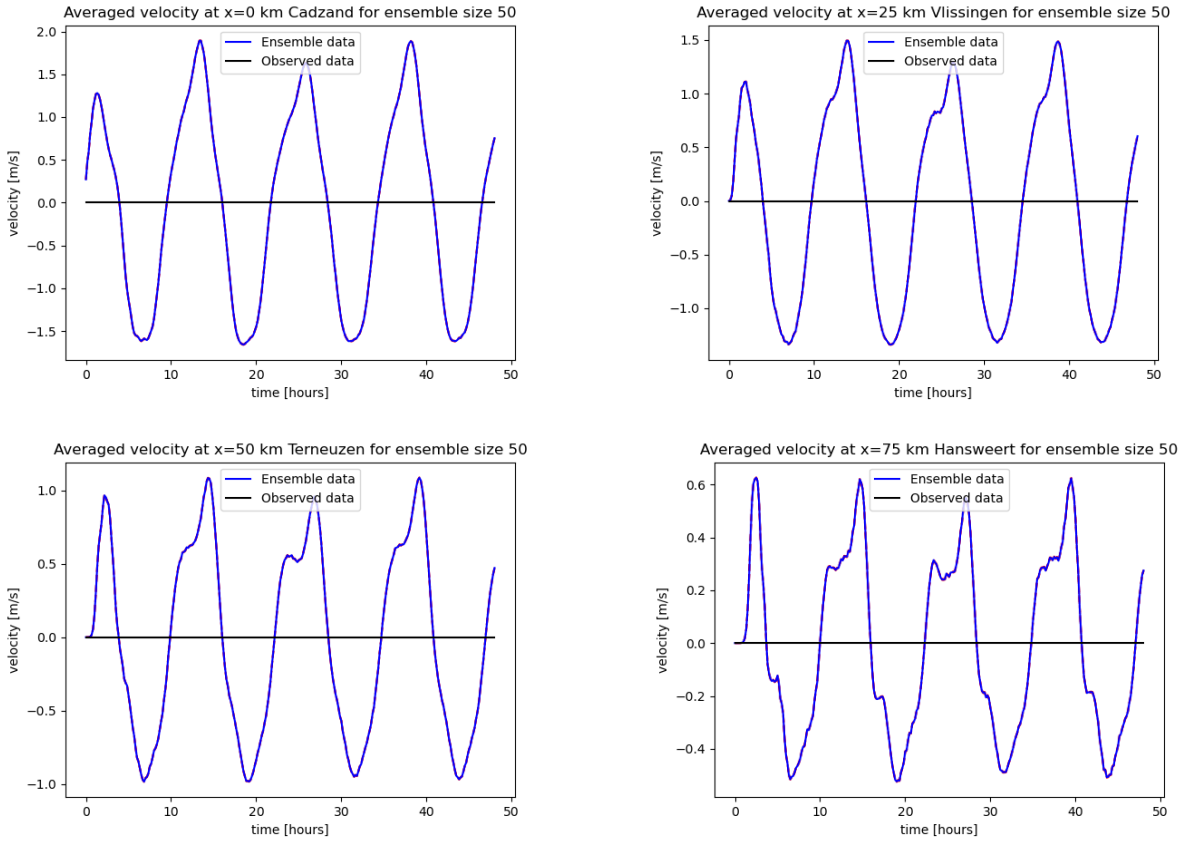


Figure 11: Plots of water velocity obtained using EnKF for $N = 50$, $R = 0.01\mathbf{I}$. The mean and standard deviation are plotted of the model water level. Note that the standard deviation is so small that these plots overlap with the mean waterlevel plot. For the observed data the 'tidal-analysis' filter was used

Table 3: Bias and RMSE values for each location with ensemble Kalman filter implemented

Locations:	**Cadzand**	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	-0.0008694098875	-0.01635282039	-0.09040443101	-0.10935029123	-0.17230786325
RMSE:	0.01824506683	0.25090000234	0.36878208704	0.3402624447	0.2767442931
InfNorm:	0.06278631087	0.6173962167	0.8778024082194542	1.064672682	0.9855536255
OneNorm:	0.01455066436	0.2166452705	0.3170359266	0.28336070733	0.2124066632

The identical twin experiment is carried out using the following steps:

1. Run the forward model with stochastic forcing at the western boundary once.
2. Store the results (the full set of states $\mathbf{x}_k$ for $k = 0, \ldots, n_t$) generated in Step 1.
3. Run a simulation with an EnKF and ensemble size N, using the generated 'data' obtained from the previous as the measurements $\mathbf{z}_k$ in the filtering step.

Additionally, the EnKF works perfectly because the twin 'observations' used in the experiment was generated using the model. As a result, the ensemble members only deviate from the observations due to added noise. Thus, the average of the ensemble members converges to the twin 'observations' as the number of ensemble members increases.

	Step 1 bias (m)	Step 1 RMSE (m)	Step 3 bias (m)	Step 3 RMSE (m)
Cadzand	9.2488×10^{-3}	1.9655×10^{-1}	5.4885×10^{-4}	6.0058×10^{-3}
Vlissingen	-1.6076×10^{-2}	3.9875×10^{-1}	6.0396×10^{-4}	4.8542×10^{-3}
Terneuzen	-1.1553×10^{-1}	5.9268×10^{-1}	5.8554×10^{-4}	4.0678×10^{-3}
Hansweert	-1.5069×10^{-1}	5.8915×10^{-1}	6.0553×10^{-4}	3.2885×10^{-3}
Bath	-2.2989×10^{-1}	4.3673×10^{-1}	6.0101×10^{-4}	4.5427×10^{-3}

Table 4: Bias and RMSE values for the generated (twin) data compared to the observations given in the `tide` files (columns 1 and 2) and for the 50-member EnKF results compared to the generated (twin) data (columns 3 and 4)

