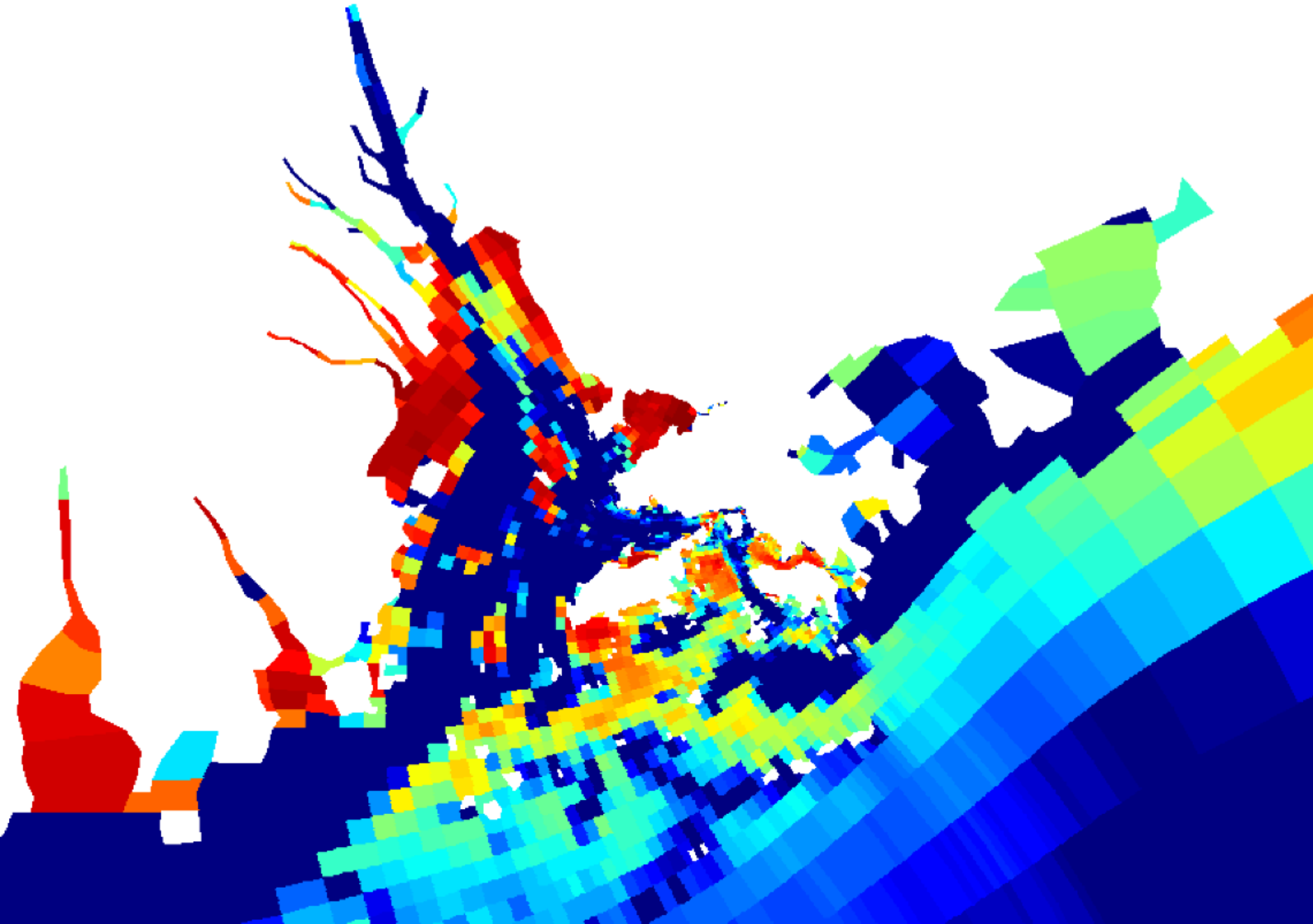
AN ACCURATE AND ROBUST FINITE VOLUME METHOD FOR THE ADVECTION DIFFUSION EQUATION

BY

PAULIEN VAN SLINGERLAND



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

AN ACCURATE AND ROBUST FINITE VOLUME METHOD FOR THE ADVECTION DIFFUSION EQUATION

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

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

PAULIEN VAN SLINGERLAND

Delft, The Netherlands,
June 2007

Thesis Committee:

Prof.dr.ir. C. Vuik (Delft University of Technology)
Ir. L. Postma (WL | Delft Hydraulics)
Dr. M. Genseberger (WL | Delft Hydraulics)
Dr. J.L.A. Dubbeldam (Delft University of Technology)

