Technische Universiteit Delft
Faculteit Elektrotechniek, Wiskunde en Informatica
Delft Institute of Applied Mathematics

Nitrogen dioxide source detection in satellite images, using inverse time integration methods.

(Stikstofdioxidebron detectie in satellietbeelden, gebruikmakende van inverse tijd integratie methoden.)

Verslag ten behoeve van het
Delft Institute of Applied Mathematics
als onderdeel ter verkrijging

van de graad van

BACHELOR OF SCIENCE
in
TECHNISCHE WISKUNDE

door

Bodille Petronella Maatje Blomaard

Delft, Nederland
Juli 2021

BSc verslag TECHNISCHE WISKUNDE

"Nitrogen dioxide source detection in satellite images, using inverse time integration methods."

"(Stikstofdioxidebron detectie in satellietbeelden, gebruikmakende van inverse tijd integratie methoden.)"

BODILLE PETRONELLA MAATJE BLOMAARD

Technische Universiteit Delft

Begeleider
Dr. N.V. Budko

Overige commissieleden
Prof.Dr.ir. A.W. Heemink
Dr.ir. A.J. Segers
Dr.ir. J.P. Tokaya

Juli, 2021, Delft

Abstract

To regulate pollution emission of NO_2 and greenhouse gasses like carbon dioxide and methane, satellite images of the TROPOspheric Monitoring Instrument (TROPOMI) are used to map the spread of air pollution. This instrument delivers near-daily 'snapshots' of the NO_2 concentration over the whole Earth. In this report, first, the spread of NO_2 pollution is modelled using a so called convection-diffusion equation. This equation is constructed from a convection term, which contributes to how much the wind transports matter, and a diffusion term which described the natural mixing of gases. Next, NO_2 pollution sources were reconstructed using the inverse of the time propagation of the model. Thus, known NO_2 concentrations are used to reconstruct temporal information. By finding out what is missing between data snapshots, sources can be reconstructed. The verification of the algorithm was first performed on artificial data, after which it was applied to experimental data.

The convection-diffusion equation that was used is a two dimensional equation, using longitude and latitude as Cartesian coordinates. The equation was solved by the method of lines, i.e. first a spatial discretisation was made using the Finite Volume Method, after which time integration methods were used to propagate the problem in time. In this report two time integration methods formed the basis for the time propagation and two source reconstruction algorithms. The algorithms take two NO_2 snapshots $\mathbf{y}_n$ and $\mathbf{y}_{n+m}$, with m time steps in between. First the inverse form of Forward Euler is used, which performed well on artificial data on a small time scale. Afterwards the Backward Euler method is used to define the inverse problem of finding the source function. This algorithm performed well on artificial data with 24 hours between snapshots. The algorithms were found to be sensitive to noise and to be influenced greatly by the second NO_2 snapshot. After applying the algorithm to experimental data it was found that the algorithm is able to localise extended sources of emissions, but is not capable of pin-pointing individual point sources. In addition, sources with low emission rates were left undetected due to information being lost in the noise. Emission rates were not reconstructed accurately either, although it was able to find the main sources of emissions. The performance of the algorithm will take advantage from more frequent data. With this improvement, this method is a viable option for finding not only NO_2 sources, but can also be applied to other tropospheric gases.

Contents

1 Introduction

Nitrogen dioxide (NO_2) is an atmospheric trace gas that plays a leading role in air quality issues. Not only does it directly affect human health, it is also essential for the formation of acid rain and ozone in the troposphere, which are harmful for the environment. As the world is working on trying to slow down climate change, the need for monitoring instruments rises. In October 2017, the Sentinel-5 satellite was launched with on board the TROPOspheric Monitoring Instrument (TROPOMI). This instrument is capable of mapping the air quality of the entire Earth, giving daily 'snapshots' of, among other things, NO_2 concentrations. These snapshots provide the concentration of NO_2 in the troposphere, and can be used to investigate the dispersion of pollution. The modelling of the spread of pollution, as well as finding the locations of pollution sources, has been a topic of interest for years. With the coming of TROPOMI, air pollution spread can be modelled more precisely than ever before.

The objective for this report is to use satellite images of NO_2 concentrations over the Netherlands, produced by TROPOMI, to find the sources of NO_2 emissions. The basis for the research lies in the application of a two-dimensional convection-diffusion equation to describe the transport of the pollution. After which the inverse source problem is tackled by using the inverses of time integration methods. Which leads to the research question that will be answered in this report:

Can an algorithm based on an inverse time integration method be used to accurately reconstruct NO_2 pollution sources?

This report will be an orientational research into the possibilities of using the inverse of the Forward and Backward Euler method as bases for source reconstruction algorithms.

The report will start by giving information on NO_2 and a brief description of TROPOMI in Section 2. In the following Section, the data that will be used in this report is featured, including the measures that need to be taken to make the data usable. Thereafter, in Section 4 the model that will be used to numerically simulate the transport of NO_2 is defined. This model is then used to estimate transport parameters, by visually comparing the solutions of the model to experimental data. Following is Section 5 in which the inverse source problem is introduced and two source reconstruction algorithms are defined and tested. Finally, in Section 6 the source reconstruction algorithm is applied to experimental data. This report is finished by a conclusion and discussion, which can be found in Sections 7 and 8 respectively.

All of the code that will is written for this report is written in Python, and is available on request by reaching out to Neil Budko (n.v.budko@tudelft.nl).

2 TROPOMI

The data that is going to be used in this report consists of data products from the TROPOspheric Measuring Instrument (TROPOMI). TROPOMI is an instrument that can measure gases such as ozone, CO_2, CH_4 and NO_2 in both the stratosphere and the troposphere (the higher and lower atmosphere). In this report, however, only the tropospheric column of NO_2 will be of interest. The instrument is located on the satellite Sentinel-5. This satellite was launched on 13 October 2017, planned for a seven year mission [24]. In this section it will first be explained what nitrogen dioxide is and what the impact of this gas is on the environment. Afterwards there will be a brief explanation on how the TROPOMI instrument works and what type of data product the instrument creates. The section is concluded by a discussion on the accuracy of the instrument.

2.1 Nitrogen dioxide

Nitrogen dioxide is a gas that is produced during the combustion of fossil fuels. Therefore industry, power plants, and traffic on both the roads and the waterways are important sectors for the emission of NO_2 [12]. The areas where the yearly average NO_2 concentrations are the highest in the Netherlands are: Amsterdam and Rotterdam. Outside of the Netherlands, close to the border, Antwerp and the Ruhrgebiet are big polluters.
High concentrations of NO_2 can have a direct effect on the human health. Exposure over short periods can aggravate respiratory diseases, longer exposure can potentially increase susceptibility to respiratory infections [7]. This is due to NO_2 pollution at a surface level, on an atmospheric level the gas has an environmental effect. NO_2 and other nitrogen oxides can interact with water, oxygen and other chemicals in the atmosphere to form acid rain [7]. Acid rain has harmful effects on plants, animal life, causes corrosion on buildings and indirectly affects human health [19]. In addition, in the troposphere, NO_2 is one of the main sources of oxygen atoms for the formation of ozone. Which not only is damaging for the environment, among which it decreases plant productivity. It is also harmful for human health, it causes 0.7 million $\pm$ 0.3 million respiratory mortalities yearly worldwide [1]. Due to chemical processes in the troposphere, NO_2 has a lifetime of approximately one day [14]. So even though NO_2 is not a greenhouse gas, the indirect effects on climate change and human health are large.

2.2 How it works

The way TROPOMI works is visualised in Figure 1. As the satellite, on which TROPOMI is located, orbits the Earth it images a strip of the of the Earth onto a grid. This strip is around 2600 km wide and approximately seven kilometers long, measured during one second. In this way the instrument can cover the Earth on a daily bases. The strip is partitioned into pixels, each pixel corresponds to a column on the grid inside the instrument, and has a size of approximately 7×3.5 km^2. The incoming light is separated by wavelength and partitioned over the column on the grid in the instrument. This way, it can be compared per wavelength how much light is measured.

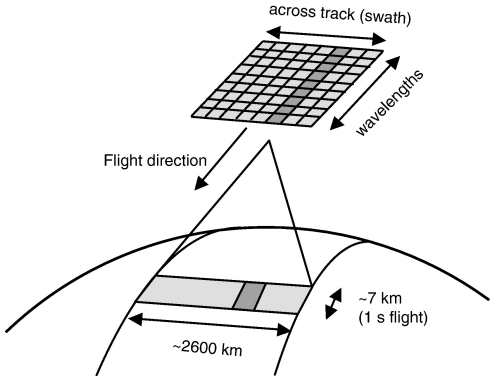


Figure 1: Overview of the working of TROPOMI [24].

The spectral bands of TROPOMI, over which the light is partitioned, cover light spectra from 270 to 500 nm, from 675 to 775 nm and from 2305 to 2385 nm. This can be seen in Figure 2. In this figure the absorption spectrum of various gases are also set out. The instrument can measure which gas is in the

troposphere and in what concentration by analysing the measured wave lengths detected. For example, if there is a large NO_2 pollution plume in the troposphere, it will absorb light of a certain wavelength as can be seen in Figure 2. This ranges somewhere between 350 and 600 nm. This means that the instrument will measure less light in the range of 350 to 600 nm. With this information the instrument can not only detect the presence of NO_2 in the troposphere, but also its concentration.

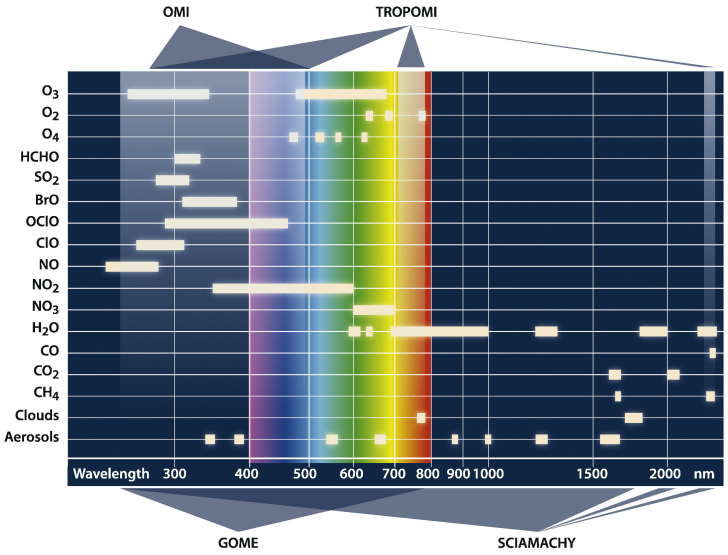


Figure 2: The spectral ranges for the instrument TROPOMI and it's predecessors: OMI, GOME and SCIAMACHY [24].

Using this method, TROPOMI creates several types of data products among which the tropospheric column of NO_2. This gives the amount of NO_2 molecules in the column above every pixel, in the troposphere.

The method for detecting pollution as stated above is the main idea behind the instrument, in reality this data retrieval is more complicated. It consists of three steps, in the first step the amount of NO_2 molecules along the effective light path from Earth through atmosphere to satellite is calculated. This is called the Slant Column Density (SCD). Next, the SCD is separated into its stratospheric and tropospheric components. In the final step the vertical stratospheric and tropospheric column densities are determined [23]. This means that there are multiple steps in which errors can occur, which could make the data less accurate. In the following section the accuracy of the data will be analysed.

2.3 Accuracy of the data

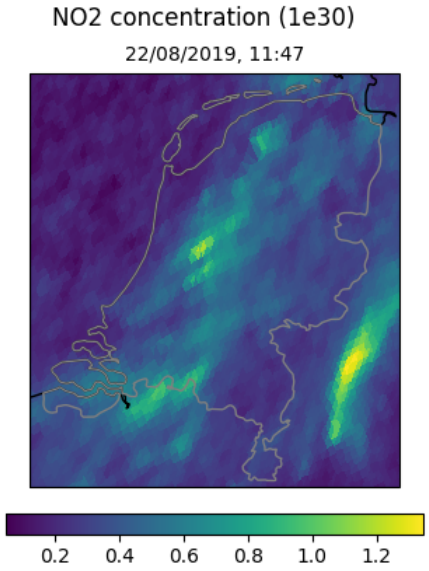
There are multiple factors which influence the accuracy of the data. As shortly mentioned in the previous section, there can be uncertainty in the steps for processing the data. In addition, the weather conditions have an influence and there can be a decrease in accuracy over highly polluted areas.
To start, the bias requirement for the tropospheric NO_2 retrieval is set to be 25% to 50% [6]. In addition it is found that the tropospheric NO_2 columns have a negative bias over polluted areas, close to the source. There, the peak of the NO_2 concentration is close to the ground. On the other hand, the concentration of NO_2 may be overestimated away from big sources [10].
Next, the influence of the weather conditions on the accuracy of the data will be analysed. The focus lies mainly on the cloud coverage. TROPOMI uses a low surface albedo (measure of reflectivity) of 2% for the signal-to-noise requirements [24]. This means that cloud-free pixels are of most interest. Since the albedo grows as the cloud fraction grows [5], more cloud coverage results in less reliable data, therefore cloud covered pixels can be filtered out.
Finally, when NO_2 is measured over strongly polluted areas, there is a bigger chance of detecting very large NO2 concentrations for individual pixels. It is currently unlikely to encounter NO2 concentrations that are not optically thin in the TROPOMI data, except in a few individual pixels [23]. Therefore the influence of strongly polluted areas will be neglected.

3 Data

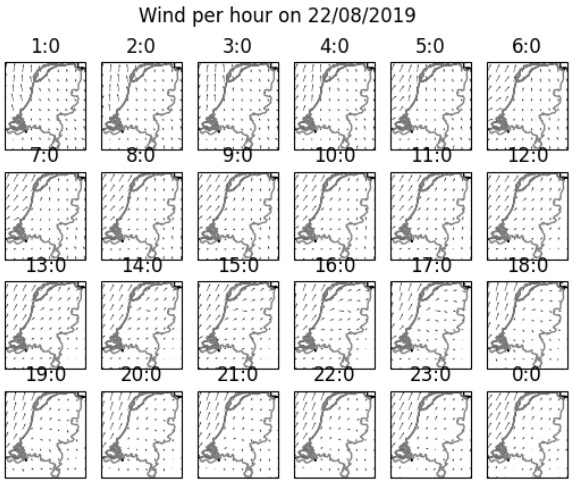
In this section the data that is received from TNO (Netherlands Organisation for Applied Scientific Research) will be shown and explained briefly, the steps needed to make the data usable will be set out as well. It will first be described how a selection is made which NO_2 data is determined to be useful. Afterwards it will be described how the data has to be interpolated, and the result of the interpolation will be shown and discussed. Finally, it will be described what transformations are needed to convert the data to the right units of measurement.

3.1 Received data sets

There are two different types of data that are received from TNO, namely data on the concentration of NO_2 and on the direction and speed of the wind. Both of this data is given for the months: June, July and August. First the NO_2 data set will be described. As described in Section 2.2, the data which TROPOMI produces currently has a pixel size of $7 \times 3.5\,\mathrm{km}^2$. For each pixel the tropospheric vertical column of nitrogen dioxide is given. That is, the amount of molecules of NO_2 are given per squared centimeter for the column above the pixel, with a precision of $1 \times 10^{15}\,\mathrm{mol./cm}^2$ [24]. This data is given roughly on a daily basis and will be referred to as 'snapshots'. Since TROPOMI is not a stationary measuring instrument, the concentration of NO_2 in the north of the Netherlands is given on a slightly different time than the concentration in the south of the Netherlands. It is found that the maximal time interval for a total measurement of NO_2 pollution over the Netherlands is one minute. Therefore it will be assumed that all of the data is measured at the same time. In Figure 3a a visualisation of the data is shown. In this figure the pixels are clearly visible and thus the picture looks rough.



(a) NO_2 concentration spread over the Netherlands on 22/08/2019.



(b) Hourly wind direction and speed in the Netherlands on 22/08/2019.

Figure 3: Visualisations of the received data.

As previously mentioned, in addition to the the NO_2 data, there is also data on the wind. This data consists of two values: U and V given per hour, corresponding to the horizontal and vertical component of the wind respectively. These components are given in m/s. Using these components the direction and speed of the wind can be determined. The wind data can in turn be visually represented by arrows, as is done in Figure 3b.

3.2 Selection of data

As was written in the beginning of this section, the first step in finding usable data is to make a selection of which NO_2 snapshots will be used. To be precise, which measurements of NO_2 concentration. As described in Section 2.3, not every day is optimal for the measuring of NO_2. Days with a lot of cloud coverage are less suitable for the measurement of NO_2 concentrations than cloud-free days. This is

because the retrievals are only reliable up to a certain cloud fraction [24]. Thus for cloudy days a lot of information is too unreliable to use and is filtered out. To make sure the data that will be used in this report has enough coverage of reliable data points over the Netherlands, a threshold is set. The condition for an NO_2 snapshot to be used in the report is that it has to have at least 3000 data points. This resulted in that only data on four days in June, two days in July and seven days in August are selected to be usable.

3.3 Interpolation of the data

After the selection of usable data, the data will be processed to be used in a numerical calculation. The NO_2 concentration and the wind speed and direction needs to be available on certain points determined by the discretisation. For the discretisation the domain will be partitioned into a grid with $N \times N$ interior points. The NO_2 data will be needed on all of these interior points and the wind data will be needed between and around all of these points. First, it will be described how the NO_2 data needs to be interpolated.
The domain on which the data for the NO_2 concentration and the wind is given is the following: $3.16 \leq x \leq 7.45$ and $50.65 \leq y \leq 53.67$ in degrees longitude and latitude respectively. Since the wind data is needed on a slightly bigger domain than the interior points, the domain on which the interior points will be defined will have the following size:

$$\begin{aligned} \text{longitude:} \quad & 3.2 \leq x \leq 7.4, \\ \text{latitude:} \quad & 50.7 \leq y \leq 53.6. \end{aligned}$$

This domain will be used throughout the report and covers the Netherlands entirely, in addition, parts of Germany and Belgium are also within the domain.

3.3.1 NO_2 data

The given NO_2 data is given on a set of coordinates which are not the same as the interior points on which the data is needed. In addition, the coordinates of the measured values slightly differ per day. This is because TROPOMI is not stationary, but is on a different location every time. The data will be interpolated on a grid with $N_y \times N_x$ interior points using linear interpolation, on the domain specified in the previous section. If extrapolation is needed to find the value of a certain interior point, the value zero is given.
To check whether the interpolation is usable it can be visually compared to the given data. For this the Figures 3a and 4 can be compared. It can be seen that the interpolated data looks less blurred and more precise. Other than a sharper image, the figures show the same clouds of NO_2 pollution. Thus there is no information lost in the interpolation.

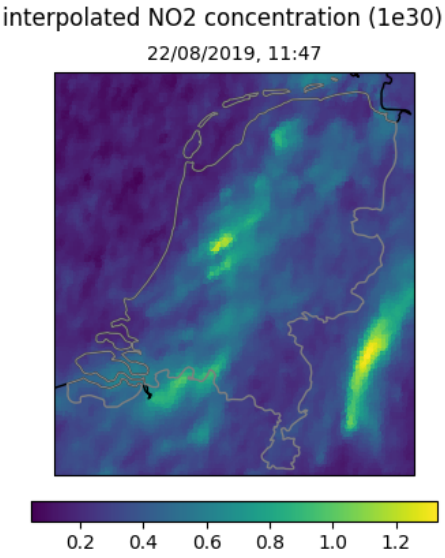


Figure 4: Visual representation of the interpolated data on 22/08/2019.

3.3.2 Wind data

In addition to the NO_2 data, the wind data will also need to be interpolated. This will be done both in space and in time to make sure the data is at the desired location and can be calculated for every desired time. To interpolate this data, either in space or in time, it is not possible to simply interpolate the U and V component, since these are vectors. Therefore the wind amplitude will be calculated per point. Afterwards the wind amplitudes can the interpolated to find the amplitude of the wind on the desired point. Separately the U and V components will be interpolated to find the corresponding direction of the wind. Finally the found U and V components are normalised and afterwards multiplied by the amplitude of the wind. This is necessary since the interpolation of unit vectors do not always result in unit vectors, see Appendix B. This way the information remains.
First the data will be interpolated in space. From the final discretisation equation (3) one finds that the value of the wind is wanted on different points than the NO_2 concentration. This is visualised in Figure 5, the black circles are the points (x_i, y_j) where the NO_2 concentration is calculated. The stars represent the points on which the wind data is needed, for the red coloured stars the vertical component (V) is needed. For the white stars the horizontal component (U) of the wind is needed.

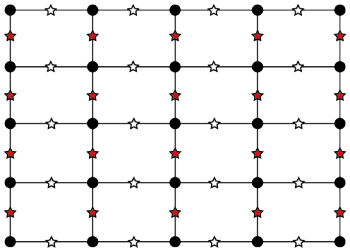


Figure 5: Overview of data-points.

As stated above, the three components of the wind data: the amplitude and the U and V components, will be interpolated separately. The amplitude of the wind vector will be needed for all stars in Figure 5. The normalised U component will be calculated for the white stars and the normalised V components will be calculated for the red stars. This results in two different matrices with a different shape. The matrix containing the U components will be a $N_y \times (N_x + 1)$ matrix, whereas the matrix containing the V components of the wind will be a $(N_y + 1) \times N_x$ matrix.
In addition, the wind data will have to be interpolated in time, to match the NO_2 data. For this a function will be written that returns the U and V component matrices for the desired time. Again to preserve the right information the amplitude and and directions of the wind are interpolated separately. For the NO_2 data it was sufficient to fill all of the values outside of the convex hull of the received data points with zeroes. For the wind data this resulted into problems with the discretisation. Therefore the domain containing the Netherlands was taken to be the size as defined in Section 3.3, such that no extrapolation is necessary.