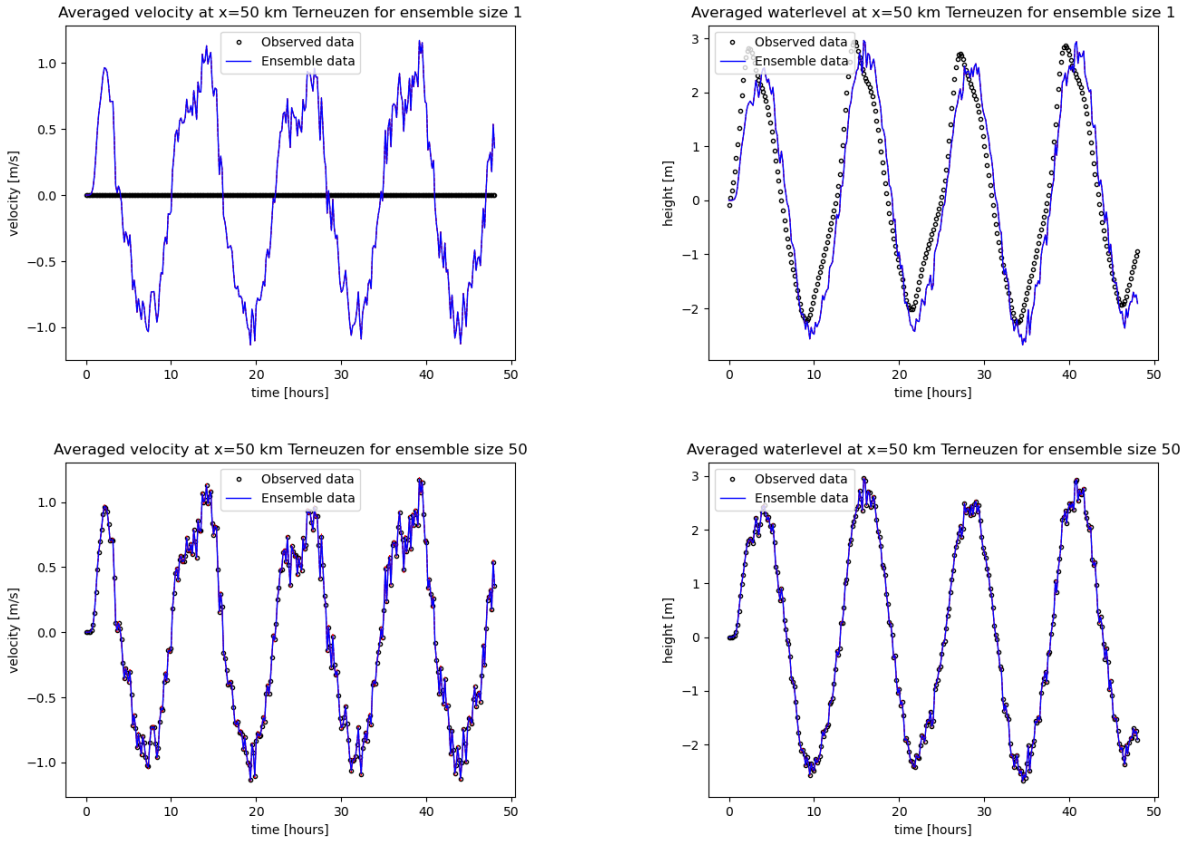


Figure 12: Virtual observations for the twin experiment plotted against `tide` observations (top row) and 50-ensemble EnKF results with virtual observations used in the filtering step (bottom row)

By plotting the estuary states at different times together with the observed data that is used in the Kalman filter, we see that the model conforms to the data less than expected. See figure 13.

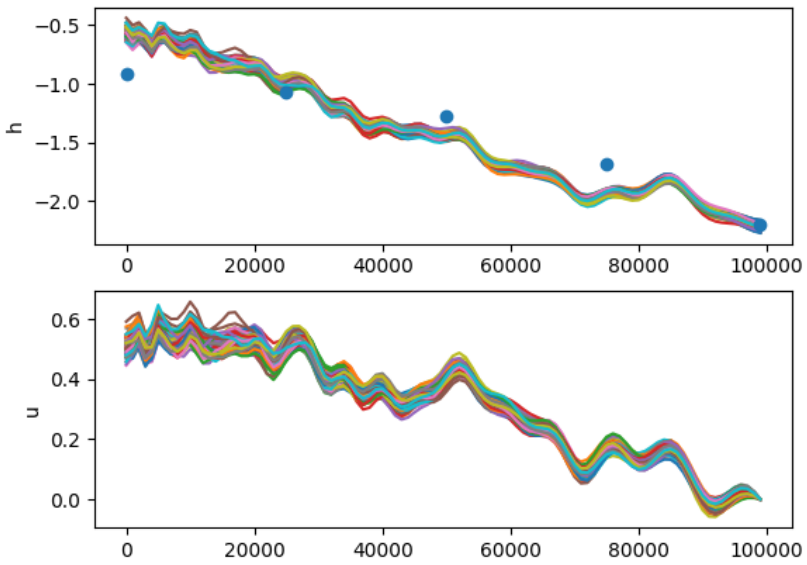


Figure 13: A plot of all the water level and velocity of the ensemble states and the observations used for assimilation. Note that the ensemble states are relatively close, but not the same as the observation data used in the enseble Kalman filter (blue dots).

This behavior can be explained through the fact that the correlation between the water level at one position in the bay and the other position in the bay is quite high. Therefore, it is not possible for the ensemble Kalman filter to adapt the model in such a way that is fits through each observation value.

If we would modify the stochastic noise forcing at the west boundary in such a way that it is less correlated from time to time, the fitting to the observed data would be much closer.

Nonetheless, from the RMSE and Bias values, we do see that the model with ensemble Kalman Filtering performs better than the one without this filter. Note that a perfect fit to the data is not desired, as the observations are also subject to noise ($\sigma_R = 0.1$, as $\mathbf{R} = 0.01\mathbf{I}_5$).

7 Question 7

Assignment
The Ensemble Kalman filter is supposed to get more accurate with a larger number of ensemble members. To what solution should the EnKF converge for a large ensemble in this case? Why? What is the expected convergence rate? How can we verify this? Perform an experiment to check this.
Implementation

When the number of ensemble members in the EnKF is large, the filter is expected to converge to a more accurate solution. Thus, we expect the model solution to lie closer to the observed data. The reason behind this improvement is that a larger ensemble provides a better representation of the uncertainty in the system, allowing for a more robust estimation of the state variables.

The (non-ensemble) Kalman filter analysis mean and covariance are

$$\bar{x}_{k,a} = \bar{x}_{k,f} + K(d - H\bar{x}_f) \tag{18}$$
$$C_{k,a} = C_{k,f} - C_{k,f}H^T(HC_{k,f}H^T + R)^{-1}HC_{k,f} \tag{19}$$

where $\bar{x}_{k,f}$ and $C_{k,f}$ denote the forecast state mean and covariance at time-step k, H denotes the observation operator, and R is the observation error covariance. At every time-step k, the N-ensemble mean of the filtered state ($\bar{\mathbf{x}}^N_{k,a}$) and the N-ensemble covariance of the filtered state ($\mathbf{C}^N_{k,a}$) respectively converge to $\bar{x}_{k,a}$ and $C_{k,a}$ at a rate of $N^{-0.5}$ [3].

Five simulation runs are performed with different random number generator seeds and the ensemble Kalman filter with increasing ensemble sizes. The RMSE values from the different runs are plotted in Figure 14. A curve is then fitted to the mean value of the RMSEs. We obtain

$$RMSE_{mean} = 0.216 + N^{-0.864}. \tag{20}$$

.

Since $\lim_{N\to\infty} \frac{N^{-0.864}}{N^{-0.5}} = 0$ we can say that the convergence rate of our implementation is as expected (or even faster, if Eq. (20) holds).

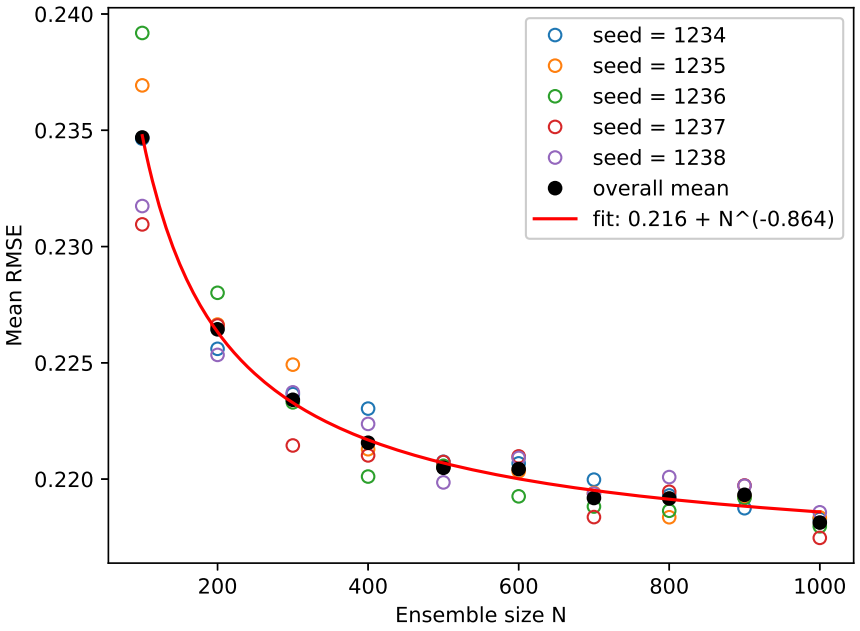


Figure 14: Mean of RMSE values at the five locations for 5 simulation runs with ensemble sizes $N = 100, \ldots, 1000$

8 Question 8

Assignment
For your answer to question 6, you selected an initial condition for the Kalman filter. What choice did you make there? Describe an alternative choice for the initial condition. Also perform an experiment with this alternative choice and study the impact on the results. Can you explain what happens, using the theory?

Implementation

In question 6 we assumed the initial condition to be a waterlevel of 0 m and a vertical velocity of 0 m/s everywhere. This can be seen in figure 15.

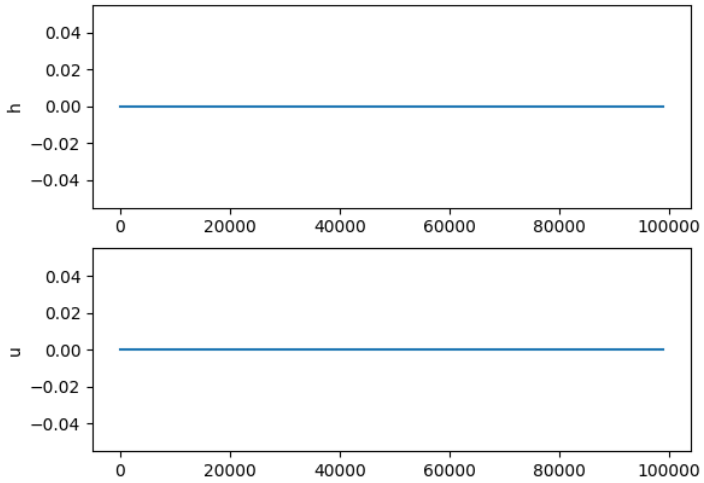


Figure 15: The zero initial conditions of the model, as used in question 6 and 7

This is of course not a realistic assumption for modelling the tides in a bay, in reality there are already tidal waves in the estuary. It is not perfectly flat everywhere.

Due to the fact that we start with a zero solution everywhere, we see that the waterlevel deviates quite a bit for the first few hours from the observed values for the measurement locations further into the estuary; here the wave from the forcing from the left boundary condition takes a few hours to arrive.

A more realistic initial model condition would be the waterlevel and the vertical velocity corresponding to a tidal wave with the same phase as the phase of the forcing of the left boundary.

To achieve this, we first ran the model with the zero intial conditions (using the 'tide' filter, with ensemble Kalman filter data assimilation, $n = 100$). We then selected the water level and vertical velocity after 37.25 hours as the initial conditions. 37 hours and 15 minutes is exatly three times the tidal phase (12 hours and 25 minutes), so this should give a reasonable estimate for the waterlevel and vertical velocity at t=0.

The resulting initial condition values can be seen in figure 15. Using these intial conditions, we can run the model with ensemble Kalman filter again ($n = 100$). We see that in this case, the model is more representative for the measuring stations that lie to the east of the estuary for the first hours, as can be seen when comparing figure 17 to figure 10.

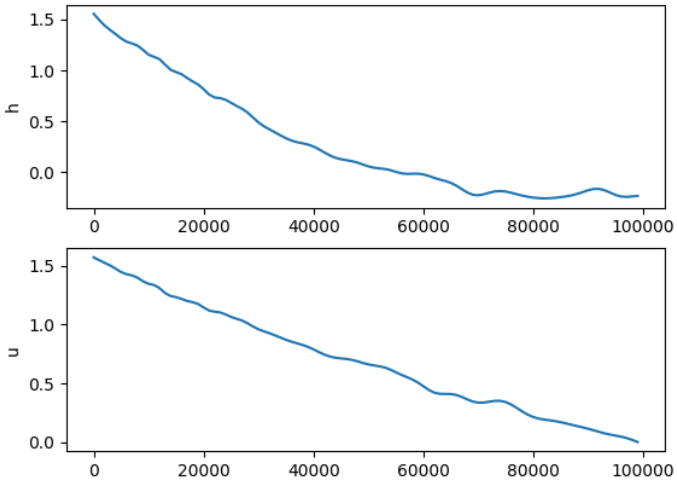


Figure 16: The initial conditions of the model, sampled from the model with zero initial conditons at 37 hours and 15 minutes.