3.4 Transformation of units

In Appendix A an approximation is made to show that longitude and latitude can be used as Cartesian coordinates for the discretisation of the model. This means that the units of measurements of the data will need to be converted to work with longitude and latitude. The units of the NO_2 data and the wind data will be separately converted, starting with the wind data.
The wind data is given in meter per second, but should be converted to degrees per hour, since the time is given in hours. To do this conversion, first the unit can be changed to kilometers per hour, this is done by multiplying by 3.6. Now, kilometer has to be converted to longitude and latitude to receive the wind data in degrees per hour. Kilometers can be written in degrees in the following way [16]:

$$\begin{aligned} \text{latitude: } & 1^\circ \approx 111\,\text{km}, \\ \text{longitude: } & 1^\circ \approx 111\cos(\text{latitude})\,\text{km} \quad \Rightarrow \quad 1^\circ \approx 66.6\,\text{km}. \end{aligned}$$

As mentioned in Appendix A, cos(latitude) was taken as a constant $\cos\hat{\theta}_0$, since the latitude takes value between 50.65 and 53.70 the cosine takes value between 0.63 and 0.59. For simplicity $\cos(\text{latitude}) = \cos\hat{\theta}_0$ will be approximated by 0.6. This gives that one degree in longitude is approximately equal to 67 km.

Using this, the horizontal (U) and vertical (V) components of the wind vectors can be rewritten separately in the following way:

$$\text{U: } 1\,\mathrm{m\,s^{-1}} \approx 3.6\,\mathrm{km/h} \approx 5.4 \times 10^{-2}\,\mathrm{deg/h}$$
$$\text{V: } 1\,\mathrm{m\,s^{-1}} \approx 3.6\,\mathrm{km/h} \approx 3.2 \times 10^{-2}\,\mathrm{deg/h}$$

This means that the received wind data has to be multiplied with a constant in the following way: the U component will be multiplied by 5.4×10^{-2} and the V component will be multiplied by 3.2×10^{-2} It is important to note that this conversion should be done after interpolation of the data, since one degree in longitude is not equal to one degree in latitude the length of a vector cannot be described in degrees only.

Next, the units of measurement of the concentration u have to be converted from molecules per centimeter squared (mls/cm^2) to molecules per degree squared (mls/deg^2). A square degree in longitude and latitude is equal to the multiplication of one degree in longitude times one degree in latitude, this gives that: $1^{\circ^2} \approx 7392$ km^2 $\approx 7.392 \times 10^{13}$cm^2. This gives that the NO_2 data should be multiplied by 7.392×10^{13} to convert it to molecules per degree squared.

Now, using the above methods the data received is converted to the units: molecules, degrees longitude, degrees latitude, and hour. In the rest of the report the variables x and y will be used to represent longitude and latitude respectively.

4 Numerical simulation of NO_2 transport

The goal of the first part of this report is to simulate NO_2 pollution plumes above the Netherlands. This will be done by using a two dimensional convection diffusion equation. This is a partial differential equation which consists out of a convection part, which determines the extent to which the matter is transported by the wind, and a diffusion part, which determines how much the matter spreads out over time. The exact equation that will be used is explained in the following section.
The convection diffusion problem that is going to be used is solved by using the method of lines. That is, the problem is first spatially discretised, after which the problem is integrated in time. This process is documented in the Sections 4.2 and 4.3.
The final subsections of this section will focus on the determination of parameters. First the value for the time step and the grid size will be determined to avoid stability and discretisation errors. This section will be concluded by an estimation of transport parameters.

4.1 Two-dimensional convection-diffusion model

A linear Eulerian air pollution dispersion model in three dimensions is based on the following partial differential equation, see e.g. [15]:

$$\frac{\partial u}{\partial t} + w^x \frac{\partial u}{\partial x} + w^y \frac{\partial u}{\partial y} + w^z \frac{\partial u}{\partial z} = \frac{\partial}{\partial x}\left(D_x \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(D_y \frac{\partial u}{\partial y}\right) + \frac{\partial}{\partial z}\left(D_z \frac{\partial u}{\partial z}\right) + f,$$

where $u(x, y, z, t)$ is the concentration of NO_2, $w^x(x, y, z, t)$, $w^y(x, y, z, t)$ and $w^z(x, y, z, t)$ are the Cartesian wind components, and $D_x(x, y, z, t)$, $D_y(x, y, z, t)$ and $D_z(x, y, z, t)$ are the components of the turbulence diffusion tensor $D(x, y, z, t)$. The term $f(x, y, z, t)$ is the source term.

In this report a simplified horizontal dispersion model in two dimensions, x and y, as in [9] is considered:

$$\frac{\partial u}{\partial t} + w^x \frac{\partial u}{\partial x} + w^y \frac{\partial u}{\partial y} = D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) - \alpha u + f,$$

where $u(x, y, t)$ is the horizontal distribution of NO_2, $w^x(x, y, t)$ and $w^y(x, y, t)$ are the horizontal components of the wind, D is the diffusion coefficient, and $f(x, y, t)$ is the source function. In addition, there is a sink term αu with $\alpha \geq 0$. The derivation of this equation in latitude-longitude coordinates can be found in Appendix A.

We shall consider the following conservative form of this two-dimensional convection-diffusion problem:

$$\begin{aligned} &\frac{\partial u}{\partial t} = D\Delta u - \nabla \cdot (\mathbf{w}u) - \alpha u + f, \quad (x, y) \in \Omega, \quad t > 0; \\ &u(x, y, t) = g(x, y, t), \quad (x, y) \in \Gamma, \quad t > 0; \\ &u(x, y, 0) = u_0(x, y), \quad (x, y) \in \overline{\Omega}, \end{aligned} \tag{1}$$

where $\mathbf{w} = \langle w^x, w^y \rangle$ is the vector function of the wind and g, u_0 are given functions.

The meaning of the diffusion term requires explanation. Indeed, the molecular diffusion of NO_2 in air is so small that it can be ignored. In the original paper [8] the molecular diffusion coefficient was given a value of $1.54 \times 10^{-1}\,\mathrm{cm}^2/\mathrm{s}$ at 21.1 °C at 1 atm pressure. In other, more recent, papers a similar value can be found [13]. This means that the molecular diffusitivity is of order 10^{-12}, which is very small in comparison to the amplitude of the wind.

The approximately diffusive transport of NO_2 is caused by the mixing of NO_2 with the air due to small eddy currents in the ever turbulent atmosphere. Therefore, the diffusion constant D is effective in nature and is usually estimated from the observed lateral expansion of extended NO_2 plumes.

From the records of tests with manned balloons [17], the horizontal eddy-diffusitivity has been found to be in the order of $2 \times 10^8 \mathrm{cm}^2/\mathrm{s}$. This will give the effective diffusion coefficient D in the order of $9.7 \times 10^{-4} \mathrm{deg}^2/\mathrm{h}$. Since the determination of this constant is based on a dated study, the coefficient D is still subject to change. The order of this coefficient will however remain between: 1×10^{-4} and 1×10^{-2}.

4.2 Spatial discretisation

The PDE (1) will be discretized using the Finite Volume Method (FVM). In simulations the homogeneous Dirichlet boundary condition $g = 0$ is assumed.

The first step of the FVM is to define the grid and the control cells. The received data that is given is defined on a certain domain, and interpolated on a $N_y \times N_x$ grid. The interior points of this domain will be taken as interior points for the discretisation. Around the interior points there will be the boundary points, thus if there will be $N_y \times N_x$ interior points, the total grid will have $(N_y+2)\times(N_x+2)$ points. For data over the domain $(x, y) \in [L_1, L_N] \times [H_1, H_N]$, one gets the grid steps $h_x = \frac{L_N - L_1}{N_x - 1}$ and $h_y = \frac{H_N - H_1}{N_y - 1}$.

The vertex-centered FVM will be used for both internal and boundary points of the grid, since the Dirichlet boundary conditions are assumed. Each internal point has a control volume (cell) Ω of size $h_x \times h_y$ associated with it as shown in Figure 6, while points on the boundary Γ have control half-cells, and the four corner points have control quarter-cells.

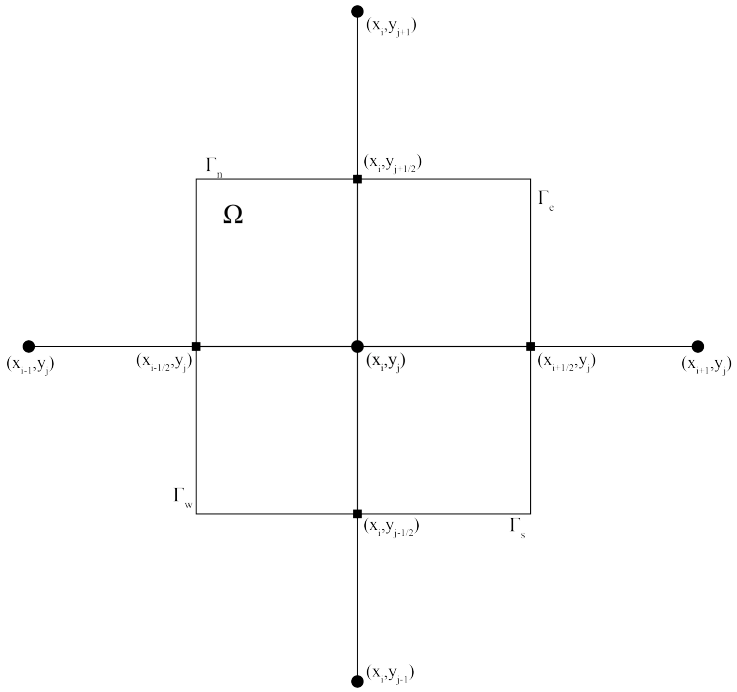


Figure 6: Control cell.

The partial differential equation (1) is first re-written as:

$$\frac{\partial u}{\partial t} = \nabla \cdot (D\nabla u - \mathbf{w}u) - \alpha u + f$$

This equation can be integrated over the control cell, afterwards the Gauss divergence theorem is applied.

$$\int_\Omega \frac{\partial u}{\partial t}\, d\Omega = \int_\Omega \left(\nabla \cdot (D\nabla u - \mathbf{w}u) - \alpha u + f\right)\, d\Omega$$
$$\int_\Omega \frac{\partial u}{\partial t}\, d\Omega = \int_\Gamma \left((D\nabla u - \mathbf{w}u) \cdot \mathbf{n}\right)\, d\Gamma - \int_\Omega \alpha u\, d\Omega + \int_\Omega f\, d\Omega$$

The integral over the boundary can be split into four parts representing each of the sides of the control cell: north(n), east(e), south(s) and west(w). Denoting the components of the wind vector as $\mathbf{w} = \langle w^x, w^y \rangle$ we arrive at:

$$\int_\Omega \frac{\partial u}{\partial t}\, d\Omega = \int_{\Gamma_n} \left(D\frac{\partial u}{\partial y} - w^y u\right)\, d\Gamma + \int_{\Gamma_e} \left(D\frac{\partial u}{\partial x} - w^x u\right)\, d\Gamma + \int_{\Gamma_s} \left(-D\frac{\partial u}{\partial y} + w^y u\right)\, d\Gamma +$$
$$\int_{\Gamma_w} \left(-D\frac{\partial u}{\partial x} + w^x u\right)\, d\Gamma - \int_\Omega \alpha u\, d\Omega + \int_\Omega f\, d\Omega$$

For the integrals over boundaries a one-point integration will be applied, for the integrals over the control cell Ω the point (x_i, y_j) is taken. Multiplying this by the width and length of the control cell, h_x and h_y respectively, results in an approximation for the integrals over the control cell. For the integrals over the boundary of the control cell, the middle point of the boundary is chosen as integration point.

This means that for integration over the northern boundary the point (x_i, y_{j+1}) is taken, for reference see Figure 6. This is in turn multiplied with the length of the specific boundary, in the case of the northern and southern boundary this is h_x. For the derivatives a cell centered discretisation is taken. Furthermore, $u(x_i, y_j, t)$ will be written as $u_{i,j}$ and the dependence on time will be suppressed. This leads to the following discretisation for the interior points:

$$\begin{aligned} h_x h_y \frac{du_{i,j}}{dt} &= Dh_x \frac{u_{i,j+1} - u_{i,j}}{h_y} - h_x w^y_{i,j+\frac{1}{2}} u_{i,j+\frac{1}{2}} + Dh_y \frac{u_{i+1,j} - u_{i,j}}{h_x} - h_y w^x_{i+\frac{1}{2},j} u_{i+\frac{1}{2},j} \\ &\quad - Dh_x \frac{u_{i,j} - u_{i,j-1}}{h_y} + h_x w^y_{i,j-\frac{1}{2}} u_{i,j-\frac{1}{2}} - Dh_y \frac{u_{i,j} - u_{i-1,j}}{h_x} + h_y w^x_{i-\frac{1}{2},j} u_{i-\frac{1}{2},j} \\ &\quad - h_x h_y \alpha u_{i,j} + h_x h_y f_{i,j}. \end{aligned} \tag{2}$$

Dividing this by $h_x h_y$ we obtain:

$$\begin{aligned} \frac{du_{i,j}}{dt} &= D\frac{u_{i,j+1}(t) - u_{i,j}(t)}{h_y^2} - \frac{1}{h_y} w^y_{i,j+\frac{1}{2}} u_{i,j+\frac{1}{2}}(t) + D\frac{u_{i+1,j}(t) - u_{i,j}(t)}{h_x^2} \\ &\quad - \frac{1}{h_x} w^x_{i+\frac{1}{2},j} u_{i+\frac{1}{2},j}(t) - D\frac{u_{i,j}(t) - u_{i,j-1}(t)}{h_y^2} + \frac{1}{h_y} w^y_{i,j-\frac{1}{2}} u_{i,j-\frac{1}{2}}(t) \\ &\quad - D\frac{u_{i,j}(t) - u_{i-1,j}(t)}{h_x^2} + \frac{1}{h_x} w^x_{i-\frac{1}{2},j} u_{i-\frac{1}{2},j}(t) - \alpha u_{i,j} + f_{i,j} \end{aligned}$$

For the u on the boundary of the control cell, the value can be approximated by interpolation, so we have:

$$\begin{aligned} u_{i,j+\frac{1}{2}}(t) &= \frac{u_{i,j}(t) + u_{i,j+1}(t)}{2} \\ u_{i+\frac{1}{2},j}(t) &= \frac{u_{i,j}(t) + u_{i+1,j}(t)}{2} \\ u_{i,j-\frac{1}{2}}(t) &= \frac{u_{i,j}(t) + u_{i,j-1}(t)}{2} \\ u_{i-\frac{1}{2},j}(t) &= \frac{u_{i,j}(t) + u_{i-1,j}(t)}{2} \end{aligned}$$

This can be substituted into the equation, which gives:

$$\begin{aligned} \frac{du_{i,j}}{dt} &= D\frac{u_{i,j+1}(t) - u_{i,j}(t)}{h_y^2} - w^y_{i,j+\frac{1}{2}} \frac{u_{i,j}(t) + u_{i,j+1}(t)}{2h_y} + D\frac{u_{i+1,j}(t) - u_{i,j}(t)}{h_x^2} \\ &\quad - w^x_{i+\frac{1}{2},j} \frac{u_{i,j}(t) + u_{i+1,j}(t)}{2h_x} - D\frac{u_{i,j}(t) - u_{i,j-1}(t)}{h_y^2} + w^y_{i,j-\frac{1}{2}} \frac{u_{i,j}(t) + u_{i,j-1}(t)}{2h_y} \\ &\quad - D\frac{u_{i,j}(t) - u_{i-1,j}(t)}{h_x^2} + w^x_{i-\frac{1}{2},j} \frac{u_{i,j}(t) + u_{i-1,j}(t)}{2h_x} - \alpha u_{i,j} + f_{i,j}. \end{aligned}$$

Which is equivalent to:

$$\begin{aligned} \frac{du_{i,j}}{dt} &= u_{i,j-1}\left(\frac{D}{h_y^2} + \frac{w^y_{i,j-\frac{1}{2}}}{2h_y}\right) + u_{i-1,j}\left(\frac{D}{h_x^2} + \frac{w^x_{i-\frac{1}{2},j}}{2h_x}\right) \\ &\quad + u_{i,j}\left(-\frac{2D}{h_x^2} - \frac{2D}{h_y^2} + \left[\frac{w^y_{i,j-\frac{1}{2}}}{2h_y} + \frac{w^x_{i-\frac{1}{2},j}}{2h_x} - \frac{w^x_{i+\frac{1}{2},j}}{2h_x} - \frac{w^y_{i,j+\frac{1}{2}}}{2h_y}\right] - \alpha\right) \\ &\quad + u_{i+1,j}\left(\frac{D}{h_x^2} - \frac{w^x_{i+\frac{1}{2},j}}{2h_x}\right) + u_{i,j+1}\left(\frac{D}{h_y^2} - \frac{w^y_{i,j+\frac{1}{2}}}{2h_y}\right) + f_{i,j}. \end{aligned} \tag{3}$$

The discretisation of the boundary points is a simple substitution of the boundary condition. This results in: $u_{0,j} = 0$, $u_{101,j} = 0$, $u_{i,0} = 0$ and $u_{i,101} = 0$.
The obtained discretisation can be written as a system of equations in the following fashion:

$$\frac{d\mathbf{u}}{dt} = A(t)\mathbf{u} + \mathbf{f}(t) \tag{4}$$

To create the discretisation matrix, a horizontal numbering is used. Each point with coordinates (x_i, y_j) corresponds to row number $n = i + (j-1)N$. The total matrix will be a $N^2 \times N^2$ block matrix, which has the form:

$$A = \begin{bmatrix} L_1 & L_2 & & & \\ L_3 & L_1 & L_2 & & \\ & \ddots & \ddots & \ddots & \\ & & L_3 & L_1 & L_2 \\ & & & L_3 & L_1 \end{bmatrix}$$

A is a block-diagonal matrix that consists of $N \times N$ block-matrices L. To describe the matrices L_1, L_2 and L_3, the discretisation will be rewritten in a more compact form using the following discretisation:

$$\frac{du_{i,j}}{dt} = u_{i,j-1}\,(a) + u_{i-1,j}\,(b) + u_{i,j}\,(e) + u_{i+1,j}\,(c) + u_{i,j+1}\,(d) + f_{i,j}.$$

Then L can be written as:

$$L_1 = \begin{bmatrix} e & c & & & \\ b & e & c & & \\ & \ddots & \ddots & \ddots & \\ & & b & e & c \\ & & & b & e \end{bmatrix}, L_2 = \begin{bmatrix} d & & \\ & \ddots & \\ & & d \end{bmatrix}, L_3 = \begin{bmatrix} a & & \\ & \ddots & \\ & & a \end{bmatrix}$$

In (4) the vector **u** contains all points $u_{i,j}$ with $i, j \in \{1, \ldots, N\}$, ordered using the same horizontal numbering as for the discretisation matrix. The vector **f** consists of all values $f_{i,j}$ $i, j \in \{1, \ldots, N\}$ of the source function. Again, the same horizontal numbering is applied.

4.2.1 Analysis of the discretisation matrix

For the analysis of the stability of the system and later for the stability of the time discretisation method, it is important to analyse the matrix A. It can first be noted that the matrix A is not symmetric. The next thing that will be analysed of the matrix A are the eigenvalues. The eigenvalues can be closer investigated by using Gershgorin's circle theorem: every eigenvalue λ of matrix A satisfies:

$$|\lambda - a_{i,i}| \leq \sum_{j=1, j\neq i}^{N} |a_{i,j}|.$$

This theorem will give an area in the complex plane in which the eigenvalues can be found. This area is the union of as many discs as the amount of rows matrix A has. Applying Gershgorin's theorem to the obtained matrix A from Section 4.2, gives five different types of rows. These rows are the radii of the discs, to find the smallest eigenvalue, the largest radius will be taken. This leads to the following interval:

$$|\lambda - e| \leq |a| + |b| + |c| + |d|. \tag{5}$$

The found interval represents a circle with center: e and radius: $|a|+|b|+|c|+|d|$. The smallest eigenvalue will be on the most left point of this disc. This implies that the minimal eigenvalue is located on the real axis of the imaginary plane. Which in turn gives that the minimal eigenvalue is real. For the minimal eigenvalue, the eigenvalue will be bounded first, after which this will be minimised numerically using the received wind data. This means that for the smallest eigenvalue, the expression can be written out and

bounded by using the triangle inequality.

$$
\begin{aligned}
\lambda_{\min} = \min_{i,j} \Bigg[& -\frac{2D}{h_x^2} - \frac{2D}{h_y^2} + \frac{w^y_{i,j-\frac{1}{2}}}{2h_y} + \frac{w^x_{i-\frac{1}{2},j}}{2h_x} - \frac{w^x_{i+\frac{1}{2},j}}{2h_x} - \frac{w^y_{i,j+\frac{1}{2}}}{2h_y} - \alpha \\
& - \left| \frac{D}{h_y^2} + \frac{w^y_{i,j-\frac{1}{2}}}{2h_y} \right| - \left| \frac{D}{h_x^2} + \frac{w^x_{i-\frac{1}{2},j}}{2h_x} \right| - \left| \frac{D}{h_x^2} - \frac{w^x_{i+\frac{1}{2},j}}{2h_x} \right| - \left| \frac{D}{h_y^2} - \frac{w^y_{i,j+\frac{1}{2}}}{2h_y} \right| \Bigg] \\
\leq \min_{i,j} \Bigg[& -\frac{2D}{h_x^2} - \frac{2D}{h_y^2} + \frac{w^y_{i,j-\frac{1}{2}}}{2h_y} + \frac{w^x_{i-\frac{1}{2},j}}{2h_x} - \frac{w^x_{i+\frac{1}{2},j}}{2h_x} - \frac{w^y_{i,j+\frac{1}{2}}}{2h_y} - \alpha \\
& - \left| \frac{D}{h_y^2} \right| - \left| \frac{w^y_{i,j-\frac{1}{2}}}{2h_y} \right| - \left| \frac{D}{h_x^2} \right| - \left| \frac{w^x_{i-\frac{1}{2},j}}{2h_x} \right| - \left| \frac{D}{h_x^2} \right| - \left| -\frac{w^x_{i+\frac{1}{2},j}}{2h_x} \right| - \left| \frac{D}{h_y^2} \right| - \left| -\frac{w^y_{i,j+\frac{1}{2}}}{2h_y} \right| \Bigg] \\
\leq \min_{i,j} \Bigg[& -\frac{4D}{h_x^2} - \frac{4D}{h_y^2} + \frac{w^y_{i,j-\frac{1}{2}} - w^y_{i,j+\frac{1}{2}}}{2h_y} + \frac{w^x_{i-\frac{1}{2},j} - w^x_{i+\frac{1}{2},j}}{2h_x} - \frac{\left|w^x_{i-\frac{1}{2},j}\right| + \left|w^x_{i+\frac{1}{2},j}\right|}{2h_x} - \frac{\left|w^y_{i,j-\frac{1}{2}}\right| + \left|w^y_{i,j+\frac{1}{2}}\right|}{2h_y} - \alpha \Bigg]
\end{aligned}
\tag{6}
$$

This means that there is a difference in the smallest eigenvalue depending on the direction and speed of the wind, the value for D and the refinement of the discretisation grid. The minimal eigenvalue can be determined for each value of D and N_x, N_y. For a refinement of $N_y \times N_x = 100 \times 100$, the minimal eigenvalues are set out in Table 1. The values are numerically determined on the basis of the received wind data.