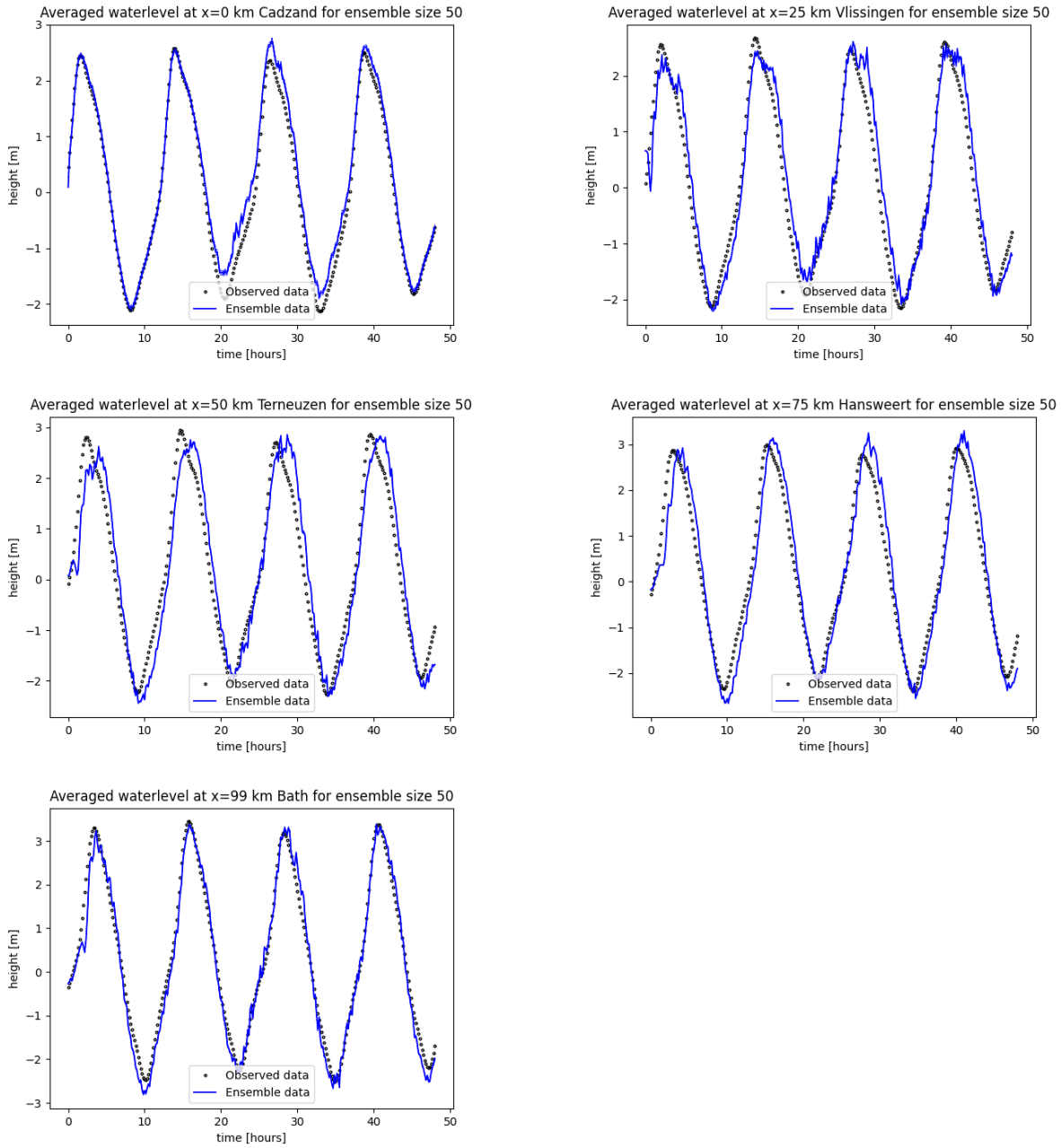


Figure 17: The waterlevel solution at the different observation point and the observed data at that point. This model was run with initial conditions obtained from the previous model solution at 37 hours and 15 minutes. Note that the solutions for Terneuzen, Hansweert and Bath have become more accurate for the first few hours, compared to figure 10. The standard deviations are plotted in blue as well, but are not clearly visible due the them being relatively small.

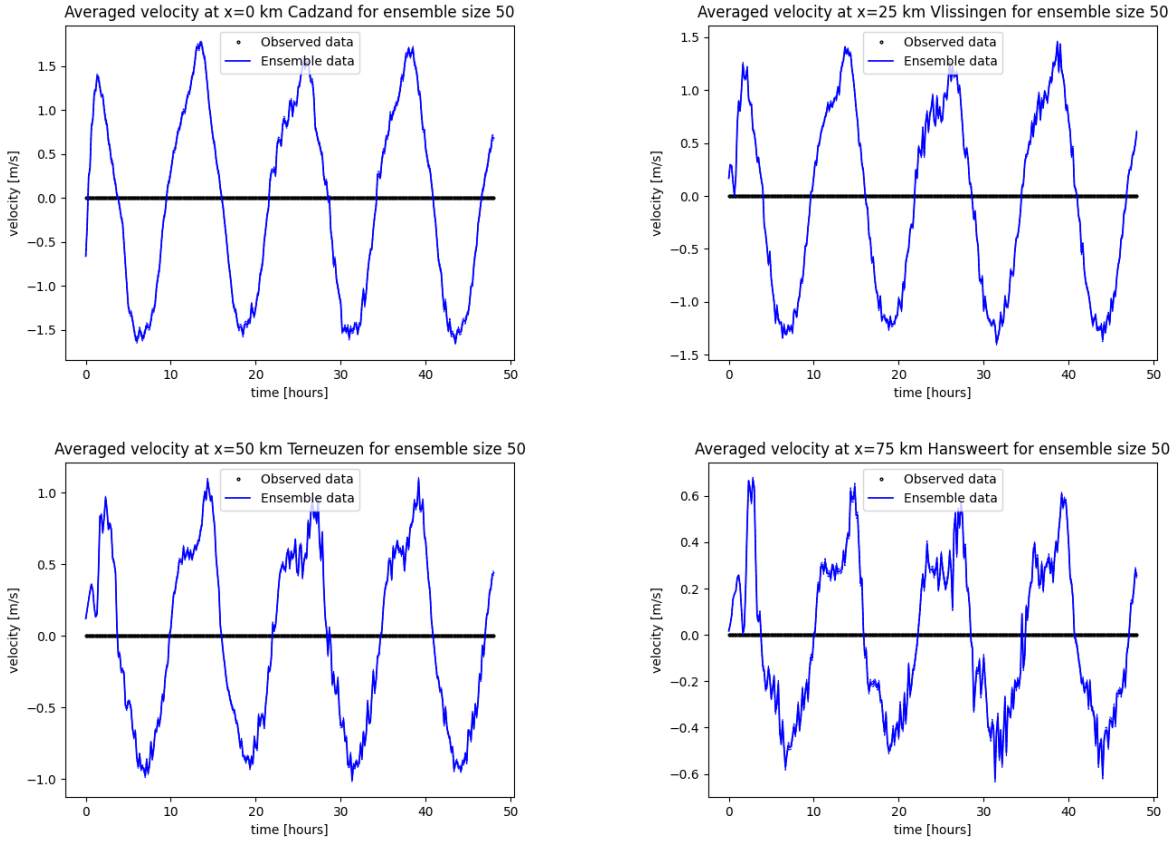


Figure 18: The vertical velocity model solution at the different observation point and the observed data at that point. This model was run with initial conditions obtained from the previous model solution at 37 hours and 15 minutes.

This increase in accuracy also becomes apparent from the measurement norms, as can be seen in the tables 5 and 6.

Table 5: The different error metrics comparing the model solution for an ensemble size 50 to the observed values. Here the zero initial condition was used, see figure 15

Locations:	**Cadzand**	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.1631200165	0.1195663687	0.01792567702	-0.01901097876	-0.09424939884
RMSE:	0.2348891068	0.4018519994	0.5291003354	0.4957720458	0.3114989236
InfNorm:	0.6095411284	1.051415158	1.150344393	1.291497356	1.211746779
OneNorm:	0.1870193223	0.3404989526	0.4590593582	0.4110073633	0.2452722861

Table 6: The different error metrics comparing the model solution for an ensemble size 50 to the observed values. Here the initial conditions were sampled from another model run, see figure 16. The relative improvement of the error metric compared to the zero initial condition case (table 5) is color coded, red for worse performance, green for improvement. The model now tends to underestimate wave height at the observation points farther east.

Locations:	**Cadzand**	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.002218046372	-0.006337223622	-0.078984484109883	-0.097979362993160	-0.1593923417
RMSE:	0.01284477866	0.2365144032	0.3423962082	0.3100978052	0.2423133792
InfNorm:	0.04502148807	0.6987499101	1.002620779	1.023163695	0.7777576883
OneNorm:	0.01023107245	0.2034616505	0.2893812767	0.2568221882	0.1876496648

9 Question 9

Assignment
Now, we move on to the real observation with the storm effects included (files with 'waterlevel' in the name). Note that the meassurements at the boundary (Cadzand) are missing. Modify your implementation to assimilate these observations. Make some plots to study the results. How accurate is the result in comparison to running the model without a Kalman filter. Quantify this with some statistics. Comment on your results, also in relation to the theory.

Implementation
In the filtering step, the matrix $\mathbf{H}$ (in Algorithm 2) is modified into a $4 \times 2n$ matrix with ones at indices associated with waterlevel measurements at Vlissingen, Terneuzen, Hansweert, and Bath and zeros everywhere else. The matrix $\mathbf{R}$ is now a 4×4 matrix.

In the absence of data from Cadzand, we need to set a western boundary condition. The initial conditions formulated for Question 8 (with ensemble size $N = 100$) are also used for all simulations in Questions 9 and 10.

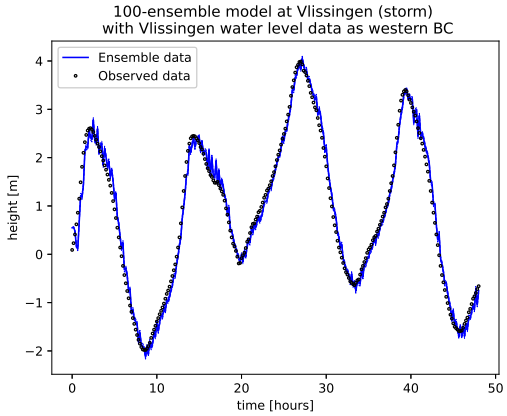
We can assume that $h_{Cadzand} \approx h_{Vlissingen}$ and use data from Vlissingen with stochastic forcing as the western boundary condition. The resulting error statistics for the model with no data assimilation is in Table 7 and the error statistics for the model with data assimilation is shown in Table 8.

Table 7: The different error metrics comparing the model solution for an ensemble size 100 to the observed values. Here the initial condition from Question 8 was used and h values observed at Vlissingen are used as the western boundary condition. The mean of the ensemble members is taken, but data assimilation is not done.

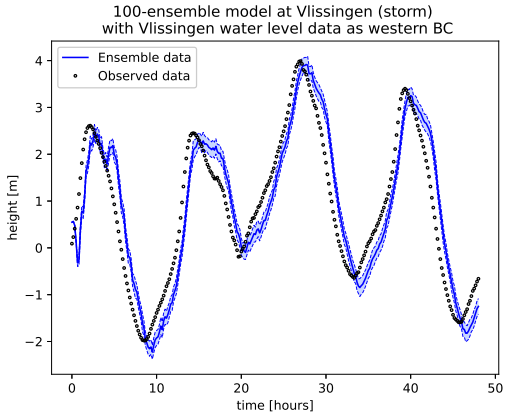
Locations:	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.015583762240876885	-0.1296164254668183	-0.21807387151850335	-0.2862216467331818
RMSE:	0.6384861846697015	0.8508729342515848	0.8941774535323384	0.7244078596341107
InfNorm:	1.4157060190342716	1.8041933272078412	1.9978576581240581	2.2833059192436123
OneNorm:	0.5641751807716279	0.7642424055215256	0.7953989751313012	0.6183941450891989

Table 8: The different error metrics comparing the model solution for an ensemble size 100 to the observed values. Here the initial condition from Question 8 was used and h values observed at Vlissingen are used as the western boundary condition. The ensemble Kalman filter is implemented.

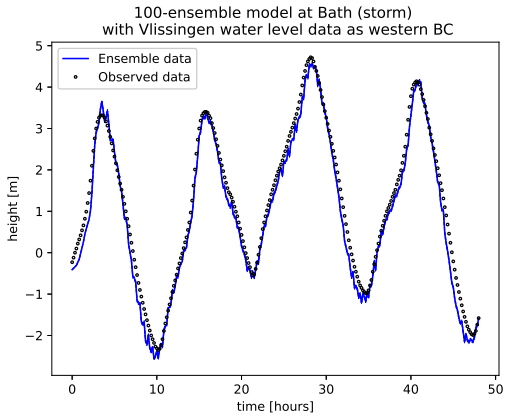
Locations:	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.011143154008687255	-0.07883729516525023	-0.12994668201172063	-0.16486062450147707
RMSE:	0.1993153463374018	0.27010372200929295	0.27853063769233466	0.2579564116408521
InfNorm:	0.6565143194682908	0.8023528684082972	10.9281472495298275	0.7122032719123399
OneNorm:	0.16233153022730148	0.22197057525095926	0.21919790173847864	0.21153353791602447



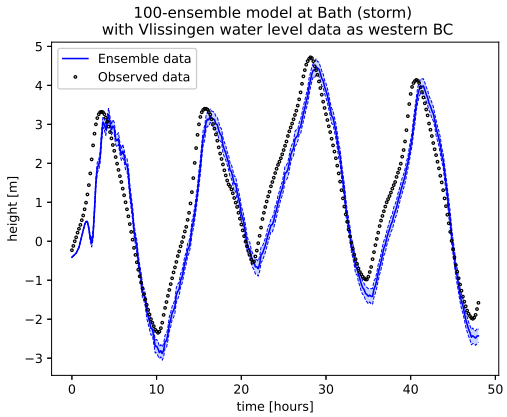
(a) Simulation results at Vlissingen with EnKF



(b) Simulation results at Vlissingen without EnKF



(c) Simulation results at Bath with EnKF



(d) Simulation results at Bath without EnKF

Figure 19: Simulation results with Vlissingen `waterlevel` data as the western boundary condition

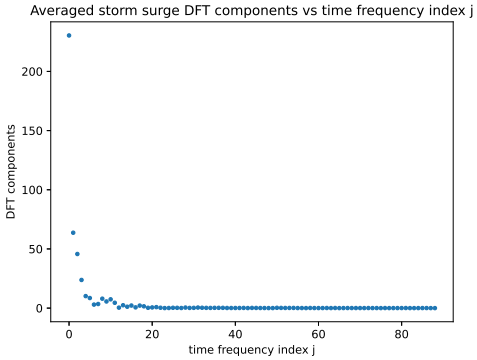
Alternatively, we can also model the storm surge at Vlissingen, Terneuzen, Hansweert and Bath, add this to the filtered Cadzand data, and use the results as the western boundary condition. At each of the locations, the storm surge effects take place from the times listed in Table 9. One-dimensional DFT is done for 178 observations over a 29.5-hour period (henceforth referred to as the storm surge period).