Diffusion coefficient	Month	$\lambda_{\min}$
$D = 0.001$	June	-38.97
	July	-24.22
	August	-26.96
$D = 0.005$	June	-66.51
	July	-51.75
	August	-54.50
$D = 0.01$	June	-100.93
	July	-86.17
	August	-88.91

Table 1: Lower bound on the minimal eigenvalues per month, with $\alpha = 0$ and $N_y \times N_x = 100 \times 100$ in 2019.

The value for the minimal eigenvalue also depends on the refinement of the grid, not only does a more refined grid result in a smaller value for h_x, h_y. It also results in different values for the wind, since a more refined grid results in smaller gradients. If this difference in the wind is assumed to be minimal and solely h_x, h_y is decreased by the refinement, the minimal eigenvalues for different grid sizes can be approximated. These approximated values are set out in Table 2. It is clear that the refinement of the grid has a big impact on the value of the smallest eigenvalue.

Diffusion coefficient	Month	$\lambda_{\min}$
$D = 0.001$	June	-92.32
	July	-62.65
	August	-68.17
$D = 0.005$	June	-203.58
	July	-173.91
	August	-179.43
$D = 0.01$	June	-342.65
	July	-312.99
	August	-318.51

Table 2: Approximation for the lower bound of the minimal eigenvalues per month, with $\alpha = 0$ and $N_y \times N_x = 200 \times 200$ in 2019.

The found values for the minimal eigenvalue will later be used to analyse the stability of time discretisation methods.

4.3 Numerical time integration

Next, the convection-diffusion equation will be integrated in time. This can be done using various time-integration techniques, starting with the Forward and Backward Euler methods. These two methods will play an important role in the reconstruction of the source function in Section 5. In what follows we denote $\mathbf{u}_n = \mathbf{u}(t_n)$, $\mathbf{f}_n = \mathbf{f}(t_n)$ and $A_n = A(t_n)$. The Forward Euler (FE) method uses the following single-point approximation:

$$\begin{aligned} \frac{d\mathbf{u}}{dt} &= A(t)\,\mathbf{u} + \mathbf{f}, \\ \int_{t_n}^{t_{n+1}} \frac{d\mathbf{u}}{dt}\,dt &= \int_{t_n}^{t_{n+1}} A(t)\,\mathbf{u}\,dt + \int_{t_n}^{t_{n+1}} \mathbf{f}\,dt \\ \mathbf{u}_{n+1} - \mathbf{u}_n &= \Delta t A_n \mathbf{u}_n + \Delta t \mathbf{f}_n + \mathcal{O}((\Delta t)^2), \\ \mathbf{u}_{n+1} &\approx [I + \Delta t A_n]\mathbf{u}_n + \Delta t \mathbf{f}_n \end{aligned} \tag{7}$$

The Backward Euler (BE) method uses a different single-point approximation:

$$\begin{aligned} \int_{t_n}^{t_{n+1}} \frac{d\mathbf{u}}{dt}\,dt &= \int_{t_n}^{t_{n+1}} A(t)\,\mathbf{u}\,dt + \int_{t_n}^{t_{n+1}} \mathbf{f}\,dt \\ \mathbf{u}_{n+1} - \mathbf{u}_n &= \Delta t A_{n+1} \mathbf{u}_{n+1} + \Delta t \mathbf{f}_{n+1} + \mathcal{O}((\Delta t)^2), \\ \mathbf{u}_{n+1} &\approx \mathbf{u}_n + \Delta t A_{n+1} \mathbf{u}_{n+1} + \Delta t \mathbf{f}_{n+1} \\ [I - \Delta t A_{n+1}]\mathbf{u}_{n+1} &\approx \mathbf{u}_n + \Delta t \mathbf{f}_{n+1}, \end{aligned} \tag{8}$$

where a linear system must be solved at each time step to find $\mathbf{u}_{n+1}$, which means that the Backward Euler method results in an implicit time-stepping scheme. The benefit of the BE method with respect to the much simpler FE method is in the unconditional stability of the resulting time-stepping scheme.

There exist higher-order time-integration techniques, such as the Runge-Kutta method, which use multi-point approximations (quadratures) for the time integrals in the right-hand side. These methods also result in implicit time-stepping schemes. Take, for example, the two-point Trapezoidal method, where the integral is approximated using the values of the integrand at both ends of the interval $[t_n, t_{n+1}]$:

$$\begin{aligned} \int_{t_n}^{t_{n+1}} \frac{d\mathbf{u}}{dt}\,dt &= \int_{t_n}^{t_{n+1}} A(t)\,\mathbf{u}\,dt + \int_{t_n}^{t_{n+1}} \mathbf{f}\,dt \\ \mathbf{u}_{n+1} - \mathbf{u}_n &= \Delta t \left[\frac{1}{2} A_n \mathbf{u}_n + \frac{1}{2} A_{n+1}\mathbf{u}_{n+1}\right] + \Delta t \left[\frac{1}{2}\mathbf{f}_n + \frac{1}{2}\mathbf{f}_{n+1}\right] + \mathcal{O}((\Delta t)^3), \\ \left[I - \frac{\Delta t}{2} A_{n+1}\right] \mathbf{u}_{n+1} &\approx \left[I + \frac{\Delta t}{2} A_{n+1}\right] \mathbf{u}_n + \frac{\Delta t}{2}\left[\mathbf{f}_n + \mathbf{f}_{n+1}\right]. \end{aligned} \tag{9}$$

As can be seen, here too a linear system needs to be solved at every time step. This method has a higher numerical precision, compared to both the FE and the BE methods, and is unconditionally stable.

4.3.1 Stability of the Forward-Euler method

Consider the source-free problem and the FE time-stepping scheme:

$$\begin{aligned} \mathbf{u}_{n+1} &= [I + \Delta t A_n]\mathbf{u}_n, \\ \mathbf{u}_{n+1} &= \prod_{i=0}^{n} [I + \Delta t A_i]\mathbf{u}_0 \leq [I + \Delta t A_j]^n \mathbf{u}_0, \end{aligned} \tag{10}$$

where A_j is the matrix of the same form as A_i, but with the time-independent wind data, such that the inequality above holds (the worst case wind). It is not clear how to construct the upper-bound matrix A_j given the matrices A_i, $i = 1, \ldots, n$ in the general case. However, this analysis can easily be applied in the situation where the direction of the wind remains the same and only its amplitude changes with time.

The solution is expected to decay with n, i.e., $\lim_{n\to\infty} \mathbf{u}_{n+1} = \mathbf{0}$. It is well-known that for any $A \in \mathbb{C}^{k\times k}$, $\lim_{k\to\infty} A^k = 0$, if and only if $\rho(A) < 1$, where $\rho(A)$ is the spectral radius of A, i.e., $\rho(A) = \max\{|\lambda_1|, \ldots, |\lambda_k|\}$, where λ_i are the eigenvalues of A.

Therefore, the FE method is stable if

$$\begin{aligned}
&\max_i |\tilde{\lambda}_i| < 1,\\
&|\tilde{\lambda}_i| < 1, \quad \forall\, i,\\
&|\tilde{\lambda}_i|^2 < 1, \quad \forall\, i,\\
&|1 + \Delta t\lambda_i|^2 < 1, \quad \forall\, i,\\
&\left(1 + \Delta t\,\mathrm{Re}(\lambda_i)\right)^2 + \left(\Delta t\,\mathrm{Im}(\lambda_i)\right)^2 < 1, \quad \forall\, i,\\
&0 < \Delta t < -\frac{2\mathrm{Re}(\lambda_i)}{\mathrm{Re}(\lambda_i)^2 + \mathrm{Im}(\lambda_i)^2} \leq -\frac{2\mathrm{Re}(\lambda_i)}{\mathrm{Re}(\lambda_i)^2}, \quad \forall\, i,\\
&0 < \Delta t < -\frac{2}{\min_i\, \mathrm{Re}(\lambda_i)},
\end{aligned} \tag{11}$$

where λ_i are the eigenvalues of the (worst-case) matrix A_j and $\tilde{\lambda}_i = 1 + \Delta t\lambda_i$ are the eigenvalues of the matrix $I + \Delta t A_j$. To calculate the bound $\min_i \mathrm{Re}(\lambda_i)$ we use the results of the Section 4.2.1.

It was previously found that the smallest eigenvalue depends on the diffusion coefficient, the wind direction and amplitude, the parameter α and the refinement of the grid. The found values for the minimal eigenvalue, set out in Table1, will be used to give constraints for the stability.

$$\begin{aligned}
\Delta t < -\frac{2}{\lambda_{\min}}&\\
\Rightarrow \quad \Delta t \leq \frac{-2}{-38.97} = 0.05 \quad &\text{for } \alpha = 0,\ D = 1\times 10^{-3} \text{ and } N_x = N_y = 100\\
\Rightarrow \quad \Delta t \leq \frac{-2}{-66.51} = 0.03 \quad &\text{for } \alpha = 0,\ D = 5\times 10^{-3} \text{ and } N_x = N_y = 100\\
\Rightarrow \quad \Delta t \leq \frac{-2}{-100.93} = 0.02 \quad &\text{for } \alpha = 0, D = 1\times 10^{-2} \text{ and } N_x = N_y = 100
\end{aligned} \tag{12}$$

From these calculations it becomes clear that as the diffusion coefficient increases, the Δt has to decrease for the method to be stable. The term α is above taken to be zero, as this parameter will take value in the order 10^0 or lower, the influence of the time constraint is minimal. The influence of the refinement of the discretisation does however have a large impact. Using the values for the minimal eigenvalues from Table 2, the stability criterion can be determined for a refinement of $N_y \times N_x = 200\times 200$. In that case, the following criteria are found:

$$\begin{aligned}
\Delta t \leq \frac{-2}{-92.32} = 0.022 \quad &\text{for } \alpha = 0,\ D = 1\times 10^{-3} \text{ and } N_x = N_y = 200\\
\Delta t \leq \frac{-2}{-203.58} = 0.0098 \quad &\text{for } \alpha = 0,\ D = 5\times 10^{-3} \text{ and } N_x = N_y = 200\\
\Delta t \leq \frac{-2}{-342.65} = 0.0058 \quad &\text{for } \alpha = 0, D = 1\times 10^{-2} \text{ and } N_x = N_y = 200
\end{aligned} \tag{13}$$

These calculations will be repeated outside of the report for the grid sizes that will be used.

4.4 Numerical effects due to convection

The problem defined in Equation (1) is a singularly perturbed problem. This is because the following is true: $|\mathbf{w}| >> D$, which makes the problem a convection dominated problem. To measure how much the convection dominates the diffusion, the Péclet number is used: $P_e = \frac{|\mathbf{w}|L}{D}$, where L is the length of the domain. For a large Péclet number a boundary layer will occur which will cause problems in the numerical treatment [11]. If there is a large gradient, the solution can start to oscillate. Specifically, for the mesh Péclet number: p_h, the condition $p_h \leq 1$ is necessary for a monotone solution. This gives the following condition for the step size: $\frac{h}{L} \leq \frac{2}{|P_e|}$. Note that there is a different Péclet number for the x- and y-direction, P_e^x, P_e^y. The Péclet number depends on the diffusion coefficient D, since it was determined that this coefficient ranges between $1 \times 10^{-4} \leq D \leq 1 \times 10^{-2}$, the Péclet numbers for multiple values for D are calculated. The calculated values can be found in Table 3.

D	P_e^x	P_e^y	$h_x = h_y$	n_x	n_y
1×10^{-4}	27300	18850	0.0003	14000	9670
5×10^{-4}	5460	3770	0.0015	2800	1940
1×10^{-3}	2730	1885	0.003	1400	967
5×10^{-3}	546	377	0.015	280	194
1×10^{-2}	273	188.5	0.03	140	97

Table 3: The Péclet number for different D, based on the maximum amplitude of the wind on 31 August 2019 of approximately 12 m/s.

As can be seen in Table 3, the Péclet numbers for both the x- and y-direction are large for each D, which confirms that the problem is a convection dominated problem. To make sure the mesh Péclet number is below one, which makes the solution monotone, the mesh can be refined. For $D = 1 \times 10^{-4}$ a step size of less than 0.0003 is needed, whereas for $D = 5 \times 10^{-3}$ a step size of 0.015 is needed. These step sizes hold for both the x- and y-direction.
From the values for n_x and n_y, that can be seen in Table 3, it is clear that a the diffusion coefficient grows, the number of grid points needed decreases. The refinements of the grid, where $N_x \geq 1400$, will cost too much computing power for a laptop with 16GB of memory. Therefore, in this report, the largest refinement of the grid that will be worked with is: 194×280. For future research this grid size could be refined. If the refinement of the grid shows to need to much memory, other solutions can be considered. For example, the refinement can be only applied to the boundary layer or upwind differencing can be used for the discretisation. The latter still has a problem, since the direction of the wind changes over time, the variable $\mathbf{w}$ can be either positive or negative. This means that for each individual point it needs to be determined whether forward or backward differencing is needed.

4.5 Estimation of transport parameters

To find the discretisation parameters, artificial data was created, where the source function is known. This is done by first defining a source function, in the first instance with a couple of point sources. For a second set of experiments extended sources are considered. The point sources that were chosen are based on the data of the NO_2 concentration on the 24th of August 2019. The wind data that was used is also based on this date. To be precise, the wind data that is used is from the 24 hours before the data measurement on the 24th of August 2019, thus from 12:50 23/08/2019. In Figure 7 the NO_2 snapshot is shown as well as the locations that are be used for the first test source function. These are found by choosing the point with the highest concentration of NO_2 for three different plumes. This resulted in the following coordinates: (4.01,51.99), (4.51,52.43) and (6.59,51.55). Approximately located in Ijmuiden, Rotterdam and the Ruhrgebiet.

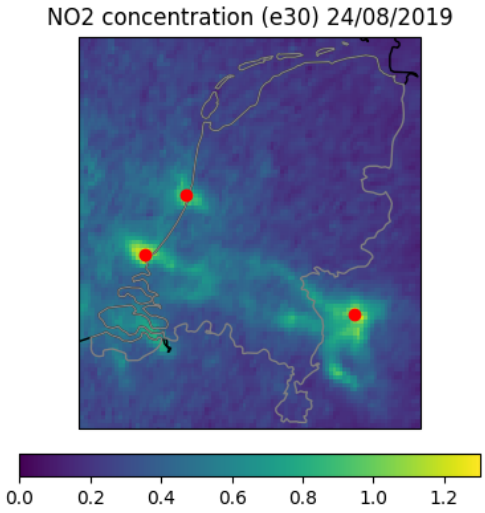


Figure 7: Three chosen locations for sources in the test source function.

The found coordinates are used as (x_s, y_s) for the artificial source function (14). This source function is used to describe point sources:

$$f(t) = \sum_{i=1}^{N_s} q_s \delta(x - x_s)\delta(y - y_s), \tag{14}$$

where N_s is the number of point sources, q_s is the strength of each source and (x_s, y_s) are the coordinates of the point sources. In this equation δ stands for the Dirac delta function. In the system of equations this will result in a vector $\mathbf{f}$ with q_s for each point source and zero otherwise.
For each different source a different strength can be used, but are assumed to be constant over time. First the diffusion coefficient and the amplitudes of the sources are analysed, by showing the effect of different values. Afterwards, the influence of the decay term: αu is be investigated. The goal of these analyses is to find a set of parameters that best simulate the data given. Therefore, the goal for this section is to find a set of parameters such that the solution of the discretisation after 24 hours resembles the data as closely as possible. To create the artificial data, the wind data of 23/08/2019 12:50 until 24/08/2019 12:50 is used. That means that the objective is that the resulting data after 24 hours resembles the data in Figure 7 as closely as possible.
Looking at Figure 7, there are two different hypotheses that can be made on how the parameters should be set. The first hypothesis is that the NO_2 moves mostly by convection and the diffusion is near zero. This means that the clouds become curved and elongated by the wind. Thus that means that the pollution comes from relatively small point sources and get transported by the wind. To test this hypothesis, the point sources as discussed in the beginning of this section are be used. Using that, the parameters: D and α are varied in order to find a combination that best describes the data.

The second hypothesis is: that the convection dominates the diffusion less, that the curved NO_2 clouds are more due to the location of the sources and less because of the wind. To test this hypothesis, a map is created with known industrial areas and highways, where emission of NO_2 takes place. This map serves as the source function, which does not consist of point sources, but of extend sources.

It is important to note that for the first hypothesis the size of the grid is $N_x \times N_y = 194 \times 280$. For the second, the grid size is taken to be $N_x \times N_y = 100 \times 100$. This means that for some combinations of parameters the mesh Péclet number is larger than one and thus there can be discretisation errors present, see Section 4.4. The goal of this section is, however, to asses the influence of the magnitudes of the parameters. This influence is still visible when discretisation errors are present. Therefore to make the determination of parameters based on the propagation of artificial data time efficient, the grid size remains 194×280 and 100×100 throughout the section.

4.5.1 Hypothesis one: localized sources

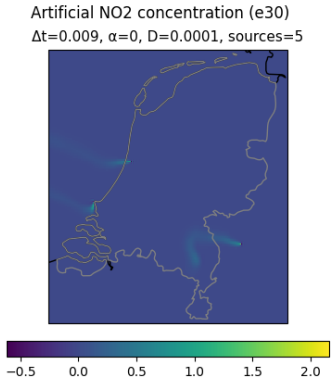
To begin, the hypothesis that the sources are point sources is tested. For this hypothesis, first the diffusion coefficient is studied, after which the influence of the decay term is analysed. Finally there is also a brief discussion on the case that there are multiple sources in close proximity. For each of these experiments, the Forward Euler method is used to propagate in time. The time steps is set to $\Delta t = 0.009$, as this was found to be stable for the Forward Euler method (12).

Diffusion coefficient D: As previously written, the first parameter that will be analysed is the diffusion coefficient. To do this, first the amplitude of the source functions is set. Each source can be given an individual emission rate, which is the amplitude of the source. To be precise, the sources will all have a source strength of 5. This number is given in $e10^{30}$ mls/deg^2 h, and is purely an estimation. In addition, we have that: $\alpha = 0$. The grid used for the tests in this section has a size of 194×280 interior points. This means that according to the Péclet number there are numerical errors present in the solution for $D < 5 \times 10^{-3}$. This expresses itself in negative values in the solution, or oscillations.
In Figure 8, the influence of the magnitude of the diffusion coefficient is shown. Comparing the four Figures 8a, 8b,8c and 8d shows that for a larger diffusion coefficient the clouds become less defined. The NO_2 'plumes' that are visible in Figure 7 have clear edges, with a large gradient on the edge. From the test, this can be seen for the smallest value of the diffusion coefficient D. As the D grows, the edges of the plumes become less defined and more saturated.
As well as on the diffusion coefficient, the figures give information on the influence of the source function. In Figure 7, the maximum NO_2 concentration is around 1.3×10^{30}mls/deg^2. From Figure 8 it can be noted that as the diffusion coefficient grows larger, the maximal concentration of NO_2 lowers. This maximum is located at the source for all four sub figures. So as the diffusion coefficient decreases, the pollution plumes get more defined, but the concentration of NO_2 at the point source gets larger.
Based solely on this experiment it is not yet possible to define an emission rate for the sources. For this, also the influence of the decay term α will be studied.

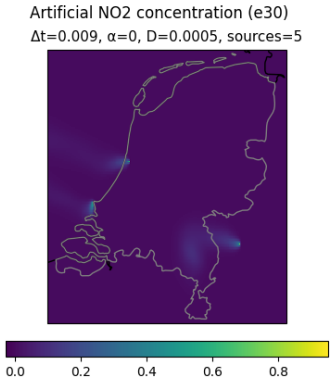
Decay term α: Next, the influence of the decay term α is analysed. Again, the other parameters: $\Delta t, D$ and the strength of the sources are set, whilst the parameter α is varied. The diffusion coefficient is taken to be $D = 0.0005$, the sources have a strength of 5 and the time step remains $\Delta t = 0.009$ The result of this experiment can be seen in Figure 9. There are two main things that stand out, firstly it is clear from the figures that as the α grows, the plumes start to fade. For $\alpha = 0.15$ in Figure 9d the plumes seem to have disappeared, there are only small plumes left around the sources. However, the maximum NO_2 concentration does not change for the varying α.