Location	**surge begins (h)**	**surge ends (h)**
Cadzand	18	47.5
Vlissingen	17.6	47.1
Terneuzen	17.2	46.7
Hansweert	16.8	46.3
Bath	16.4	45.9

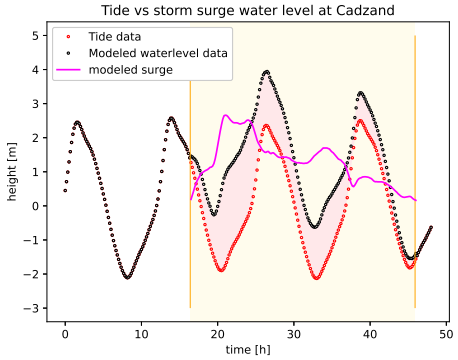
Table 9: Start and end times (in hours) of storm surge effects at the various harbors

The resulting components are then averaged. The averaged components are shown in Figure 20a. The filtered `tide` data, real `waterlevel` data, observed storm surge effect during the storm surge period, and the inverse Fourier transform of the averaged components in Figure 20a are plotted together in Figure 21. The inverse Fourier transform of the averaged components is then added to the `tide` heights observed at Cadzand for $18\text{h} \leq t \leq 47.5\text{h}$. The results are plotted in Figure 20b.

Locations:	Vlissingen	Terneuzen	Hansweert	Bath
Bias:	0.004137950829528959	-0.14267241901515604	-0.23268493353086764	-0.30136117245859084
RMSE:	0.3497951528181346	0.5352636837977665	0.5650624732343079	0.42232231197083797
InfNorm:	0.859077157515437	1.338877701243987	1.5323809548561638	1.5044962167149762
OneNorm:	0.2947911692314723	0.4732904271132865	0.48360500192577044	0.3604184427137092



(a) The amplitudes of the time frequency components as a function of the frequency index j. This gives the amplitudes of the various time frequencies $\omega_t[j] = \frac{2\pi j}{178}$



(b) Filtered water level data in Cadzand with averaged storm surge effect, obtained by applying an inverse Fourier transform to the results plotted in Figure 20a.

Figure 20: DFT components and resulting modeled `waterlevel` data for Cadzand

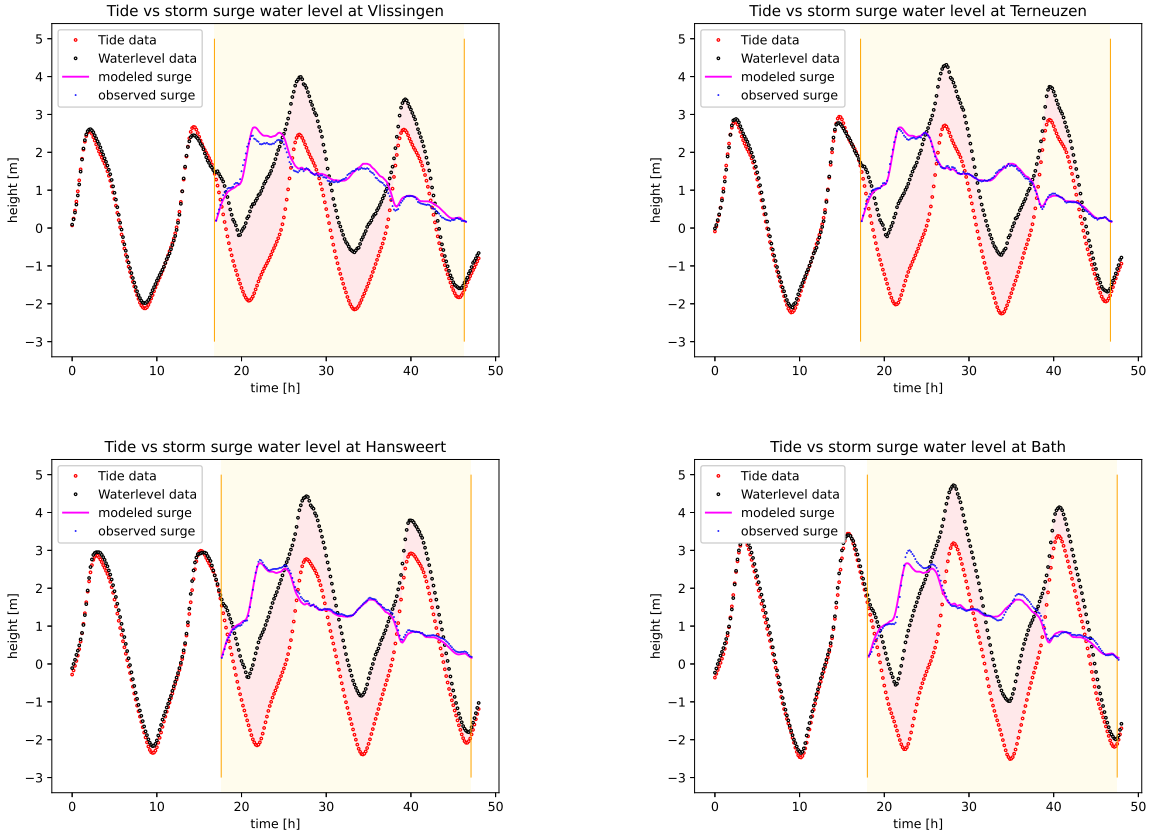
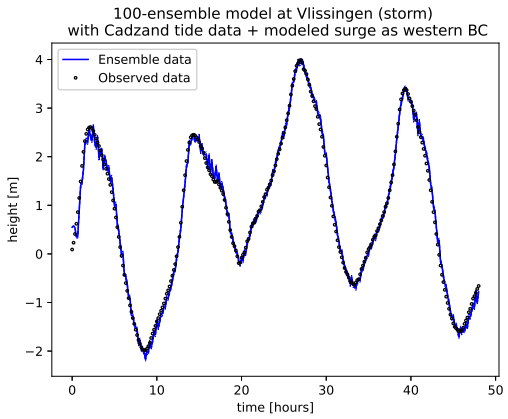
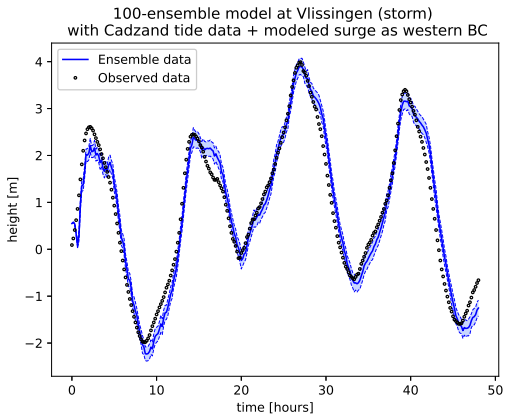


Figure 21: Storm surge effects in the observed data for four locations where both filtered (`tide`) and real (`waterlevel`) data are observed

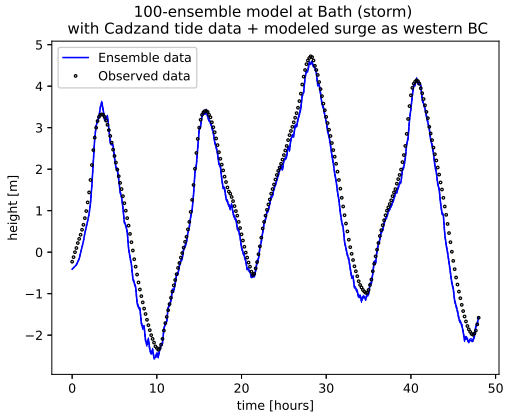
Locations:	Vlissingen	Terneuzen	Hansweert	Bath
Bias:	0.008239002552927413	-0.08103303922192215	-0.13180977919095543	-0.16685599902545825
RMSE:	0.12385164349308055	0.22007648662489	0.25271826313187445	0.2618545693022713
InfNorm:	0.5445918119879689	0.65349576304217	0.821228301033911	0.6744598595555065
OneNorm:	0.09294076755326468	0.17844948963914017	0.2026287027885392	0.21087486594390473



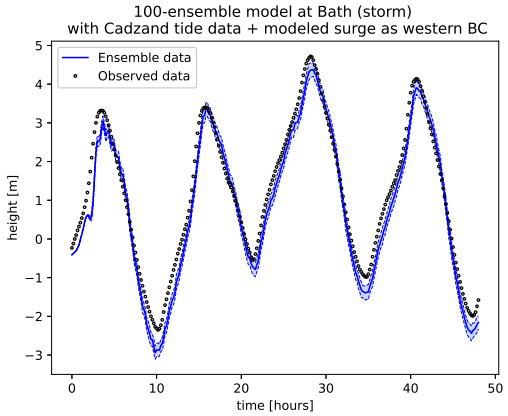
(a) Simulation results at Vlissingen with EnKF



(b) Simulation results at Vlissingen without EnKF



(c) Simulation results at Bath with EnKF



(d) Simulation results at Bath without EnKF

Figure 22: Simulation results with Cadzand `tide` data and modeled storm surge effect as the western boundary condition

For both western boundary condition choices, implementing the EnKF leads to smaller error statistics (except for the bias at Vlissingen when using the simulated Cadzand data), which indicates that data was assimilated into the model during the filtering step. Using Cadzand `tide` data with averaged storm surge effect as opposed to Vlissingen `waterlevel` data results in smaller error statistics. This may be because there appears to be a phase difference between the wave at Vlissingen and the wave at Cadzand.

10 Question 10

Assignment
In real life, the most important aspect of an application of data-assimilation is probably the accuracy of the forecasts (predicting into the future). Now try to mimic this situation in an experiment. Make some forecasts starting some hours before the peak of the storm and check how the accuracy of forecasts for the peak waterlevel depend on lead-time (=time of peak minus start of forecast). Explain the results and provide advice on how to further improve them.

Implementation

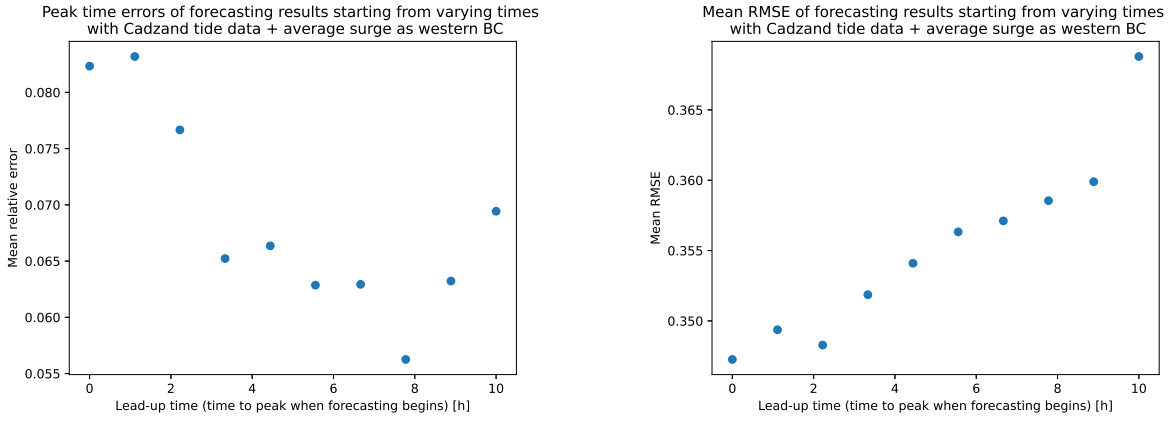
The initial conditions from Question 8 and augmented Cadzand `tide` data is used as the western boundary condition as in Question 9. We assume that the `tide` data can be collected in regular conditions before the storm, so `tide` data for generating initial and boundary conditions can be used for forecasting. Error statistics for forecasts done for $24 \leq t \leq 48$ are shown in Table 10 and those for $16 \leq t \leq 48$ are shown in Table 11.

Table 10: Table of error statistics for forecasting results starting at $t = 24$h. Initial conditions from Question 8 were used and Cadzand `tide` data with modeled storm surge was used as the western boundary condition.

Locations:	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.028968178987153745	-0.08558647180143387	-0.1545147916920594	-0.2085980939896563
RMSE:	0.2688825236222012	0.39736465144898964	0.40947728578768094	0.3217469891262647
InfNorm:	0.7809591743809614	1.0101984211638388	1.010416427373671	0.7925783325287408
OneNorm:	0.19537749181787537	0.32234867072593737	0.33103882955834	0.2710980022796601

Table 11: Table of error statistics for forecasting results starting at $t = 16$h. Initial conditions from Question 8 were used and Cadzand `tide` data with modeled storm surge was used as the western boundary condition.