Multiple point sources: Finally, for the first hypothesis, the influence of multiple sources close to each other is investigated. As an example, Figure 9a and Figure 10 can be compared. The only difference between these two figures is that, for the second figure two sources are added. For that figure there are three sources in close proximity. The distance between the sources is between 4-11 points, which comes down to a maximum distance of less than 18 km between two sources. The comparison gives that adding multiple sources in close proximity leads to a bigger plume with a higher NO_2 concentration. Note that the maximum NO_2 concentration does not change significantly.

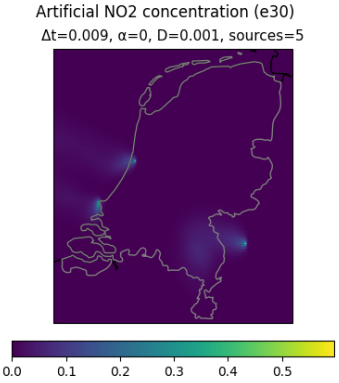
Conclusion: On the basis of the first hypothesis: that the pollution is emitted by sparsely located point sources, conclusions can be made on the parameters that best replicate experimental data. For these conclusions, Figure 7 is used as a reference. The plumes that are sought after are ones with a high gradient on the edge, thus with well defined edges. This appears in the solution for a diffusion coefficient $D \leq 5 \times 10^{-4}$ and a decay parameter of $\alpha = 0$. In addition, the case where multiple sources were placed in close proximity looks more closely to the actual data, than very sparsely place point sources. Therefore,



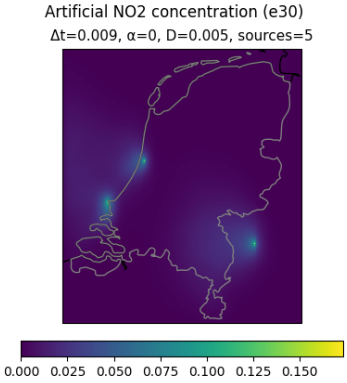
(a) $D = 1 \times 10^{-4}$



(b) $D = 5 \times 10^{-4}$



(c) $D = 1 \times 10^{-3}$



(d) $D = 5 \times 10^{-3}$

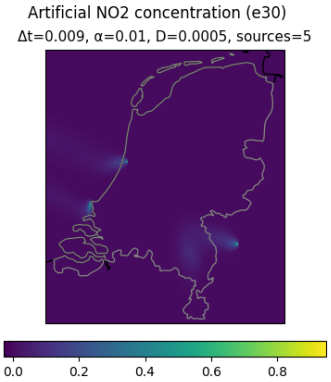
Figure 8: The influence of the diffusion coefficient D. The negative values present in the Figures are due to a mesh Péclet number larger than one.

according to the findings of this section there is little diffusion in comparison to convection and there is zero, or close to zero, decay in the time span of 24 hours. As well as there being multiple distinct sources close to each other, which can create large plumes.

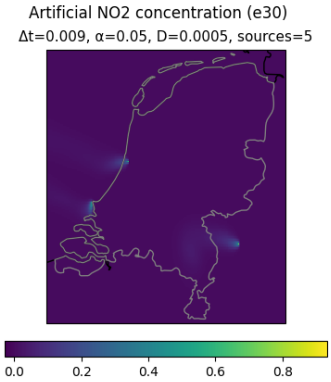
4.5.2 Hypothesis two: extended sources

From the first hypothesis it was found that there are probably multiple sources close to each other in the experimental data. Since the pollution plumes are large and defined in the data, but the maximum NO_2 concentration is fairly low. It can also be the case that in addition to point sources, there are different types of sources such as line sources. This could be due to highways, where cars produce NO_2, or waterways where cargo ships emit it. To try and replicate a possible map of sources, the map of the Netherlands can be used. On the map all of the mayor highways and known industrial zones are pinned as sources.

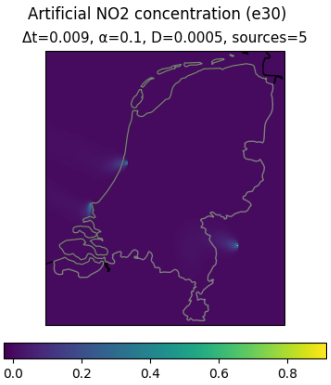
This map can be seen in Figure 11, as well as an interpolated version with 100×100 pixels that will be used for the tests. On both figures the roads and the industrial zones are distinguishable. For the next test, this image will be taken as the source function. For the roads and the industrial zones, the strength of the emission can be set manually.
To test what values the parameters: D and α should take, the strength of the sources will be set in the following way. The industrial zones(the yellow zones in Figure 11) will have a strength of 2. The roads (coloured green) will have an emission rate of 0.4. The resulting figures can be found in Appendix C, in Figure 33. From this figure it can first be noted that for $D = 0.01$ the influence of α is almost not noticeable. In addition it can be noted that as the α grows, the roads begin to become visible



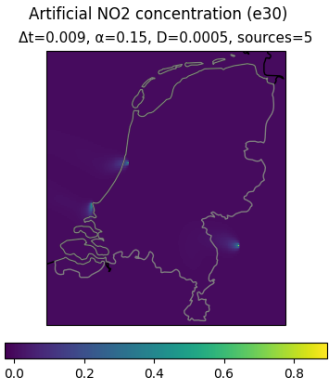
(a) $\alpha = 0.01$



(b) $\alpha = 0.05$



(c) $\alpha = 0.1$



(d) $\alpha = 0.15$

Figure 9: The influence of the decay coefficient α.

in the images. Since in Figure 7, the roads are not distinguishable, the α will be taken such that in the experiment the roads are not visible either. Which roughly means: $\alpha < 0.5$. What also stands out from Figure 7 is that between the 'Ruhrgebiet' and Rotterdam, so over Brabant and Limburg, the concentration of NO_2 is on the high end. In the experiment, the parameters for which this comes closest is for $0.001 \leq D \leq 0.005, \alpha = 0.1$.
For $D < 0.005$, there will be numerical errors in the solution for grids with a refinement of less than 194×280 interior points, see Table 3. Therefore, by combining the usability and performance of the parameters, the following parameters are taken: $D = 0.005, \alpha = 0.1$.

4.5.3 Conclusion on found parameters

From the first hypothesis, where the pollution is taught to mainly come from point sources, the parameters that looked best were: $D = 5 \times 10^{-4}$ and $\alpha = 0$. Whereas from the second hypothesis the found parameters were: $D = 5 \times 10^{-3}, \alpha = 0.1$. In Table 3 it can be seen that for a diffusion coefficient of $D = 5 \times 10^{-4}$, the grid will need to be refined to have a size of 2800 by 1940 interior points. This refinement is too large for a laptop with 16 GB of RAM. For $D = 5 \times 10^{-3}$, the refinement needed is: 280×194 . This is achievable for a laptop with 16 GB of RAM. Therefore, the diffusion coefficient found in the second hypothesis will be taken to be used as a basis for the next experiments. This leads to set of parameters that can be found in Table 4.

The Δt in Table 4, is found by using the same technique as in Section 4.3.1.

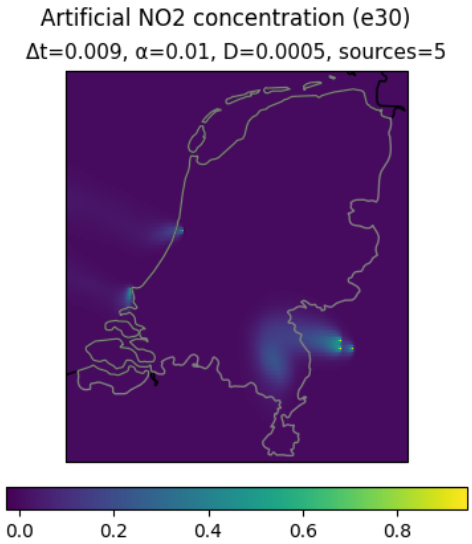
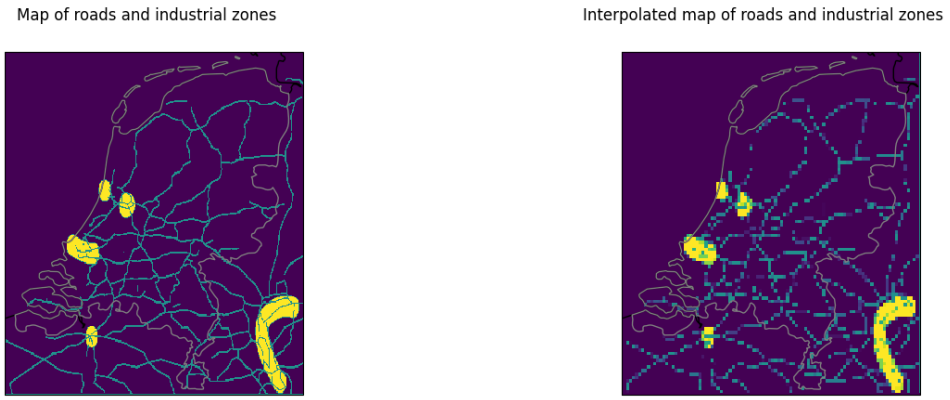


Figure 10: Multiple sources close to each other.



(a) Version with 343 by 498 pixels.

(b) Interpolated version with 100 by 100 pixels.

Figure 11: Maps of the highways and industrial zones of the domain in different resolutions.

5 Inverse source problem

In the problem there are two terms left undefined. Namely the sink term αu and the source function f. First, the source term is estimated. Based on that and the available data, the sink term is determined, this is done in another section. To find an approximation for the source function f a large assumption is made. The assumption that is made is that each source has a constant emission rate, which in turn makes the source function a constant function. This gives that $f^n = f$ holds for each n. Reconstructed source functions are represented by a vector as well, this vector is plotted to show visualise the reconstructed sources.

5.1 Existing methods for determining the source function

The determination of sources in a convection-diffusion problem is important for the prediction of the dispersion of pollutants. Not only for the spread of pollutants in the atmosphere, but also for pollutants in water or even the spread of an infectious disease. Thus there is a collection of research done on the topic. In this section some of these methods will be discussed, before the algorithm is introduced which will be used in this report.

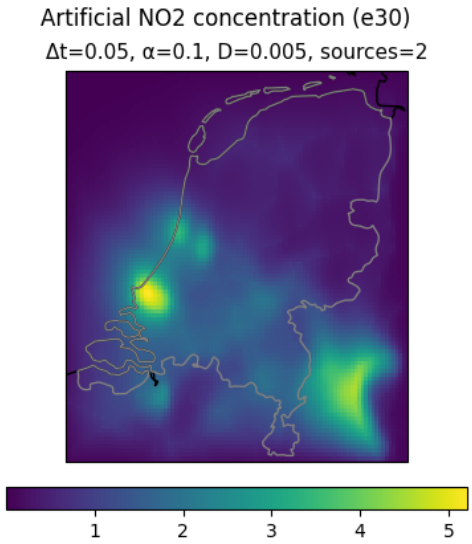


Figure 12: Simulated NO_2 concentrations based on extended sources, using parameters that best emulate experimental data.

Parameters:	D	Δt	α	$n_y \times n_x$	Source	t_0
Values:	5×10^{-3}	0.009	0.1	194×280	5	23/08/2019 12:50

Table 4: Parameters for a stable discretisation.

To start, it is important to note that there are different approaches to this inverse problem of finding sources. Two distinct main ideas are: first, using inverse numerical methods, by solving the governing equations backward in time. The second approach is using probabilistic techniques to find the probability of the location of the sources [3]. One method for numerical computations backwards in time is the Backward Beam Equation (BBE) method [4]. Another method to find the locations of sources is by writing the inverse problem as a constrained, regularized, least squares optimization problem [2]. From the probabilistic approaches, using machine learning is an example. Ensemble learning methods can be used to identify pollution sources [20].
lastly there can also be made a combination of both direct numerical inverse problems and probabilistic techniques. In [21] a method is introduced that is a combination of inverse methods and machine learning algorithms. Using the Green's function of the convection-diffusion equation and a Blind Source Separation (BSS) method in combination with a semi-supervised clustering algorithm.

5.2 Source reconstruction based on the Forward Euler method

As written in the previous section, there are multiple approaches for reconstructing source functions. For this report, inverse numerical methods will be used. To find the source function $f(x, y)$, the so-called model-based inverse problem approach, is taken. This is a method where a mathematical model of the measured data is inverted to reconstruct the source of the data. The mathematical model of the NO_2 data consists of the boundary-value problem (1), the Finite-Volume spatial discretisation, and the chosen time-integration method. In the first case, Forward Euler was taken as the time integration method. The discretised source function $\mathbf{f}$ is a part of this model, and thus the task is to find $\mathbf{f}$ assuming that the two snapshots of the NO_2 data and the corresponding intermediate wind data are given.

Starting with a simple case where the given NO_2 snapshots correspond to the two consecutive time steps of the Forward Euler method. The *forward* problem is the one of finding the snapshot $\mathbf{u}_{n+1}$ given the snapshot $\mathbf{u}_n$, and all other parameters. The explicit solution of this forward problem is:

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t A_n \mathbf{u}_n + \Delta t \mathbf{f}. \tag{15}$$

The *inverse* problem with respect to this model is considered to be the one of finding $\mathbf{f}$, given the snapshots $\mathbf{u}_n$,$\mathbf{u}_{n+1}$ and the wind data. This inverse problem can be solved explicitly as:

$$\mathbf{f} = \frac{1}{\Delta t}\left(\mathbf{u}_{n+1} - \mathbf{u}_n\right) - A_n\mathbf{u}_n.$$

Hence, when both $\mathbf{u}_n$ and $\mathbf{u}_{n+1}$ are known the source function can be reconstructed immediately. Unfortunately, in practice, this is not the case, since the current satellite provides the NO_2 data only roughly every 24 hours. This is a large time interval, that cannot be bridged by one time step of a time-integration method.

To find an equation which can determine the source function from time-sparse NO_2 data and time-dense wind data, the Forward Euler method is written out for multiple time steps. The updates with the time-varying matrix A_n and a time-invariant source vector $\mathbf{f}$ are:

$$\begin{aligned}
\mathbf{u}_{n+1} &= [I + \Delta t A_n]\mathbf{u}_n + \Delta t\mathbf{f},\\
\mathbf{u}_{n+2} &= [I + \Delta t A_{n+1}]\mathbf{u}_{n+1} + \Delta t\mathbf{f}\\
&= [I + \Delta t A_{n+1}][I + \Delta t A_n]\mathbf{u}_n + [I + \Delta t A_{n+1}]\Delta t\mathbf{f} + \Delta t\mathbf{f},\\
\mathbf{u}_{n+3} &= [I + \Delta t A_{n+2}]\mathbf{u}_{n+2} + \Delta t\mathbf{f}\\
&= [I + \Delta t A_{n+2}][I + \Delta t A_{n+1}][I + \Delta t A_n]\mathbf{u}_n\\
&\quad + [I + \Delta t A_{n+2}][I + \Delta t A_{n+1}]\Delta t\mathbf{f} + [I + \Delta t A_{n+2}]\Delta t\mathbf{f} + \Delta t\mathbf{f},\\
&\quad \dots\\
\mathbf{u}_{n+m} &= \prod_{i=0}^{m-1}[I + \Delta t A_{n+i}]\mathbf{u}_n + \sum_{i=1}^{m-1}\prod_{j=i}^{m-1}[I + \Delta t A_{n+j}]\Delta t\mathbf{f} + \Delta t\mathbf{f}.
\end{aligned} \tag{16}$$

Here m is the number of time steps that the Forward Euler algorithm is required to take for a robust and stable simulation of the NO_2 data between the $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$ snapshots. Note that in the last row the left multiplication convention is used, i.e. $\prod_{i=1}^{n} A_i = A_n \cdots A_2 A_1$. This notation will be used throughout the report.
From Equation (16) it follows that the source vector $\mathbf{f}$ can be recovered by solving the following linear problem:

$$S\mathbf{f} = \mathbf{u}_{n+m} - \tilde{\mathbf{u}}_{n+m} \tag{17}$$

$$S = \Delta t\left(\sum_{i=1}^{m-1}\prod_{j=i}^{m-1}[I + \Delta t A_{n+j}] + I\right) \tag{18}$$

$$\tilde{\mathbf{u}}_{n+m} = \prod_{i=0}^{m-1}[I + \Delta t A_{n+i}]\mathbf{u}_n, \tag{19}$$

where $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$ are the vectors of the measured NO_2 concentrations at t_n and t_{n+m}, the matrices A_i, $i = n, n+1, \dots, n+m-1$; containing the wind information, are given, and the system matrix S is assumed to be invertible. Notice that the vector $\tilde{\mathbf{u}}_{n+m}$ represents the NO_2 concentration that would have been observed at t_{n+m} if there was no source, i.e., if $\mathbf{f} = \mathbf{0}$, whereas $\mathbf{u}_{n+m}$ is the actual observed NO_2 concentration.
There are two approaches to solving the linear problem in (17). The first approach is a direct approach, where the matrix S is constructed as shown in (18). Afterwards a direct sparse solver is used to solve the linear system (17). The other approach is based on a recurrence relation to compute the matrix-vector product $S\mathbf{v}$, in order to omit matrix-matrix multiplications. This recurrence relation is then used to solve (17) with an iterative method.

5.2.1 Direct approach

The linear system (17) can be solved with either direct or iterative solver. To use a direct solver, the system matrix S should be constructed explicitly as shown in Equation (18) and the vector $\mathbf{u}_{n+m}$ is computed as shown in Equation (19). Subsequently, the equation (17) is solved with a direct sparse solver utilizing, for instance, the *LU* decomposition. In order to make sure the linear system is solvable despite its large size, the matrix S is constructed as sparse. The perk of this approach is that it is the most direct way to find $\mathbf{f}$ yielding small reconstruction errors. One disadvantage is that the explicit

construction of the system matrix S via (18) will accumulate numerical errors. Another disadvantage is that the matrix S is significantly less sparse than its constituent matrices, making it computationally challenging to solve large-scale problems (fine spatial discretisation) and problems with many intermediate time steps between the NO_2 snapshots.

5.2.2 Recurrence relation

The explicit construction of the matrix S and the computationally expensive LU decomposition can both be avoided by using an iterative instead of the direct linear solver. Iterative linear solvers require only matrix-vector products and need significantly less storage. To compute the matrix-vector product, the explicit construction of the matrix S is not necessary. Instead, a recurrence relation for computing the product $S\mathbf{v}$ for any $\mathbf{v}$ is derived:

$$\begin{aligned}
\mathbf{v}_0 &= \Delta t\mathbf{v}, \\
\mathbf{v}_1 &= [I + \Delta t A_{n+1}]\mathbf{v}_0 + \mathbf{v}_0, \\
\mathbf{v}_2 &= [I + \Delta t A_{n+2}][I + \Delta t A_{n+1}]\Delta t\mathbf{v} + [I + \Delta t A_{n+2}]\Delta t\mathbf{v} + \Delta t\mathbf{v} \\
&= [I + \Delta t A_{n+2}]\mathbf{v}_1 + \mathbf{v}_0 \\
&\dots \\
\mathbf{v}_{i+1} &= [I + \Delta t A_{n+i+1}]\mathbf{v}_i + \mathbf{v}_0, \\
S\mathbf{v} &= \mathbf{v}_{m-1}.
\end{aligned} \tag{20}$$

This recurrence relation can be written as a compact recurrence scheme:

$$\begin{aligned}
&\mathbf{v}_0 = \Delta t\mathbf{v}, \\
&\text{for } \; i = 1, \dots, m-1 \; \text{ do:} \\
&\quad \mathbf{v}_i = [I + \Delta t A_{n+i}]\mathbf{v}_{i-1} + \mathbf{v}_0 \\
&S\mathbf{v} = \mathbf{v}_i
\end{aligned} \tag{21}$$

This recurrence relation is implemented in Python as a function that returns $S\mathbf{v}$ for any input vector $\mathbf{v}$.

Two different iterative solvers will be used to solve the linear system (17), namely, the Generalized Minimum Residual Method (GMRES) [18] and the Bi-Conjugate Gradient Stabilized Method (BiCGStab) [22].

5.3 Numerical experiments with synthetic data

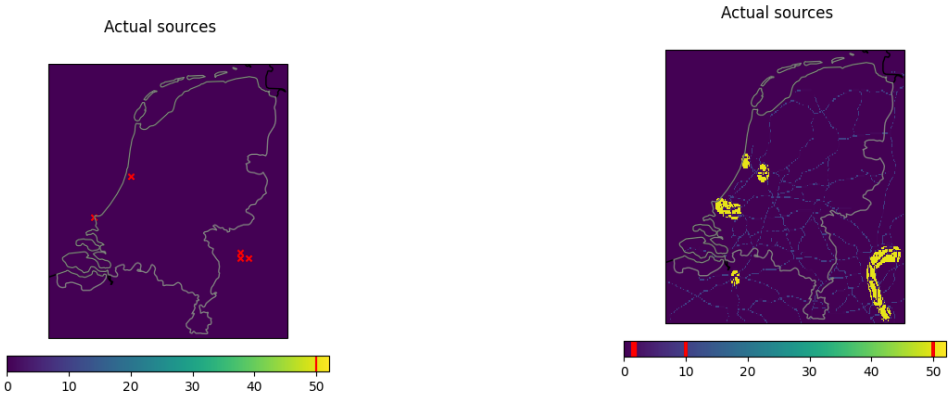
In this section the influence of various factors on the quality of reconstruction of the source function $\mathbf{f}$ is analysed. With the help of numerical experiments, to get a grasp on the values for the parameters and the solving method. First, the difference in the performance of direct and iterative solvers is investigated. Then, the differences between the two kinds of iterative methods are analyzed. Additionally, the reconstruction of different types of sources is investigated. Next, the influence of erroneously estimated transport parameters D and α are investigated. Finally, the effect of noise in the data on the reconstruction results is considered.

The numerical experiments presented in this section are all based on synthetic NO_2 data computed with artificial known sources and actual experimental wind data from 23/08/2019 12:50 to 24/08/2019 12:50. The artificial known sources that are used, are shown in Figure 13. For the source function with point sources, markers are set to indicate the location of the source. The placement of these markers are found by finding local maxima in the data over a certain threshold. this is also done for the found source functions after applying the algorithms. In addition, the strength of each source is indicated by a coloured line on the colorbar of each figure. To find the sources, local maxima are used.

Parameters:	D	Δt	α	$n_y \times n_x$	Source	t_0
Values:	5×10^{-3}	0.009	0.1	194× 280	50	23/08/2019 12:50

Table 5: Parameters that will be used to create synthetic data.

The Forward Euler method is used to generate the NO_2 snapshots $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$. The relevant parameters for these simulations can be found in Table 5. The transport parameters D and α are



(a) Source function with five point sources.

(b) Source function using a map of roads and industrial areas.

Figure 13: Visual representations of the sources used to create the artificial data, the red lines on the colorbar indicate the strengths of each source.

estimated based on the visual comparison of multiple simulations and experimental data, as explained in Section 4.5. The strength of the sources is chosen larger than in Section 4.5, to show the results of the experiments more clearly. For the case of artificial point sources the simulated NO_2 snapshots $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$ are shown in Figure 14.

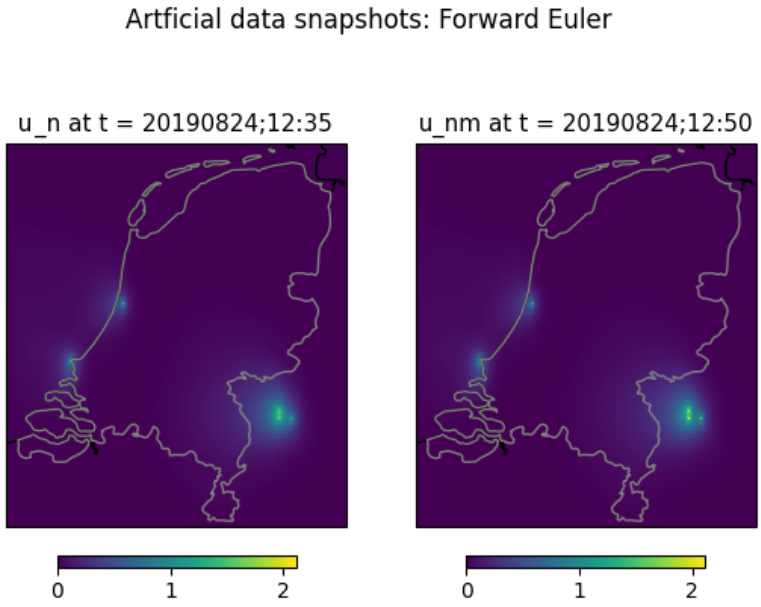


Figure 14: Simulated NO_2 snapshots, $\mathbf{u}_n$ (left) and $\mathbf{u}_{n+m}$ (right), 28 time steps apart, corresponding to artificial points sources and actual wind data.

5.3.1 Comparison of linear solvers

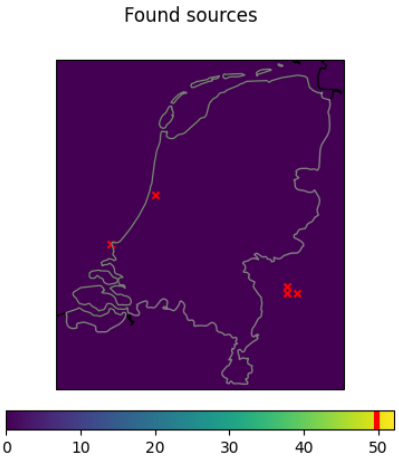
The direct and the iterative algorithms for reconstructing the source function are compared by applying different algorithms to find the same source function and comparing the results. For each test, the parameters will be kept as in Table 5. First, the direct algorithm was tested, which, however, could only be applied with smaller grids due to the lack of memory to perform the LU factorization. Specifically, to use the direct algorithm on a matrix with $194 * 280 \times 194 * 280$ a computer with more than 16 GB of RAM is needed. The construction of the matrix S, as in the equation (18), did work with a CPU time of 284 seconds.

Then, the iterative source reconstruction algorithm, that can handle much larger grids, was tested. Two different iterative solvers, GMRES and BiCGStab, for the linear system (17), employing the recurrence relation (37), have been implemented. For both solvers the tolerance was set to: `tol=1e-6`. The two iterative solvers are compared both visually, by assessing the performance, and by comparing the relative residual norm ($\hat{r}_k$) per iteration. The relative residual norm is defined as:

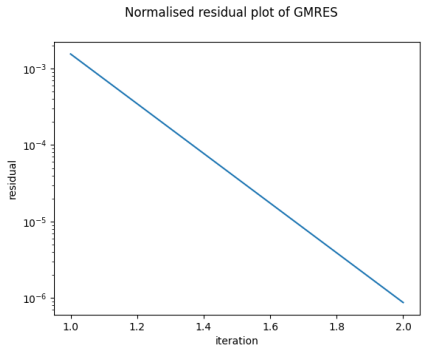
$$\hat{r}_k = \frac{\|R\mathbf{f}_k - (\tilde{\mathbf{u}}_n - \mathbf{u}_n)\|_2}{\|\tilde{\mathbf{u}}_n - \mathbf{u}_n\|_2}. \tag{22}$$

In Figure 15a the visual representations of the actual and the reconstructed source functions are shown for the GMRES solver `scipy.sparse.linalg.gmres()`.

It is clear that the algorithm is able to find both the locations and the strengths of the sources. While the reconstructed $\mathbf{f}$ contains some numerical noise, its amplitude does not exceed 2.59 and it is mostly localized around the actual point sources. The reconstructed strengths of the sources are between 49.62 and 50.06, i.e., within 0.05 percent of the actual strength. The global error in the L_2 norm is 5.58.



(a) Visual comparison of the actual source function and the found source function.

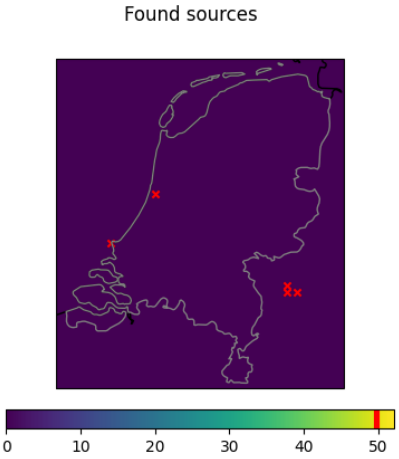


(b) Normalised residual plot.

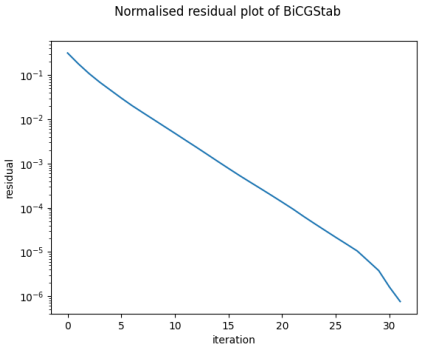
Figure 15: Performance of the iterative GMRES solver, the red lines on the colorbar indicate the strengths of each source.

In Figure 16a, the visual comparison is presented of the actual and reconstructed source functions for the BiCGStab solver `scipy.sparse.linalg.bicgstab()`. The difference between the reconstructed sources for the two iterative solvers is insignificant. Namely, the L_∞ (max) norm of this difference is in the order of 1×10^{-5}.

To compare the performance of the two iterative solvers, the behavior of the corresponding residuals was analyzed, the total number of iterations till convergence, as well as the total CPU time. The residuals as functions of iteration counts are shown in Figure 15b and Figure 16b. The GMRES method converges to the target tolerance in just three iterations. Whereas the BiCGStab method shows a rapid initial convergence, followed by a stage with a slower convergence rate, requiring 32 iterations to reach the target tolerance. Table 6 summarizes these observations, as well as the total time it took the solvers to solve the linear system. In addition to solving the system, it cost 40 seconds to construct the vector



(a) Visual comparison of the actual source function and the found source function.



(b) Residual plot.

Figure 16: Performance of the iterative BiCGStab solver, the red lines on the colorbar indicate the strengths of each source.

$\tilde{\mathbf{u}}_{n+m}$. From this it is found that for the tolerance of `tol=1e-6`, the GMRES method is both faster and uses less iterations.

solver	CPU time	iterations
GMRES	623 s	3
BiCGStab	786 s	32

Table 6: Performance of iterative solvers.

The algorithms can also be applied to reconstruct a more complicated, spatially distributed, source function based on a map of major roads and industrial zones, as shown in Figure 11. The result of reconstruction can be seen in Figure 17. Again, both iterative solvers determine the spatial distribution of the sources well. The L_∞ norm of the difference between the two iterative solvers is 2×10^{-3}. Again the GMRES method converges to the target tolerance of 1×10^{-6} using less iterations. With GMRES using 2 iterations versus BiCGStab using 24. In this case, however, the BiCGStab method was slightly faster than the GMRES method. The methods used 566 seconds and 675 seconds respectively.

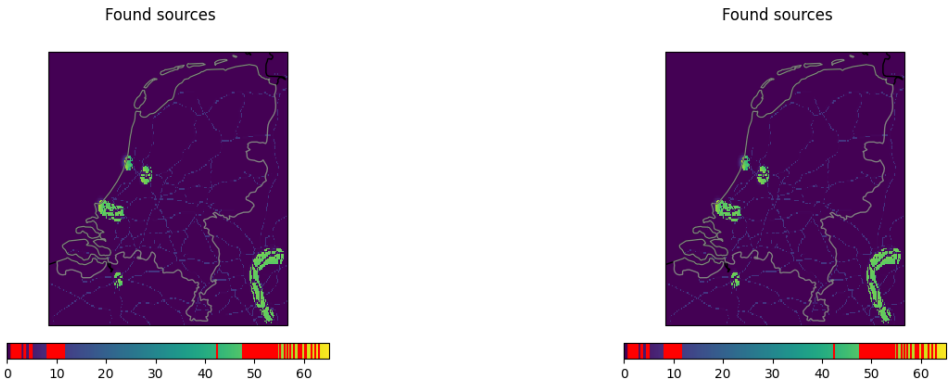


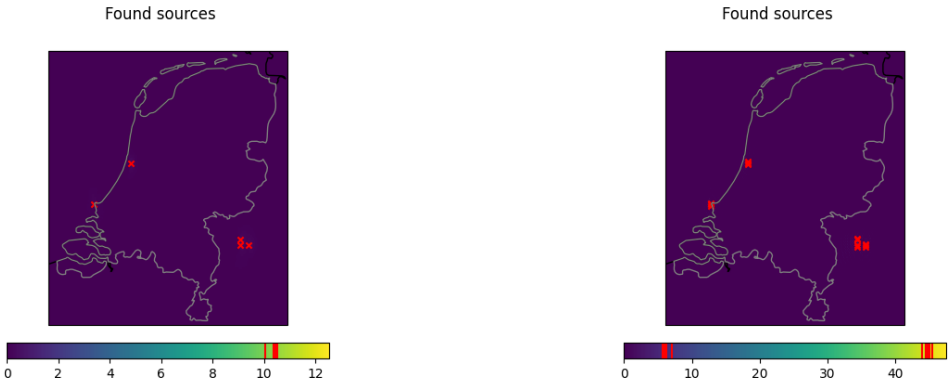
Figure 17: Reconstruction of a spatially distributed source function by iterative solvers: GMRES (left), BiCGStab (right). The red lines on the colorbar indicate the strengths of each source.

From the comparison of the direct and the iterative algorithms it is found that the GMRES method

is the fastest. In addition, the difference in performance between the GMRES method and the BiCGStab method is insignificant. Therefore, in the following numerical experiments the GMRES method will be used in the algorithm for the reconstruction of the source function **f**.

5.3.2 Effect of errors in transport parameters

The source reconstruction algorithm (17)–(19) assumes the knowledge of the transport parameters, i.e., of the diffusion coefficient D and the extinction rate α. In this section the effect of errors in the estimated D and α are analysed. For this, the artificial data that is used remains the same as in the previous section, with the parameters as stated in Table 5. Then, the algorithms is ran, but using 'wrong' parameters. For the first test the parameter D is taken smaller than what is used to create the data. To be precise, $D = 1 \times 10^{-3}$ is used in the algorithm. This means that the diffusion coefficient is underestimated. The result of the test can be seen in Figure **??**. The locations of the point sources are found exactly, but the found strengths of the sources are too low. The found sources have a strength around 10, whereas the actual sources have a strength of 50. Another thing that stands out is that the amount of iterations needed for the calculation is less than for a correctly guessed D, the GMRES method only needed one iteration to solve the system.



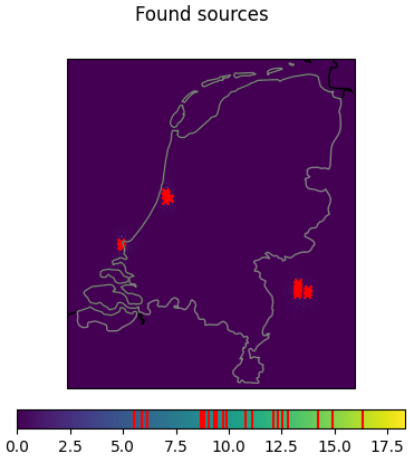
(a) Underestimation of diffusion, with $D = 1 \times 10^{-3}$. (b) Overestimation of diffusion, with $D = 7 \times 10^{-3}$.

Figure 18: Comparison of the effect of underestimation versus overestimation of the diffusion coefficient, using the iterative GMRES solver. The red lines on the colorbar indicate the strengths of each source.

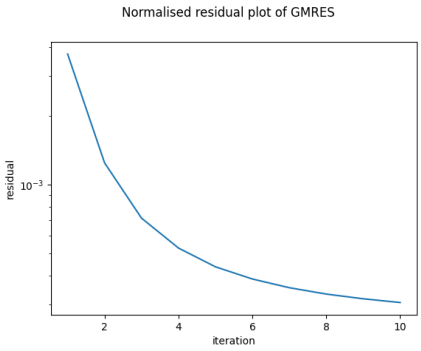
The next step in analysing the influence of a wrongly guessed diffusion coefficient D, is to overestimate it. To start, the diffusion coefficient is taken as $D = 7 \times 10^{-3}$, the results are shown in Figure **??**. Again, the locations are estimated perfectly. The strengths of the sources are estimated relatively close to the actual strengths. The strengths are only slightly underestimated.

In addition, it is tested what happens if the diffusion coefficient is overestimated more. For this, the diffusion coefficient is taken to be $D = 1 \times 10^{-2}$. The result of this test is shown in Figure 19a, as well as the residual of the GMRES method per iteration in Figure 19b. The method is set to stop after 200 iterations, this meant in this case that the residuals did not manage to converge close to zero. For the found source function this means that there is more noise, and there are more 'sources' found.

Apart from the diffusion coefficient, there is also the parameter α, which determines the extinction rate. To study the influence of a wrongly determined parameter α, all of the parameters except for α itself are set to be as in Table 5. First the extinction rate is underestimated, for this $\alpha = 0$ is taken. Afterwards, the extinction rate is overestimated by taking $\alpha = 1$. The results of these tests are shown in Figures 20a and 20b. For both values of α, the locations of the found sources are equal to the locations of the actual sources. In addition, the strengths of the found sources are close to the strengths of the actual sources. Being between 49.49 - 49.90 for $\alpha = 0$ and for $\alpha = 0$ the strengths are between: 50.83 - 51.83. Thus, the influence of the extinction rate parameter α is not as large as the influence of the diffusion coefficient D. The effect that can be seen is that for an overestimated α the strengths of the sources are slightly overestimated and vice versa for an underestimated α.

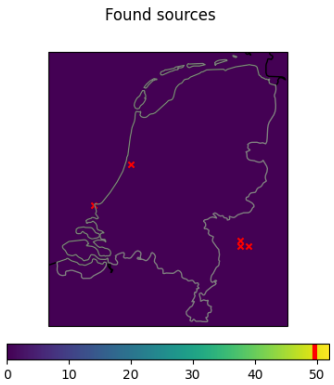


(a) The found source function, using $D = 0.01$.

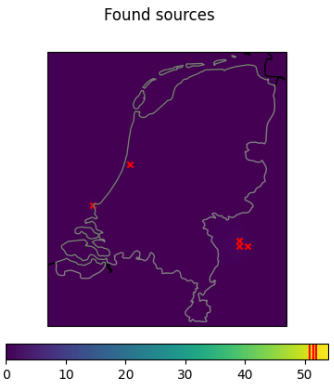


(b) Residual plot.

Figure 19: Performance of the algorithm using a largely overestimated diffusion coefficient, using the iterative GMRES solver. The red lines on the colorbar indicate the strengths of each source.



(a) Underestimation of extinction rate, with $\alpha = 0$.



(b) Overestimation of extinction rate, with $\alpha = 1$.

Figure 20: Comparison of the effect of underestimation versus overestimation of the extinction rate parameter α, using the iterative GMRES solver. The red lines on the colorbar indicate the strengths of each source.

5.3.3 Effect of a non-constant source function

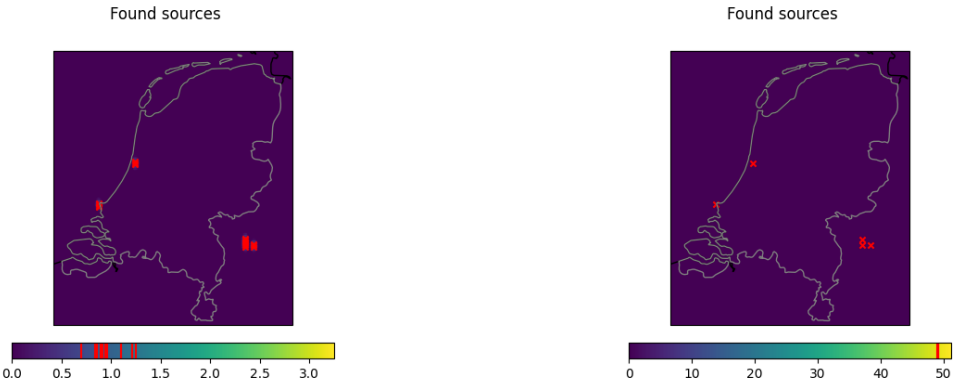
In the introduction of Section 5 a big assumption was made, that the sources are constant between NO_2 snapshots. Although this might be true for short time intervals, for an interval of multiple hours this is not realistic. Therefore it is interesting to see the effect of a non-constant source in the artificial data. First, a source function is created that is constant for a period of time, after which the source is set to zero. This simulates a source that can be turned on and off. To be precise, artificial data is created in a similar fashion as in Section 5.3, but the source function is altered. The source function has a constant strength for $n + \frac{m}{2}$ time steps, the last $\frac{m}{2}$ time steps the source function is set to zero. This comes down to Equation (23). In a similar fashion Equation 24 is created, but in that case the source function is set

to zero for the first $\frac{m}{2}$ time steps after data snapshot $\mathbf{y}_n$.

$$f_{h1}(t) = \begin{cases} \sum_{i=1}^{5} 50\delta(x - x_s)\delta(y - y_s) & \text{, for } t < 12:43 \\ 0 & \text{, for } t \geq 12:43 \end{cases} \tag{23}$$

$$f_{h2}(t) = \begin{cases} \sum_{i=1}^{5} 50\delta(x - x_s)\delta(y - y_s) & \text{, for } t < 12:35 \text{ or } t \geq 12:43 \\ 0 & \text{, for } 12:35 \leq t < 12:43 \end{cases} \tag{24}$$

Using the two Equations (23), (24) artificial data is created using the parameters from 5. This data is then used for the source reconstruction algorithm, with the parameters equal to the parameters used to create the data. In Figure 27a the found sources are shown for artificial data where the sources were set



(a) Found sources with artificial data created with source function: $f_{h1}(t)$.

(b) Found sources with artificial data created with source function: $f_{h2}(t)$.

Figure 21: Performance of the algorithm on artificial data created with sources that are non-constant, using the iterative GMRES solver. The red lines on the colorbar indicate the strengths of each source.

to be zero during the second half of the time between $\mathbf{y}_n$ and $\mathbf{y}_{n+m}$. The location of the sources is found accurately, with some noise. The strengths of the sources, however, have not been reconstructed well. The strengths are severely underestimated.
In Figure 21b the found sources are shown for artificial data where the sources were set to be zero during the first half of the time between $\mathbf{y}_n$ and $\mathbf{y}_{n+m}$. In this case, the algorithm managed to find both the locations and the strengths of the sources accurately.
Even though there are differences between the found sources for non-constant sources versus constant sources, it is not possible to see from the reconstructed source function what types of sources are present. In the real situation, where there will be more sources present of different strengths, sources that are close to zero emissions at the end of the time interval between $\mathbf{y}_n$ and $\mathbf{y}_{n+m}$ will likely be overlooked. Thus it seems that the strength of the source at the end of this time interval is of importance. As this will influence the strength of the source that is reconstructed.

5.3.4 Effect of noise in the NO_2 data

In reality both the NO_2 snapshots and the wind data are not given exactly and contain measurement errors. Mathematically this can be modelled as random additive noise with some, generally unknown, distribution function, which is often assumed to be a zero-mean normal distribution.