Locations:	**Vlissingen**	**Terneuzen**	**Hansweert**	**Bath**
Bias:	0.03844950209297211	-0.08345861941033648	-0.15724587212354701	-0.216844795871159
RMSE:	0.29392858306468944	0.42189769380878783	0.43020603739747504	0.32915423090006535
InfNorm:	0.907534226664648	0.9990652648232402	1.08329665213382	0.8135531917831882
OneNorm:	0.2202640045206151	0.35320127294533504	0.36074996653120245	0.2775302397501033



(a) ε_{peak} with varying lead-up times

(b) Mean RMSE with varying lead-up times

Figure 23: Mean RMSE with varying lead-up times

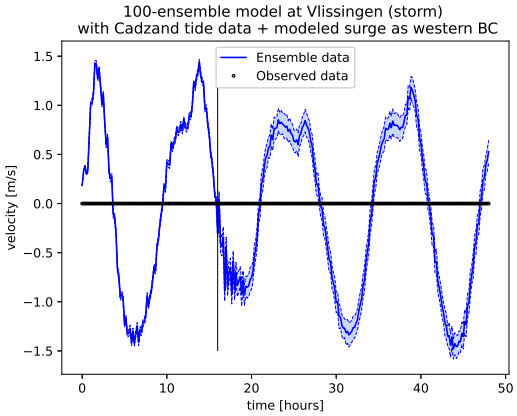
The peak differences in Figure 23a are calculated by identifying the peak $n_t = p$ for each location, then

taking

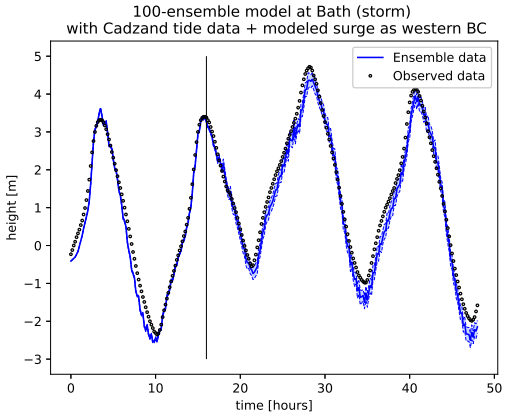
$$\varepsilon_{peak} = \left| \frac{h_{data,p} - h_{model,p}}{h_{data,p}} \right|. \tag{21}$$

Though it may seem surprising that the difference at the peak is higher when the starting forecast time is closer to the peak, the model output is noisier near the beginning of the forecast period, as seen in Figure 24. However, the overall RMSE tends to decrease with lead-up time. This is because the implementing the EnKF up until the starting forecast time essentially means we are running the model from that point in time with an improved initial condition, similar to how an improved initial condition also led to smaller error statistic values in Question 8. Thus, despite the difference between the model and observations being larger at a particular time near the start of the the forecasting period, the overall accuracy of the forecast increases.

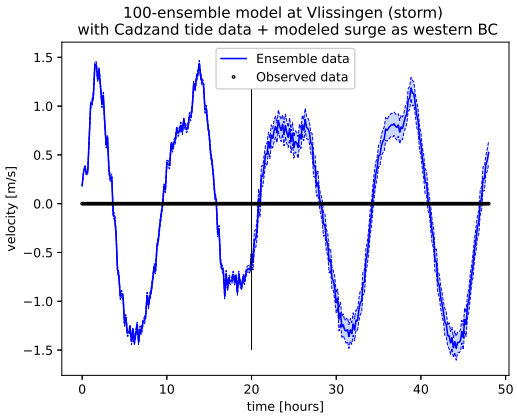
We suggest several changes that can be implemented to obtain improved forecasts. The first and simplest change involves increasing the number of ensemble members, although this comes at increased computational cost as well. The second one involves using more data for assimilation, which can be collected taking additional measurements along the estuary or taking averaged velocity data at at least one location. The third change is optimizing over σ_R (as in the observation error covariance matrix $\mathbf{R} = \sigma_R \mathbf{I}_n$). If σ_R is too large, the measurement step will not assimilate the observations, but if σ_R is too small the ensemble members are too close to each other in value to be meaningful statistically. The changes suggested in Question 2 can also improve the forecast results.



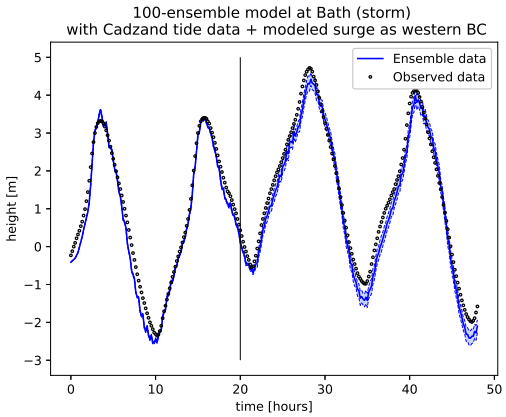
(a) Velocity forecast for $t \geq 16$h at Vlissingen



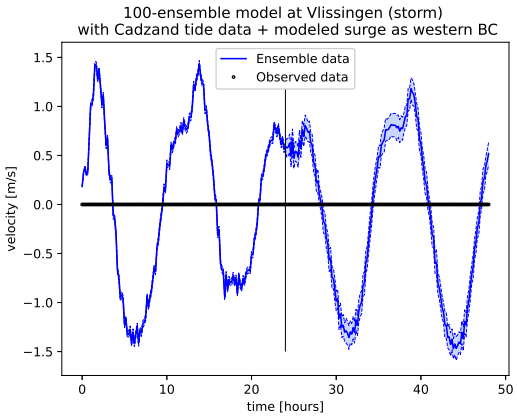
(b) Water level forecast for $t \geq 16$h at Bath



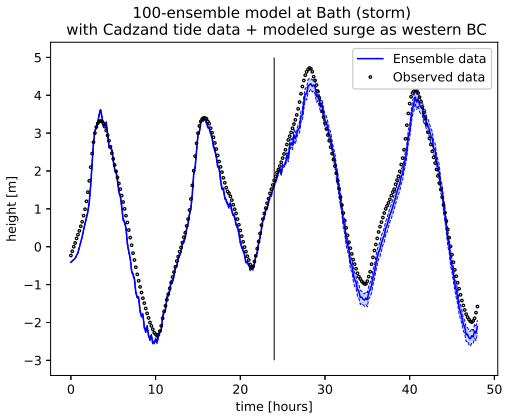
(c) Velocity forecast for $t \geq 20$h at Vlissingen



(d) Water level forecast for $t \geq 20$h at Bath



(e) Velocity forecast for $t \geq 24$h at Vlissingen



(f) Water level forecast for $t \geq 24$h at Bath

Figure 24: Forecasting at Vlissingen from different times

References

[1] M. Katzfuss, J. R. Stroud, and C. K. Wikle, "Understanding the ensemble kalman filter," *The American Statistician*, vol. 70, no. 4, pp. 350–357, 2016. DOI: 10.1080/00031305.2016.1141709. eprint: https://doi.org/10.1080/00031305.2016.1141709. [Online]. Available: https://doi.org/10.1080/00031305.2016.1141709.

[2] J. M. Lewis, S. Lakshmivarahan, and S. Dhall, "Reduced-rank filters," in *Dynamic Data Assimilation: A Least Squares Approach* (Encyclopedia of Mathematics and its Applications), Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2006, pp. 534–560. DOI: 10.1017/CBO9780511526480.031.

[3] E. Kwiatkowski and J. Mandel, "Convergence of the square root ensemble kalman filter in the large ensemble limit," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 3, no. 1, pp. 1–17, 2015. DOI: 10.1137/140965363. eprint: https://doi.org/10.1137/140965363. [Online]. Available: https://doi.org/10.1137/140965363.