First, assume that the wind data are exact and the measurement noise is only present in the NO_2 snapshots $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$. Then, the linear problem (17) can be written as:

$$S\mathbf{f} = F(\mathbf{u}_n + \boldsymbol{\epsilon}_n) - (\mathbf{u}_{n+m} + \boldsymbol{\epsilon}_{n+m}), \tag{25}$$

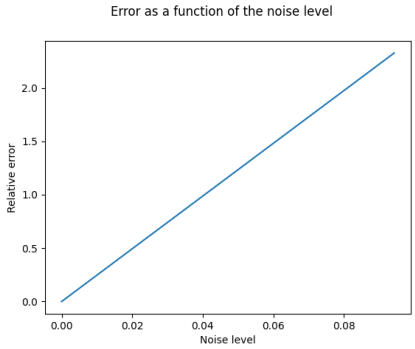
where $F\mathbf{u}_n = \tilde{\mathbf{u}}_{n+m}$, and $\boldsymbol{\epsilon}_n$, $\boldsymbol{\epsilon}_{n+m}$ are the noise vectors. The formal solution of this problem is

$$\mathbf{f}_\epsilon = S^{-1}[F(\mathbf{u}_n + \boldsymbol{\epsilon}_n) - (\mathbf{u}_{n+m} + \boldsymbol{\epsilon}_{n+m})] = \mathbf{f}_{\text{ex}} + \boldsymbol{\delta}, \tag{26}$$

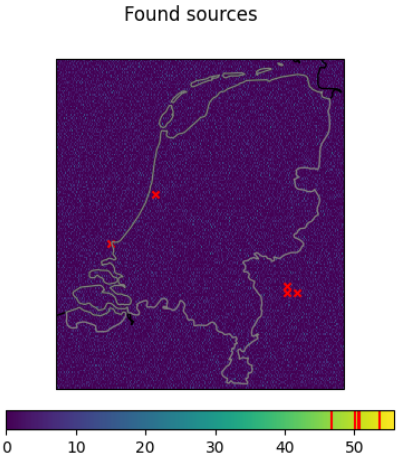
where $\mathbf{f}_{\text{ex}} = S^{-1}[F\mathbf{u}_n - \mathbf{u}_{n+m}]$ is the "exact" solution that would have been obtained in the absence of noise, and $\boldsymbol{\delta} = S^{-1}[F\boldsymbol{\epsilon}_n - \boldsymbol{\epsilon}_{n+m}]$ is the noise in the solution due to the noise in the data.

Hence, the obtained solution $\mathbf{f}_\epsilon$ will, in general, be different from the exact solution $\mathbf{f}_{\text{ex}}$. The amount and type of noise in $\mathbf{f}_\epsilon$ will depend on both the magnitude and content of the vectors $\boldsymbol{\epsilon}_n$, $\boldsymbol{\epsilon}_{n+m}$, and the transformation that these vectors will undergo under the multiplication by the matrices S^{-1} and $S^{-1}F$. One quantity of interest is the global error in the solution that can be defined as the norm $\|\boldsymbol{\delta}\|_2$ and its dependence on the magnitude of the noise.

Let the noise vectors be taken from the same multivariate normal distribution $\mathcal{N}(\mathbf{0}, \sigma^2 I)$. The two vectors $\boldsymbol{\epsilon}_n$, $\boldsymbol{\epsilon}_{n+m}$ are then expected to have the same norm $\|\boldsymbol{\epsilon}_n\|_2 =, \|\boldsymbol{\epsilon}_{n+m}\|_2 = \|\boldsymbol{\epsilon}\|_2$. The error in the solution is computed $\|\boldsymbol{\delta}\|_2 = \|\mathbf{f}_\epsilon - \mathbf{f}_{\text{ex}}\|_2$ as a function of the noise level in the data $\|\boldsymbol{\epsilon}\|_2$. This is shown in Figure 22a, from the graph it can be concluded that the relative error is approximately ten times the level of noise. The source reconstruction algorithm is then applied to artificial data with added noise. To



(a) Plot of the relative error versus the noise level, to be precise:
$\|\boldsymbol{\delta}\|_2/\|\mathbf{f}_{\text{ex}}\|_2$ versus $\|\boldsymbol{\epsilon}\|_2/\|\mathbf{u}_n\|_2$.



(b) Source reconstruction algorithm applied to artificial data with 5% added noise.

both the snapshot $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$ random normally distributed noise is added, with a standard deviation of 0.05. The result is shown in Figure 22b, where the location and strength are estimated well, but there is clearly noise present in the overall function. With amplitudes as high as 11. For experimental data with sources that do not have a strength of 50, this can form a larger problem. As sources can be left undetected.

The source reconstruction algorithm is found to be sensitive to noise in the data, which gives that the problem is ill-posed. Therefore regularisation may be useful. Due to time restrictions, this will not be studied in this report, in the Discussion in Section 8 a recommendation will be given on a regularisation method that can be used. In addition, the case for which noise is also present in the wind data will not be discussed in this report, but will be left for the Discussion.

5.4 Source reconstruction based on the Backward Euler method

The conditional stability of the Forward Euler method requires time steps Δt in the order of a minute making it practically impossible to process the actual experimental data where the NO_2 snapshots are currently taken 24 hours apart. Indeed this time scale would lead to the parameter m in (18) in the order of 2.7×10^3. The Backward Euler method, on the other hand, is unconditionally stable and permits much larger time steps. The iterations of the Backward Euler method can be written as:

$$\begin{aligned}
\mathbf{u}_{n+1} &= [I - \Delta t A_{n+1}]^{-1}\mathbf{u}_n + [I - \Delta t A_{n+1}]^{-1}\Delta t\mathbf{f}, \\
\mathbf{u}_{n+2} &= [I - \Delta t A_{n+2}]^{-1}[I - \Delta t A_{n+1}]^{-1}\mathbf{u}_n \\
&\quad + [I - \Delta t A_{n+2}]^{-1}[I - \Delta t A_{n+1}]^{-1}\Delta t\mathbf{f} + [I - \Delta t A_{n+2}]^{-1}\Delta t\mathbf{f}, \\
&\quad \dots \\
\mathbf{u}_{n+m} &= \prod_{i=1}^{m}[I - \Delta t A_{n+i}]^{-1}\mathbf{u}_n + \sum_{i=1}^{m}\prod_{j=i}^{m}[I - \Delta t A_{n+j}]^{-1}\Delta t\mathbf{f},
\end{aligned} \tag{27}$$

These iterations can also be written as:

$$\sum_{i=1}^{m}\prod_{j=i}^{m}[I-\Delta tA_{n+j}]^{-1}\Delta t\mathbf{f}=\mathbf{u}_{n+m}-\tilde{\mathbf{u}}_{n+m}, \tag{28}$$

where

$$\tilde{\mathbf{u}}_{n+m}=\prod_{i=1}^{m}[I-\Delta tA_{n+i}]^{-1}\mathbf{u}_n, \tag{29}$$

is the NO_2 concentration that would have been produced with $\mathbf{f}=\mathbf{0}$, computed with the Backward Euler method. It is easy to verify that:

$$\prod_{i=0}^{m-1}[I-\Delta tA_{n+m-i}](\mathbf{u}_{n+m}-\tilde{\mathbf{u}}_{n+m})=\sum_{i=1}^{m-1}\prod_{j=i}^{m-1}[I-\Delta tA_{n+m-j}]\Delta t\mathbf{f}+\Delta t\mathbf{f}. \tag{30}$$

Thus, the linear problem for the source vector $\mathbf{f}$ can now be written in the form:

$$R\mathbf{f}=B(\mathbf{u}_{n+m}-\tilde{\mathbf{u}}_{n+m}), \tag{31}$$

with the matrices:

$$R=\left(\sum_{i=1}^{m-1}\prod_{j=i}^{m-1}[I-\Delta tA_{n+m-j}]+I\right)\Delta t, \tag{32}$$

$$B=[I-\Delta tA_{n+1}][I-\Delta tA_{n+2}]\cdots[I-\Delta tA_{n+m}] \tag{33}$$

$$=\prod_{i=0}^{m-1}[I-\Delta tA_{n+m-i}]. \tag{34}$$

One can avoid computing the Backward Euler solution $\tilde{\mathbf{u}}_{n+m}$ by taking into account that $B\tilde{\mathbf{u}}_{n+m}=\mathbf{u}_n$, leading to the following equivalent formulation:

$$R\mathbf{f}=\tilde{\mathbf{u}}_n-\mathbf{u}_n, \tag{35}$$

where $\tilde{\mathbf{u}}_n=B\mathbf{u}_{n+m}$ is the measured concentration $\mathbf{u}_{n+m}$, "back-propagated" in time with the matrix B. The major difference with the algorithms based on the Forward Euler method is in the permitted larger time step Δt.

As was done in the algorithm using the Forward Euler method in Section 5.2.2, a recurrence relation for computing the product $R\mathbf{v}$ for any $\mathbf{v}$ is derived:

$$\begin{aligned}
\mathbf{v}_0&=\Delta t\mathbf{v},\\
\mathbf{v}_1&=[I-\Delta tA_{n+m-1}]\mathbf{v}_0+\mathbf{v}_0,\\
\mathbf{v}_2&=[I-\Delta tA_{n+m-2}][I-\Delta tA_{n+m-1}]\Delta t\mathbf{v}+[I-\Delta tA_{n+m-2}]\Delta t\mathbf{v}+\Delta t\mathbf{v}\\
&=[I-\Delta tA_{n+m-2}]\mathbf{v}_1+\mathbf{v}_0\\
&\cdots\\
\mathbf{v}_{i+1}&=[I-\Delta tA_{n+m-(i+1)}]\mathbf{v}_i+\mathbf{v}_0,\\
R\mathbf{v}&=\mathbf{v}_{m-1}.
\end{aligned} \tag{36}$$

This recurrence relation can be written as a compact recurrence scheme:

$$\begin{aligned}
&\mathbf{v}_0=\Delta t\mathbf{v},\\
&\text{for } \ i=1,\ldots,m-1 \ \text{ do:}\\
&\qquad \mathbf{v}_i=[I-\Delta tA_{n+m-i}]\mathbf{v}_{i-1}+\mathbf{v}_0\\
&R\mathbf{v}=\mathbf{v}_i
\end{aligned} \tag{37}$$

5.5 Numerical experiments with synthetic data

In this section the performance of the source reconstruction algorithm based on the Backward Euler method is assessed. First, the method is tested against the algorithm based on the Forward Euler method by using the exact same parameters as the initial test in Section 5.3.1. It is expected for the Backward Euler algorithm to perform less good than the Forward Euler method on the artificial data created by the Forward Euler method. This is because artificial data created by the Forward Euler method slightly differs from artificial data created by the Backward Euler method. By comparing Figures 14 and 23a, there is no difference to be seen. The difference in the L_2 norm is 2×10^{-4}, and in the L_∞ it is 0.013 for both $\mathbf{u}_n$ and $\mathbf{u}_{n+m}$. Firstly, the algorithm will be tested on both types of artificial data.

(a) Snapshots which are 28 time steps (for $\Delta t = 0.009$s) apart.

(b) Snapshots which are 240 time steps (for $\Delta t = 0.1$s) apart.

Figure 23: Simulated NO_2 snapshots, $\mathbf{u}_n$ (left) and $\mathbf{u}_{n+m}$ (right), corresponding to artificial point sources and actual wind data.

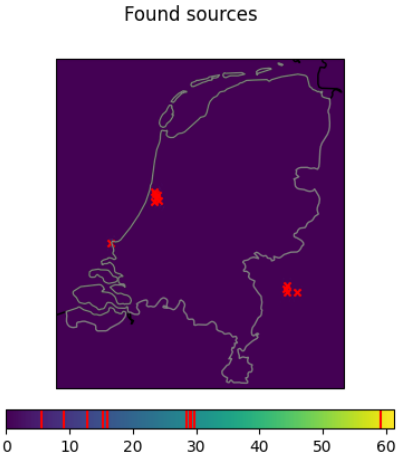
This first test is done on data with only a short amount of time between data snapshots, in the real data the time between snapshots is 24 hours. Therefore in the proceeding tests new artificial data is used. This data is, again, created by using the parameters from Table 5, except for the time step. This istaken to equal: $\Delta t = 0.1$. In addition the snapshots are taken at 24 hours difference, which results in a snapshot at 12:50 on 24/08/2019 and at 12:50 on 25/08/2019. These are shown in Figure 23b.
The first experiment that is conducted using the artificial data with 24 hours between snapshots is to analyse the influence of the time step. Afterwards, the influence of the transport parameters is shortly revisited and the effect of non-constant sources is shown.

5.5.1 Comparison of performance FE and BE method

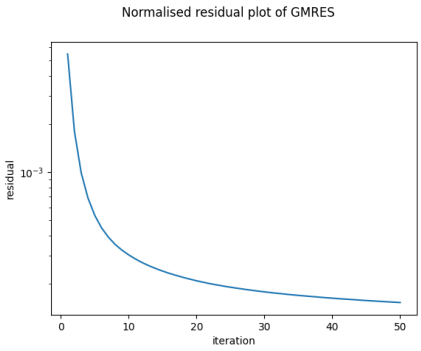
As a first test the algorithm is applied to the exact same synthetic data as was made for the Forward Euler algorithm, see Figure 14. This includes using the same parameters for the source reconstruction algorithm, which can be found in Table 4. In addition, the tolerance of the GMRES solver was again set to `tol=1e-6`. The solver did not converge to the tolerance and was stopped after 50 iterations. In Figure 24b, it can be seen that the relative residual norms converge slower as the iterations proceed. After 50 iterations, the relative residual is still in the order 1×10^{-3}. Comparing this to the convergence of the residuals of the Forward Euler source finder method, see Figure 15b, shows that the Forward Euler method is a lot faster on this data. This difference in speed is based on the time it takes GMRES method to converge, or to reach the maximum number of iterations. The time it took to construct the vector $\tilde{\mathbf{u}}_n$ was approximately 40 seconds. This is does not affect the total time for the calculation greatly, as the total simulation took approximately 4 hours.
To assess the quality of the found source function, the difference between the actual source function and the reconstructed source function is analysed. With an L_2 norm of 125.6 and an L_∞ norm of approximately 59, it is clear that the found source function is not perfect. The reason for the infinity norm to be as high is because of a wrongly estimated location for a peak.

5.5.2 Effect of a growing Δt

The main argument for using the Backward Euler method in the source finding algorithm is that it is an unconditionally stable method. Which means that the time step (Δt) can be chosen larger without



(a)



(b) Relative residual plot.

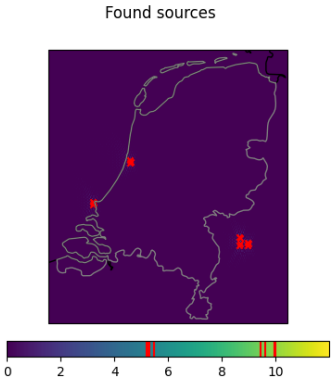
Figure 24: Performance of the source reconstruction algorithm based on the Backward Euler method, using the GMRES solver, tested on artificial data created by Forward Euler. The red lines on the colorbar indicate the strengths of each source.

causing instabilities. Therefore, the first experiment that will be conducted is to analyse the influence of the magnitude of the time step. To perform this experiment, first the artificial data from Figure 23b is used. All parameters are left equal to Table 5 except for the time step which gradually gets enlarged from $\Delta t = 0.666$h to $\Delta t = 4$h. Which means the number of time steps between the two data snapshots ranges from 36 steps to 6 steps.
The result of the experiment can be seen in Figure 25. For each value for Δt the locations of the sources are reconstructed exactly. The difference in performance lies in the reconstruction of the strengths of the sources, as well as the presence of noise and the time the computation takes. From the figure it becomes clear that as the time step grows, the accuracy of the strengths of the sources get higher. In addition, the level of noise shrinks. If the time step were to be increased to 24h, which gives one time step between the data snapshots, the found sources are almost identical to the actual sources.
The other trend that can be noticed when performing this experiment is that as the time step grows, the amount of time needed for the computation gets shorter. Where the computation with 36 time steps took over five hours, the computation with six time steps took approximately 40 minutes.

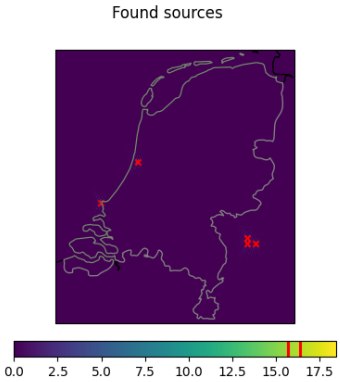
From this experiment it becomes clear that the algorithm does not perform worse with a larger time step, and thus less steps between two snapshots. This can be explained by taking a closer look at the vector $\tilde{\mathbf{u}}_n$, which plays a great role in the algorithm, see Equation (35). The vector is defined as follows:

$$\tilde{\mathbf{u}}_n = B\mathbf{u}_{n+m} = \prod_{i=0}^{m-1} [I - \Delta t A_{n+m-i}]\mathbf{u}_{n+m} = [I - \Delta t A_{n+1}] \cdots [I - \Delta t A_{n+m}]\mathbf{u}_{n+m}. \tag{38}$$

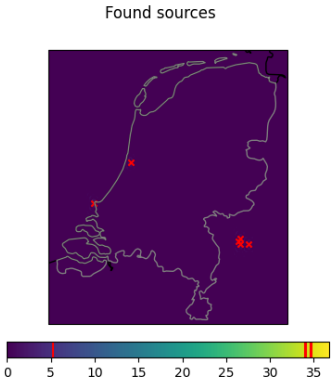
In Section 4.3.1 it was found that all eigenvalues of A_i have negative real part. Which implies that $1 - \Delta t\lambda > 1$. This in turn implies that the spectral radii of the matrices $[I - \Delta t A_i]$ are larger than one, this gives that $\lim_{k\to\infty}[I - \Delta t A_i]^k = \infty$. Which explains that for a large number of steps between NO_2 data snapshots, the vector $\tilde{\mathbf{u}}_n$ grows large. Therefore the number of steps between data snapshots should be limited.
Even though the result of this experiment would indicate that a time step of $\Delta t = 24$h would be optimal, this is not realistic. With a time step this large comes that the influence of the wind is not taken into account as much. For a convection dominated problem, which the spread of NO_2 is, neglecting the influence of the wind is not done. Especially in case the wind is not constant over the period of 24 hours. Thus, taking into account both the advantages and the disadvantages of a large time step, the time step will be taken around $\Delta t = 1$. Which is in agreement with the frequency on which the wind data is available.
This leads to the parameter values presented in Table 7, which will be used as the basis for the conduction of experiments in the following sections.



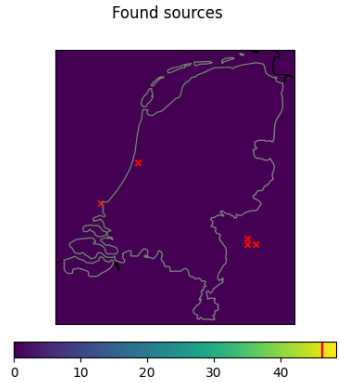
(a) Reconstructed sources with $\Delta t = 0.666$.



(b) Reconstructed sources with $\Delta t = 1$.



(c) Reconstructed sources with $\Delta t = 2$.



(d) Reconstructed sources with $\Delta t = 4$.

Figure 25: Performance of the source reconstruction algorithm based on the Backward Euler method, using the GMRES solver, tested on artificial data created by Backward Euler. The red lines on the colorbar indicate the strengths of each source.

Parameters:	D	Δt	α	$n_y \times n_x$	Source	$\mathbf{y}_n$	$\mathbf{y}_{n+m}$
Values:	5×10^{-3}	1	0.1	194× 280	50	24/08/2019 12:50	25/08/2019 12:50

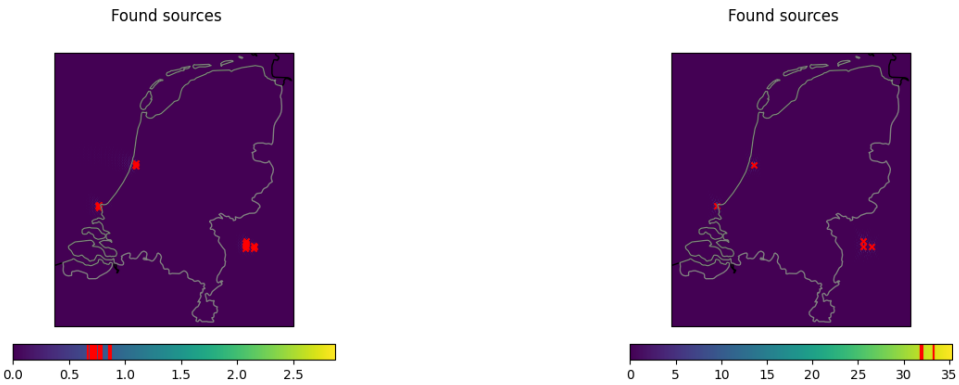
Table 7: Parameters that will be used as a basis for the testing of the source finding algorithm based on the Backward Euler method.

5.5.3 Effect of errors in transport parameters

In Section 5.3.2, the effect of errors in the transport parameters is studied for the algorithm based on the Forward Euler method. In this section similar experiments are conducted to find out if wrongly estimated transport parameters have an analogous effect on the source finding algorithm based on the Backward Euler method. Therefore, first the diffusion coefficient is varied by taking $D = 0.001$ and $D = 0.01$, thus underestimating and overestimating the parameter. The parameters other than D are kept equal to the parameters in Table 7.

The results of the underestimation versus the overestimation of the diffusion coefficient D are shown in Figure 26. If the diffusion is underestimated, the source locations are less accurately found. Instead of a clear peak that is found in the source function, there is a lot of noise around the found sources of about the same strength. In addition, the strengths of the sources are extremely underestimated, as can be seen in Figure 26a.

For an overestimated diffusion coefficient, the impact is less drastic on the found sources. The locations of the sources are estimated perfectly, but the strengths of the sources are lightly underestimated. The effects of an underestimated level of diffusion on the source reconstruction algorithm based on the Back-



(a) Underestimation of diffusion, with $D = 1 \times 10^{-3}$. (b) Overestimation of diffusion, with $D = 1 \times 10^{-2}$.

Figure 26: Comparison of the effect of underestimation versus overestimation of the diffusion coefficient on the source reconstruction algorithm based on the Backward Euler method, using the GMRES solver. The red lines on the colorbar indicate the strengths of each source.

ward Euler method are different from the effects seen in Section 5.3.2. For the algorithm discussed in this section, the influence is bigger, suggesting that the algorithm does not perform well on a largely underestimated diffusion coefficient.
The overestimation of the diffusion has a reverse relation. The algorithm based on the Backward Euler method is less affected by an overestimated diffusion coefficient, than the algorithm based on the Forward Euler method.
Since the variation of the decay parameter α did not have a significant effect on the performance of the source finding algorithm based on Forward Euler, this will not be tested again for the Backward Euler algorithm. This is due to time restrictions.

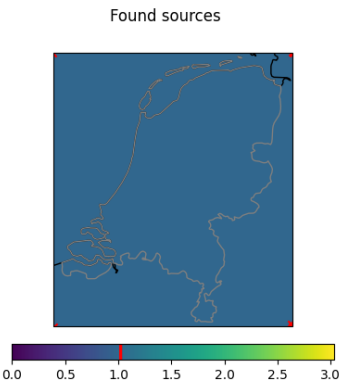
5.5.4 Effect of a non-constant source function

In Section 5.3.3 it was investigated how well the algorithm based on the Forward Euler method is at reconstructing sources that are non-constant. This was tested by using the source functions as defined in Equation (23) and (24), to create artificial data. In this section a similar approach is used to investigate whether the source finding algorithm based on the Backward Euler method can reconstruct non constant sources. The main goal is to find trends in what the effect is on the found source function. If trends are found, this can be used for the application of the algorithm on experimental data, to see if it is possible to determine what type of sources are present.
The exact source functions that are used to create the artificial data, are the following:

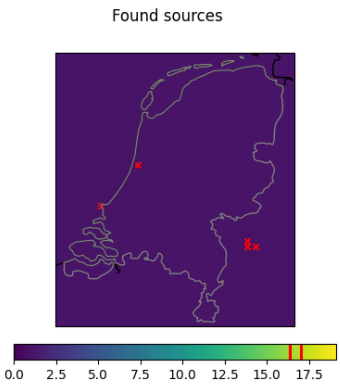
$$f_{h3}(t) = \begin{cases} \sum_{i=1}^{5} 50\delta(x - x_s)\delta(y - y_s) & \text{, for } t < 00:50 \text{ (25/08/2019)}, \\ 0 & \text{, for } t \geq 00:50 \text{ (25/08/2019)}. \end{cases} \tag{39}$$

$$f_{h4}(t) = \begin{cases} \sum_{i=1}^{5} 50\delta(x - x_s)\delta(y - y_s) & \text{, for } t < 12:50 \text{ (24/08/2019) or } t \geq 00:50 \text{ (25/08/2019)}, \\ 0 & \text{, for } 12:50 \text{ (24/08/2019)} \leq t < 00:50 \text{ (25/08/2019)}. \end{cases} \tag{40}$$

The results of this test are shown in Figure 27, where a similar trend can be seen as was found in Section 5.3.3. In case of Equation (39), the algorithm based on Backward Euler is not able to find sources. As the found source function is approximately one for each point in the domain. For the case with a source function (40), the algorithm did find sources. The locations of the found sources are estimated perfectly, but the strengths of the sources is underestimated. From this experiment it can be concluded that sources that have not been emitting NO_2 in the 12 hours before snapshot $\mathbf{y}_{n+m}$ is taken, are not reconstructed by the source finding algorithm. Sources that were emitting NO_2 during that time were found, but were underestimated in case they were not working the entire time between $\mathbf{y}_n$ and $\mathbf{y}_{n+m}$. Thus, when applying the source finding algorithm to experimental data, sources that are not constantly emitting NO_2 will be underestimated in strength, or will not be detected. On way this will be apparent



(a) Found sources with artificial data created with source function: $f_{h3}(t)$.



(b) Found sources with artificial data created with source function: $f_{h4}(t)$.

Figure 27: Performance of the BE based algorithm on artificial data created with sources that are non-constant, using the iterative GMRES solver. The red lines on the colorbar indicate the strengths of each source.

is when comparing the found sources to the NO_2 snapshots, where the second snapshot will likely have a larger influence on the reconstructed sources than the first snapshot.

6 Application of the algorithm on experimental data

In this section the source reconstruction algorithm based on the Backward Euler method was applied to experimental data. This experimental data consists of near-daily snapshots of the NO_2 concentration over the Netherlands, delivered by TROPOMI. The snapshots that are used in the report are presented in Figure 28, which consists of the data of August 2019. The source reconstruction algorithm is first

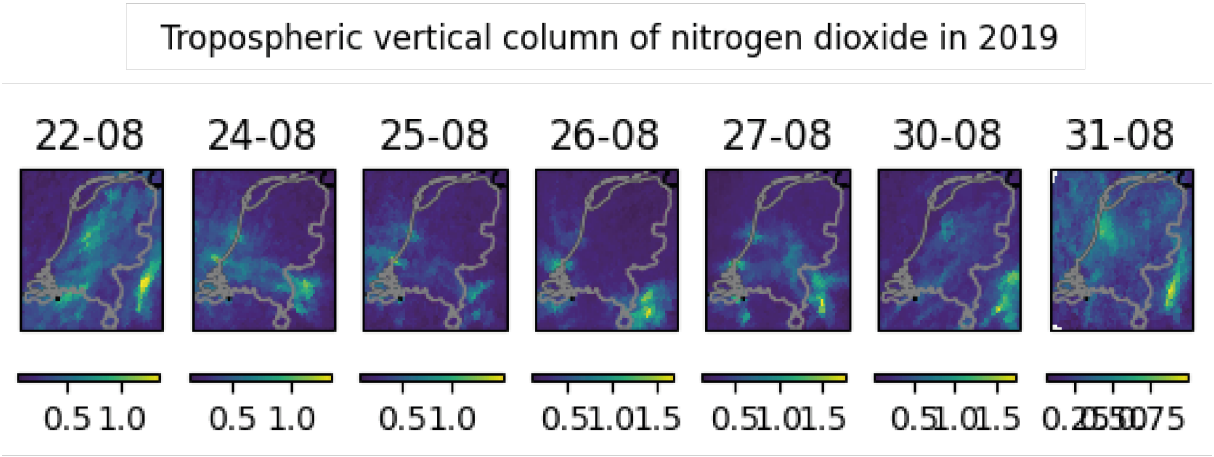
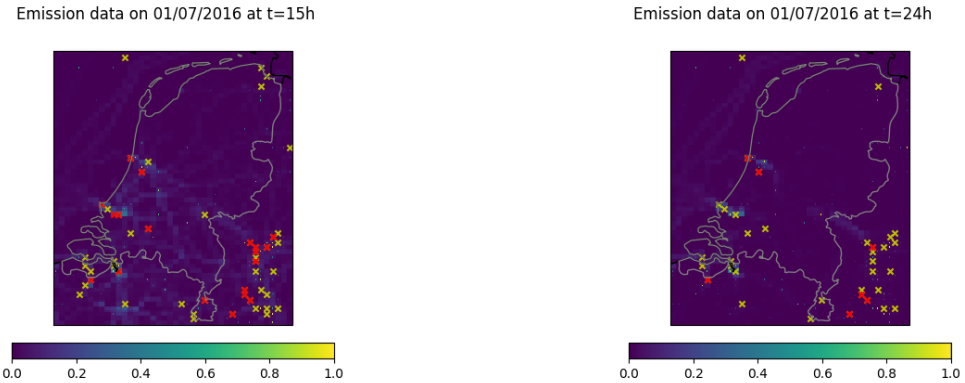


Figure 28: NO_2 data snapshots for the month August (e30 mls/deg^2).

applied to snapshots which are taken on consecutive days. The snapshots on the days: 24 and 25 August are used to test the algorithm on. The results of the algorithm are compared to the current data TNO has on the locations and strengths of NO_2 sources. Based on this comparison, an optimal Δt is chosen to apply on the remaining data of August. This data on NO_2 sources is made available on an hourly bases, and thus it is not constant over 24 hours, for the period 01/07/2016 until 14/07/2016. This is clearly not in the same time frame as the measured NO_2 concentrations, but gives an indication on the strengths and locations of known sources.



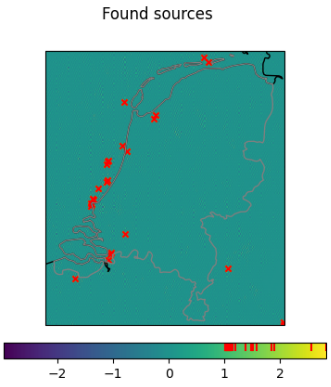
(a) Known source function at 15:00 on 01/07/2016.

(b) Known source function at 00:00 on 02/07/2016.

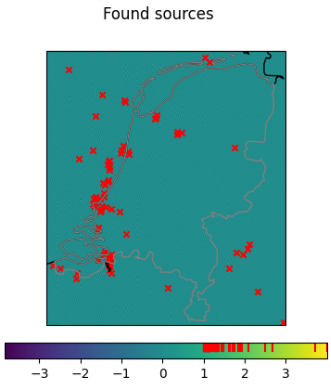
Figure 29: Source functions over time based on data from TNO (e30 mls/deg^2) on the emission of NO_2, the colorbar is scaled such that sources with a low emission rate are visible. The red crosses mark sources with an emission rate over 4×10^{30} mls/deg^{2}h, the yellow mark sources with emission rates over 1×10^{30} mls/deg^{2}h.

In Figure 29 two source functions are shown, based on the data on known NO_2 sources. Each source that has a strength of over 5×10^{30} mls/deg^{2}h is marked by a red cross. Figure 29a represents the sources,

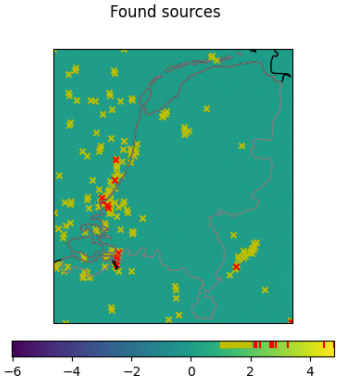
when the emission is the highest. On the other side, Figure 29b shows the sources at the time the emissions are the lowest. There are clearly differences between the two source distributions, but there are also a number of sources which they have in common. In a 24 hour cycle multiple sources in the Ruhrgebiet and in Ijmuiden remain active with emissions over 5×10^{30} mls/deg^{2}h. In the areas: Amsterdam, Rotterdam and Antwerp there are also constant emissions taking place, but at a lower amplitude.
As a primal test, the source reconstruction algorithm based on the Backward Euler method is applied to the snapshots: $\mathbf{y}_n = 20190824; 12:50$ and $\mathbf{y}_{n+m} = 20190825; 12:31$. The parameters that are used are equal to the parameters set out in Table 7, with the time step $\Delta t \in 0.79, 0.98, 1.32, 1.97$. This gives that there are between 36 and 12 time steps between $\mathbf{y}_n$ and $\mathbf{y}_{n+m}$.



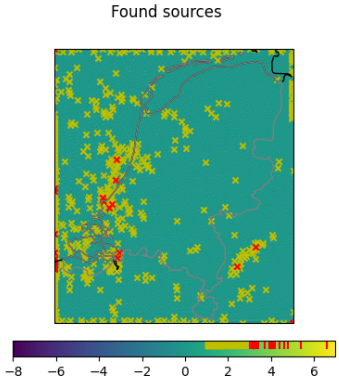
(a) Reconstructed sources with $\Delta t = 0.79$h.



(b) Reconstructed sources with $\Delta t = 0.98$h.



(c) Reconstructed sources with $\Delta t = 1.32$h.



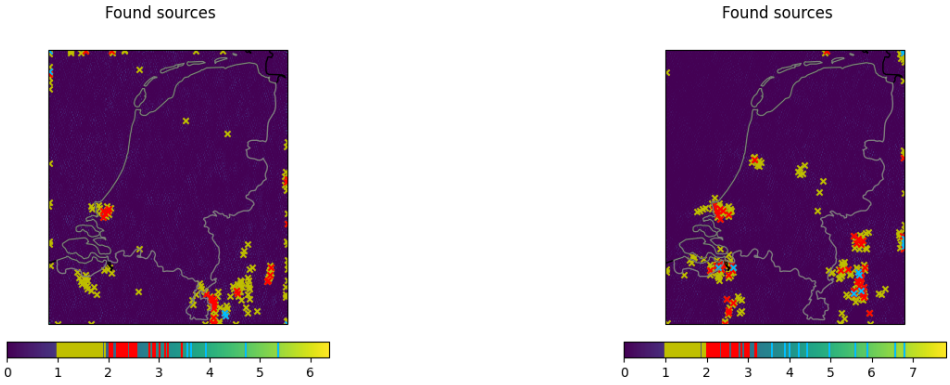
(d) Reconstructed sources with $\Delta t = 1.97$h.

Figure 30: Reconstructed sources from $\mathbf{y}_n = 20190824; 12:50$ and $\mathbf{y}_{n+m} = 20190825; 12:31$, using the source reconstruction algorithm based on the Backward Euler method. Local maxima are marked as sources using different colours.

In Figure 30, the results can be seen of the first application of the source reconstruction algorithm on experimental data. The figure shows the reconstructed sources that are found by using different values for the time step. From these results it can be deducted that as the time step grows, the amount of sources, and the strengths of these found sources, increases. This discovery, is in agreement with the findings of Section 5.5.2. Comparing Figure 30 to Figure 29, would suggest that the best performing time step has a value between: $\Delta t = 1.32$h. Since, for this time step there are sources marked in all of the areas sources are expected. For a Δt which is larger, there are too many sources marked, which makes the found source function chaotic. In addition, most found sources are located on the boundary of the domain, where few sources are located in reality. For $\Delta t < 1.32$h less sources are found. Furthermore, the sources which are found have a lower strength, only very few sources have a strength over 4×10^{30} mls/deg^{2}h.
One thing that stands out in the reconstructed sources found in Figure 30, is that there are a lot of sources located in the sea. Though there is a very lightly visible source located in the sea in Figure 29a, this is a lot lower than what is found in the reconstructed sources. The emission of NO_2 in the sea can

be explained by the emissions of cargo ships and ferries, but this is not enough to explain the amount of sources found in Figure 30c. The actual cause, is probably explained by the discovery in Section 5.5.4. That the second NO_2 snapshot has a large effect on the reconstructed sources. Indeed, in the NO_2 snapshot of 25 August the is high concentrations of NO_2 visible over the sea, which is most likely due to the off-shore wind on those days.

The chosen size of the time step: $\Delta t = 1.32$ amounts to 18 time steps between data snapshots, thus $m = 18$. This number of steps is used to apply the algorithm to more data, in order to see if there are similarities. For this, the NO_2 snapshots from 25, 26 and 27 August are used. The reconstructed sources that are found using the source finding algorithm are presented in Figure 31.



(a) Reconstructed sources between:
$\mathbf{y}_n = 20190825; 12:31$, $\mathbf{y}_{n+m} = 20190826; 12:12$.

(b) Reconstructed sources between:
$\mathbf{y}_n = 20190826; 12:12$, $\mathbf{y}_{n+m} = 20190827; 11:53$.

Figure 31: Reconstructed sources using the source reconstruction algorithm based on the Backward Euler method, local maxima are marked as sources and are pointed out in the plot and the colorbar using different colours.

Figure 31a gives the reconstructed sources over the period from 25 August to 26 August. Most of the found sources are located in the Rurhgebiet and in Rotterdam, no sources are found in Amsterdam. As for the sources located in the ocean in Figure 30, this could be explained by the discovery that the second NO_2 snapshot has a larger effect on the reconstructed sources than the first snapshot, as was found in Section 5.5.4. The NO_2 snapshot of 26 August shows low NO_2 concentrations over Amsterdam, but large concentrations over the Ruhrgebiet and Rotterdam.
Figure 31b shows the found sources over the period from 26 to 27 August. In this case, again most sources are found in the Rurhgebiet and Rotterdam, but there are also more sources found around Amsterdam, Antwerp and Brussels. This can once again be explained by the distribution of NO_2 concentrations in the second snapshot on 27 August.
The Figures 31a and 31b can be compared to Figure 29, to study the difference between the found sources and the known sources. The first thing that stands out is, that the reconstructed source function consists of a lot more sources than there are known sources, with emissions over 1×10^{30} mls/deg^{2}h. The main locations of the reconstructed sources and the known sources match on a large scale. For both, the large cities and industrial areas are pinned as NO_2 pollution sources. Where large differences start to occur is when the sources are shown with low emission rates. For the known sources, in Figure 29a, there are for example line sources visible where roads are located. The scaled version of the reconstructed sources between 25 and 26 August, in Figure 32, shows no clear lines where roads are located. The source function contains noise over the whole domain, which makes it impossible to recognise low emission sources with emissions under 1×10^{30} mls/deg^{2}h.

From the application of the algorithm based on the Backward Euler method, it is found that the algorithm is capable of finding areas with large emission rates. The algorithm is able to reconstruct the locations of industrial zones near Amsterdam, Rotterdam, Antwerp and the Ruhrgebiet. The actual strength of individual sources is not determined accurately. In addition the found sources are mainly based on the second NO_2 snapshot. Thus, the algorithm can be used to determine source locations on a large scale, but is not able to exactly determine locations of independent sources.

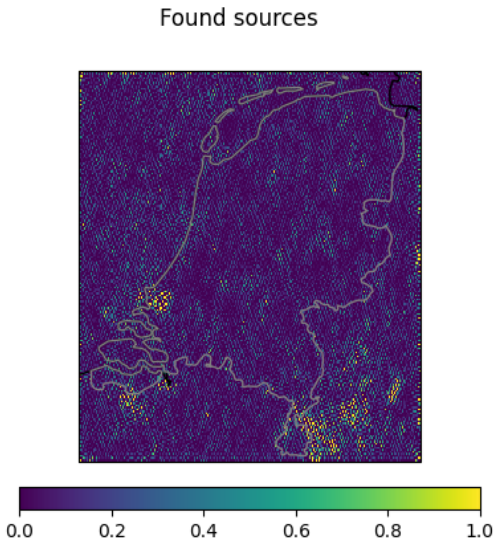


Figure 32: Reconstructed source function found in Figure 31a, scaled to show sources with emissions under 1×10^{30} mls/deg^{2}h.

To improve the reconstruction of sources, more frequent NO_2 data snapshots are needed. Such that the sources are not mainly based on the data of one NO_2 snapshot. In addition, the algorithm can take advantage of more knowledge on the transport parameters. In this Section all of the tests were conducted using the same transport parameters, in future work multiple transport parameters should be used. Finally, for more precise reconstruction of source locations using the current algorithm, more precise wind data is needed.

7 Conclusion

The objective of this report was to find out whether inverse time integration methods can be used to reconstruct NO_2 pollution sources. With the exact research question:

Can an algorithm based on an inverse time integration method be used to accurately reconstruct NO_2 pollution sources?

To answer this question, first the transport of NO_2 was modelled and transport parameters were estimated. It was concluded that a visual comparison alone is not adequate for an accurate estimation of transport parameters. Therefore, the found parameters were based on the stability of the numerical time integration method: Forward Euler, and the avoidance of numerical errors. These found parameters are set out in Table 4.
Afterwards, two different time integration methods were used as bases for source reconstruction algorithms. For the algorithm based on the Forward Euler method it was found that the method is capable of reconstructing sources when applied to artificial data, with a short period of time between data snapshots. It was additionally found that underestimation or overestimation of transport parameters had little effect on the location determination of sources, but a larger effect on the determination of the strengths of sources and the presence of noise around sources. When the algorithm was tested on artificial data created by a source function that was non-constant, it was found that is the source was 'off' before snapshot $\mathbf{y}_{n+m}$ was taken, the source was not reconstructed well. When the time between snapshots grew, the number of time steps also grew, this made the source reconstruction algorithm time- and computationally-intensive.
The second source reconstruction algorithm, based on the Backward Euler method, was capable of estimating the location and strengths of sources with 24 hours between NO_2 snapshots. It was found that the magnitude of the time steps plays a mayor role in the performance of the algorithm, as a smaller time step did not yield better results. In addition, similar trends were found for over- and underestimation of the diffusion coefficient as for the algorithm based on the Forward Euler method. The same holds for the effect of a non-constant source. Sources that did not emit any NO_2 in the 12 hours before $\mathbf{y}n + m$, were not reconstructed.
Finally, the application of the algorithm based on the Backward Euler method on experimental data has shown that the algorithm works for approximately determining the locations of emission zones. The algorithm is not capable of finding exact locations for individual sources and reconstructing sources with low emissions of less than 1×10^{30} mls/deg^{2}h.

This enables the formulation of an answer to the research question. The algorithms tested in this report based on Forward and Backward Euler, were capable of approximately reconstructing NO_2 sources. However, the algorithms were not able to accurately reconstruct emission rates and exact locations of sources. As there were only two time integration methods tested in this report, to exactly answer the research question, more research would be required exploring other algorithms and further analysing the ones implemented.

8 Discussion

As mentioned in the Introduction, the research conducted in this report is orientational, and thus it forms a basis for more research. In this Sections recommendations will be made for continuing research on this topic.
The first area where improvements can be made is on the available data. Currently the data on tropospheric NO_2 is given approximately daily and the wind data is given on an hourly basis. For the NO_2 data this means that there is a large amount of time between data snapshots, in this time the can be a lot of variations in emissions from sources. As found in Section 5.5.4, sources that only emit NO_2 during a few hours a day, are not reconstructed properly. For this to be possible more frequent data snapshots are needed. Good news on this topic is that there is a new satellite scheduled to be launched in 2023 which will produce hourly data on NO_2 concentrations. This will make it possible to reconstruct variable sources. At the same time, this highlights the problem of the hourly wind data, which is not precise enough to describe the wind speeds and directions between data snapshots. Thus, as more frequent NO_2 is coming, the need for more frequent wind data also rises.
Apart from the quality of the data, this research can also be applied to different types of data. The source reconstruction from satellite images is not as interesting for NO_2 as most sources are already known. Since the source reconstruction algorithm does not make assumptions on the locations of sources, it can be applied to source detection problems with no foreknowledge. For example, the detecting of methane leaks could be an interesting application.

The following point on which additional research could be beneficial is on the model itself. In this report a two-dimensional convection-diffusion equation is used to describe the transport of NO_2 in the troposphere. This equation has uses two main transport parameters: the diffusion coefficient D and the decay term α. Both of these parameters are taken to be constants, in future research it would be interesting to considered variable transport parameters. For example, the diffusion coefficient can be made dependent on the location, the wind speed or other weather factors such as temperature and humidity. In addition, the decay parameter can be taken variable, but this is of less importance.
Another part of the model that can be looked at is the convection term. In this report, the convection completely depends on the wind speed and direction. Part of the movement of clouds is governed by the rotation of the Earth, therefore the rotation of the Earth can be included in the convection term.

In Sections 5.3.2 and 5.5.3, the effect of errors in the transport parameters used for the source reconstruction algorithms is studied. From this analysis, no clear indications were found that point to whether the transport parameters were underestimated or overestimated. In this report, there was no focus on the area directly around the sources. In future studies, the effect should be investigated more closely by zooming in on the sources. Since it was found that the level of noise grew with an incorrectly chosen diffusion coefficient, it was not shown what shape this noise exactly has.

After discussing the data, and the model that is used to described the transport of NO_2, the source reconstruction algorithms will be discussed. In this report two different time integration methods are taken as bases for the algorithms. It was found that both algorithms had their flaws, which stem from instabilities. As the Forward Euler method is a conditionally stable method for propagating forward in time. Which resulted in a very small time step of approximately half a minute. On the other hand, the Backward Euler method is conditionally stable for propagating backwards in time. This resulted in the reverse problem, where a small time step did not give desirable results. Therefore, a time integration method is sought after which is unconditionally stable for both propagating forwards in time as backwards in time. A method that would be worth investigating, is the Trapezoidal method, which is a mixed method. This method can potentially balance out the instabilities resulting from the forward and backward propagation.

Finally, as mentioned in Section 5.3.4, it was found that the source reconstruction algorithm is sensitive to noise. It was also seen in the tests on experimental data in Section 6, that noise was present in the reconstructed source function. Therefore regularisation should be looked into. The most common regularisation method for linear ill-posed problems is Tikhonov regularization [25]. For this method to be useful, the level of noise in the data needs to be known. Therefore also other methods should be investigated, such as solving the problem over a limited solution space or to pre-process the data.
The effect of noise in the NO_2 data is briefly discussed, bit the presence of noise in the wind data is not assessed. Therefore in future research, the effect of noise in the wind data should be studied. With this,

an assessment of the level of noise in the wind data would be needed.

References

[1] Elizabeth A Ainsworth, Craig R Yendrek, Stephen Sitch, William J Collins, and Lisa D Emberson. The effects of tropospheric ozone on net primary productivity and implications for climate change. *Annual review of plant biology*, 63:637–661, 2012.

[2] Volkan Akcelik, George Biros, Andrei Draganescu, Omar Ghattas, Judith Hill, and Bart van Bloemen Waanders. Inversion of airborne contaminants in a regional model. In Vassil N. Alexandrov, Geert Dick van Albada, Peter M. A. Sloot, and Jack Dongarra, editors, *Computational Science – ICCS 2006*, pages 481–488, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[3] Juliana Atmadja and Amvrossios C. Bagtzoglou. Pollution source identification in heterogeneous porous media. *Water Resources Research*, 37(8):2113–2125, 2001.

[4] Alfred Carasso. The backward beam equation and the numerical computation of dissipative equations backwards in time. 1:48, 03 1975.

[5] A. Engström, F. A.-M. Bender, R. J. Charlson, and R. Wood. The nonlinear relationship between albedo and cloud fraction on near-global, monthly mean scale in observations and in the cmip5 model ensemble. *Geophysical Research Letters*, 42(21):9571–9578, 2015.

[6] EOP-GMQ. Sentinel-5 precursor calibration and validation plan for the operational phase, Nov 2017.

[7] EPA. Basic information about NO_2, Sep 2016.

[8] Palmes ED et al. Personal sampler for nitrogen dioxide. *American Industrial Hygiene Association journal*, 37:570–7, 10 1976.

[9] Men Habingabwa, Faustin Ndahayo, and Fredrik Berntsson. Air pollution tracking using pdes. *Rwanda Journal*, 27:63–69, 08 2012.

[10] Iolanda Ialongo, Henrik Virta, Henk Eskes, Jari Hovila, and John Douros. Comparison of tropomi/sentinel-5 precursor no¡sub¿2¡/sub¿ observations with ground-based measurements in helsinki. *Atmospheric Measurement Techniques*, 13:205–218, 01 2020.

[11] J. van Kan, S Segal, F Vermolen, and H. Kraaijevanger. *Numerical methods for partial differential equations*. Delft Academic Press, 2019.

[12] KNMI. Tropomi-metingen van stikstofdioxide (no2).

[13] S. Langenberg, T. Carstens, D. Hupperich, S. Schweighoefer, and U. Schurath. Technical note: Determination of binary gas-phase diffusion coefficients of unstable and adsorbing atmospheric trace gases at low temperature – arrested flow and twin tube method. *Atmospheric Chemistry and Physics*, 20(6):3669–3682, 2020.

[14] Sasha Madronich. Tropospheric photochemistry and its response to uv changes. In Marie-Lise Chanin, editor, *The Role of the Stratosphere in Global Change*, pages 437–461, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[15] Maria Markiewicz. *Modeling of the air pollution dispersion*, pages 303–348. Institute of Atomic Energy, 01 2006.

[16] Larry McNish. Latitude and longitude, Apr 2005.

[17] Lewis Fry Richardson. *Weather prediction by numerical process*, page 220–222. Cambridge, The University Press, 1922.

[18] Youcef Saad and Martin H. Schultz. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[19] Anita Singh and Madhoolika Agrawal. Acid rain and its ecological consequences. *Journal of Environmental Biology*, 29(1):15, 2007.

[20] Kunwar P. Singh, Shikha Gupta, and Premanjali Rai. Identifying pollution sources and predicting urban air quality using ensemble learning methods. *Atmospheric Environment*, 80:426–437, 2013.

[21] Valentin G. Stanev, Filip L. Iliev, Scott Hansen, Velimir V. Vesselinov, and Boian S. Alexandrov. Identification of release sources in advection–diffusion system by machine learning combined with green's function inverse method. *Applied Mathematical Modelling*, 60:64–76, 2018.

[22] H. A. van der Vorst. Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.

[23] Jos van Geffen, K. Folkert Boersma, Henk Eskes, Maarten Sneep, Mark ter Linden, Marina Zara, and J. Pepijn Veefkind. S5p tropomi no2 slant column retrieval: method, stability, uncertainties and comparisons with omi. *Atmospheric Measurement Techniques*, 13(3):1315–1335, 2020.

[24] J.P. Veefkind, I. Aben, K. McMullan, H. Förster, J. de Vries, G. Otter, J. Claas, H.J. Eskes, J.F. de Haan, Q. Kleipool, M. van Weele, O. Hasekamp, R. Hoogeveen, J. Landgraf, R. Snel, P. Tol, P. Ingmann, R. Voors, B. Kruizinga, R. Vink, H. Visser, and P.F. Levelt. Tropomi on the esa sentinel-5 precursor: A gmes mission for global observations of the atmospheric composition for climate, air quality and ozone layer applications. *Remote Sensing of Environment*, 120:70–83, 2012. The Sentinel Missions - New Opportunities for Science.

[25] Curtis R. Vogel. *Computational Methods for Inverse Problems*, volume 23 of *Frontiers in Applied Mathematics*. SIAM, January 2002.

Appendices

A Coordinate transformation

The received data from TNO is based on longitude and latitude, and thus is given in geographical coordinates. Since it is easier to work with a Cartesian coordinate system than with spherical coordinates, the longitude and latitude will be used as Cartesian coordinates. To substantiate this decision there will follow a short approximation.

The convection-diffusion equation with constant diffusion coefficient D and windvector $\mathbf{w}$ is the following:

$$\frac{\partial u}{\partial t} - D\Delta u + \mathbf{w} \cdot \nabla u = f$$

This equation can be written in spherical coordinates by taking the following:

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2} \\ x &= r\cos\phi\sin\theta \\ y &= r\sin\phi\sin\theta \\ z &= r\cos\theta \end{aligned}$$

This gives:

$$\frac{\partial u}{\partial t} - D\left[\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial u}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial u}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2 u}{\partial\phi^2}\right] + w_r\frac{\partial u}{\partial r} + \frac{w_\theta}{r}\frac{\partial u}{\partial\theta} + \frac{w_\phi}{r\sin\theta}\frac{\partial u}{\partial\phi} = f$$

Since the area that is taken into consideration is the surface or Earth, or at a constant level above the surface, we take $r = r_0$, this gives that $\frac{\partial u}{\partial r} = \frac{\partial u}{\partial r_0} = 0$. This means the equation can be simplified to the following:

$$\frac{\partial u}{\partial t} - D\left[\frac{1}{r_0^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial u}{\partial\theta}\right) + \frac{1}{r_0^2\sin^2\theta}\frac{\partial^2 u}{\partial\phi^2}\right] + \frac{w_\theta}{r_0}\frac{\partial u}{\partial\theta} + \frac{w_\phi}{r_0\sin\theta}\frac{\partial u}{\partial\phi} = f$$

To convert the spherical coordinates to geographical coordinates the latitude ($\hat{\theta}$) becomes: $\hat{\theta} = 90 - \theta = \frac{\pi}{2} - \theta$ and the longitude($\hat{\phi}$) becomes: $\hat{\phi} = \phi$. By the trigonometric identities, we have: $\sin\theta = \cos\frac{\pi}{2} - \theta = \cos\hat{\theta}$. Substituting this results in:

$$\frac{\partial u}{\partial t} - D\left[\frac{1}{r_0^2\cos\hat{\theta}}\frac{\partial}{\partial\hat{\theta}}\left(\cos\hat{\theta}\frac{\partial u}{\partial\hat{\theta}}\right) + \frac{1}{r_0^2\cos^2\hat{\theta}}\frac{\partial^2 u}{\partial\hat{\phi}^2}\right] + \frac{w_{\hat{\theta}}}{r_0}\frac{\partial u}{\partial\hat{\theta}} + \frac{w_{\hat{\phi}}}{r_0\cos\hat{\theta}}\frac{\partial u}{\partial\hat{\phi}} = f$$

Next, note that on the domain of the Netherlands the difference in latitude ($\hat{\theta}$) is small enough to consider $\cos\hat{\theta}$ as a constant, thus taking $\cos\hat{\theta} = \cos\hat{\theta}_0$. Which means it can be taken out of the derivative, this leads to the following:

$$\frac{\partial u}{\partial t} - D\left[\frac{1}{r_0^2}\frac{\partial^2 u}{\partial\hat{\theta}^2} + \frac{1}{r_0^2\cos^2\hat{\theta}_0}\frac{\partial^2 u}{\partial\hat{\phi}^2}\right] + \frac{w_{\hat{\theta}}}{r_0}\frac{\partial u}{\partial\hat{\theta}} + \frac{w_{\hat{\phi}}}{r_0\cos\hat{\theta}_0}\frac{\partial u}{\partial\hat{\phi}} = f$$

In addition, now $\cos\hat{\theta}$ is taken as a constant and since r is also a constant a final coordinate transformation can be made:

$$\begin{aligned} \hat{x} &= \hat{\phi}\cdot r_0\cdot\sin\hat{\theta}_0 \\ \hat{y} &= \hat{\theta}\cdot r_0 \end{aligned}$$

This gives:

$$\begin{aligned} \frac{\partial u}{\partial\hat{\theta}} = \frac{\partial u}{\partial\hat{y}}\frac{\partial\hat{y}}{\partial\hat{\theta}} \quad &\Rightarrow \quad \frac{\partial^2 u}{\partial\hat{\theta}^2} = \frac{\partial}{\partial\hat{y}}\left(\frac{\partial u}{\partial\hat{y}}\frac{\partial\hat{y}}{\partial\hat{\theta}}\right)\frac{\partial\hat{y}}{\partial\hat{\theta}} \\ &\Rightarrow \quad \frac{\partial^2 u}{\partial\hat{\theta}^2} = \frac{\partial^2 u}{\partial\hat{y}^2}\left(\frac{\partial\hat{y}}{\partial\hat{\theta}}\right)^2 + \frac{\partial u}{\partial\hat{y}}\frac{\partial^2\hat{y}}{\partial\hat{\theta}^2}\frac{\partial\hat{y}}{\partial\hat{\theta}} \end{aligned}$$

The derivatives of $\hat{y}$ can be calculated, this will give:

$$\frac{\partial \hat{y}}{\partial \hat{\theta}} = r \text{ and } \frac{\partial^2 \hat{y}}{\partial \hat{\theta}^2} = 0 \quad \Rightarrow \quad \frac{\partial u}{\partial \hat{\theta}} = r\frac{\partial \hat{y}}{\partial \hat{\theta}}$$
$$\frac{\partial^2 u}{\partial \hat{\theta}^2} = r^2\frac{\partial^2 u}{\partial \hat{y}^2}$$

In the same way it can be derived that:

$$\frac{\partial u}{\partial \hat{\phi}} = r\cos\hat{\theta}_0\frac{\partial^2 u}{\partial \hat{x}^2}$$
$$\frac{\partial^2 u}{\partial \hat{\phi}^2} = r^2\cos\hat{\theta}_0^2\frac{\partial^2 u}{\partial \hat{x}^2}$$

By substituting this, the partial differential equation can be further simplified to the following:

$$\frac{\partial u}{\partial t} - D\left[\frac{\partial^2 u}{\partial \tilde{\theta}^2} + \frac{\partial^2 u}{\partial \tilde{\phi}^2}\right] + w_{\tilde{\theta}}\frac{\partial u}{\partial \tilde{\theta}} + w_{\tilde{\phi}}\frac{\partial u}{\partial \tilde{\phi}} = f$$
$$\frac{\partial u}{\partial t} - D\Delta u + \mathbf{w}\cdot\nabla u = f$$

B Interpolation of unit vectors

In this section a proof will be given that the linear interpolation of two unit vectors does not in general give a unit vector. This is only the case when the angle between the two vectors is close to zero.

Proof. Let $\mathbf{u}_1, \mathbf{u}_2$ be two unit vectors, write: $\mathbf{u}_1 = \begin{pmatrix}\cos(\phi)\\ \sin(\phi)\end{pmatrix}, \mathbf{u}_2 = \begin{pmatrix}\cos(\hat{\phi})\\ \sin(\hat{\phi})\end{pmatrix}$. Then the linear interpolated point $\mathbf{v}$ is the following:

$$\mathbf{v} = \frac{\mathbf{u}_1 + \mathbf{u}_2}{2} = \frac{1}{2}\begin{pmatrix}\cos(\phi) + \cos(\hat{\phi})\\ \sin(\phi) + \sin(\hat{\phi})\end{pmatrix}$$

To show whether the vector $\mathbf{v}$ is a unit vector or not, the length of the vector will be calculated. We have that:

$$\begin{aligned}
|\mathbf{v}|^2 &= \left(\frac{1}{2}\left(\cos(\phi) + \cos(\hat{\phi})\right)\right)^2 + \left(\frac{1}{2}\left(\sin(\phi) + \sin(\hat{\phi})\right)\right)^2 \\
&= \frac{1}{4}\left(\cos^2(\phi) + 2\cos(\phi)\cos(\hat{\phi}) + \cos^2(\hat{\phi})\right) + \frac{1}{4}\left(\sin^2(\phi) + 2\sin(\phi)\sin(\hat{\phi}) + \sin^2(\hat{\phi})\right) \\
&= \frac{1}{2} + \frac{1}{2}\left(\cos(\phi)\cos(\hat{\phi}) + \sin(\phi)\sin(\hat{\phi})\right) \\
&= \frac{1}{2} + \frac{1}{2}\left(\frac{1}{2}\left(\cos(\phi + \hat{\phi}) + (\cos(\phi + \hat{\phi})\right) + \frac{1}{2}\left(\cos(\phi - \hat{\phi}) - (\cos(\phi + \hat{\phi})\right)\right) \\
&= \frac{1}{2} + \frac{1}{2}\cos(\phi - \hat{\phi})
\end{aligned}$$

In the fourth line the product identities of sine and cosine are used. This means that $\mathbf{v}$ is a unit vector if and only if $|\mathbf{v}|^2 = 1$, in total this means:

$$|\mathbf{v}| = 1 \iff \frac{1}{2} + \frac{1}{2}\cos(\phi - \hat{\phi}) = 1 \iff \cos(\phi - \hat{\phi}) = 1 \iff \phi = \hat{\phi}$$

Thus the interpolated vector $\mathbf{v}$ is a unit vector if and only if $\phi = \hat{\phi}$. □

This concludes that, only when the unit vectors that are to be interpolated have a similar angle, the resulting interpolated vector is a unit vector.

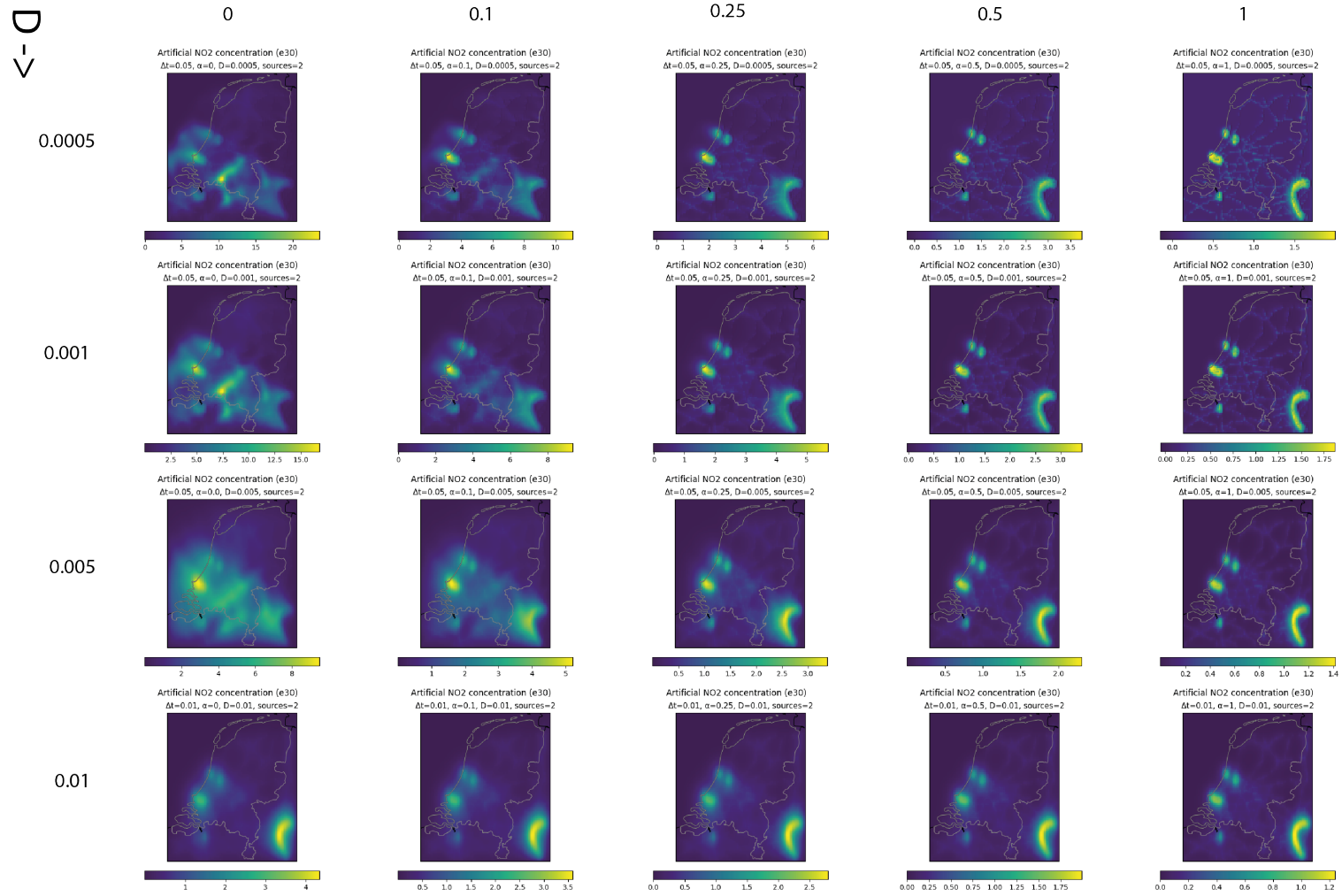


Figure 33: Comparison of the effects of the parameters α and D.