WL | delft hydraulics

TU Delft

Acknowledgements

Although there are many people that contributed to this M.Sc. project, I would like to take this oppertunity to acknowledge the kind support, reasonable criticism, and endless patience of my supervisors in particular. First of all, I would like to thank Kees Vuik for his valuable comments on a large number of concept versions of this thesis. I am very glad that the end of this thesis does not mean the end of our cooperation. Furthermore, I would like to give thanks to Leo Postma for giving me the oppertunity to study the interesting topic of water quality in the inspiring environment that WL | Delft hydraulics is. More than once, his different point of view gave me new insights in the problem. In addition, I would like to express my sincere gratitude to Menno Genseberger for his daily input and advise, that demonstrates an eye for perfectionism. His analytical thinking has helped me through many frustrating moments involving Fortran and mathematical proofs. He also introduced me to Mart Borsboom, who soon became my fourth supervisor. I owe Mart much gratitude, as his creative ideas eventually led to the local-theta scheme, which forms the key to the main problem that is considered in this thesis. It has been a great pleasure to listen to his sensible suggestions in a room filled with stuffed animals.

Contents

Acknowledgements i

1 Introduction 1

I Constructing the numerical model 3

2 Modeling water quality 5
2.1 Physical water quality model . . . 5
2.1.1 Transport . . . 5
2.1.2 Water quality processes . . . 6
2.2 Mathematical water quality model . . . 6
2.3 Summary . . . 7

3 Finite Volume Method 9
3.1 Grid . . . 9
3.2 Integral form . . . 10
3.3 Finite volume method . . . 13
3.4 The quality of a finite volume method . . . 14
3.4.1 Accuracy . . . 14
3.4.2 Robustness . . . 15
3.5 Summary . . . 17

4 Accurate explicit schemes 19
4.1 Local extremum diminishing flux functions . . . 19
4.2 Explicit schemes . . . 21
4.3 Flux corrected transport . . . 22
4.4 Summary . . . 27

5 Robust implicit theta schemes 29
5.1 Theta scheme . . . 29
5.2 Theta FCT scheme . . . 32
5.3 Summary . . . 38

6 Local-theta scheme 39
6.1 Local-theta scheme . . . 39
6.2 Flux corrected transport . . . 43
6.3 Molenkamp problem . . . 47
6.4 Revisiting Hong Kong . . . 48
6.5 Final implementation in WAQ . . . 51
6.6 Summary . . . 51

7 Summary & Recommendations **55**
7.1 Recommendations . . . 56

II Solving the numerical model **57**

8 Solution methods for linear systems **59**
8.1 Direct methods . . . 59
8.1.1 Triangular matrices . . . 59
8.1.2 General square matrices . . . 60
8.2 Iterative Methods . . . 61
8.2.1 Linear fixed point iteration . . . 61
8.2.2 Krylov methods . . . 63
8.3 Summary . . . 67

9 Preconditioning **69**
9.1 Basic preconditioning . . . 69
9.2 Preconditioners based on matrix splitting . . . 70
9.3 Preconditioners based on an incomplete LU factorisation . . . 71
9.3.1 Incomplete LU threshold . . . 71
9.3.2 Incomplete LU . . . 72
9.4 Summary . . . 74

10 Reordering **75**
10.1 Symmetric permutation . . . 75
10.2 Renumbering the adjacency graph . . . 76
10.2.1 Level-set orderings . . . 76
10.2.2 Independent set ordering . . . 78
10.2.3 Multicolor orderings . . . 79
10.3 Summary . . . 79

11 Storage of sparse matrices **81**
11.1 Coordinate format . . . 81
11.2 Compressed sparse row format . . . 82
11.3 Summary . . . 83

12 Summary **85**

A Current schemes of WAQ **87**

Chapter 1

Introduction

At present, plans are being made for the construction of Liquefied Natural Gas pipes in the sea bed off the coast of Hong Kong. To this end, dredging is necessary which causes plumes of silt in the water. The silt particles float in the water for a relatively long period of time, until they settle on the sea bed eventually. Unfortunately, both phenomena are harmful to coral reefs and Chinese white dolphins, two protected species that live in the sea near to Hong Kong. For this reason, before the plans can be carried out, it is necessary to determine how much of the ocean may be affected by those plumes.

Water is indispensable for all organisms. People use it for drinking, fishing, bathing, irrigating, shipping, and so on. As a consequence, it is very important that its quality is maintained. The quality of water is determined by the concentrations of the substances it contains, such as oxygen, salts, silt and bacteria. From the example above it is clear that it could easily be diminished. The question is: could this be foreseen?

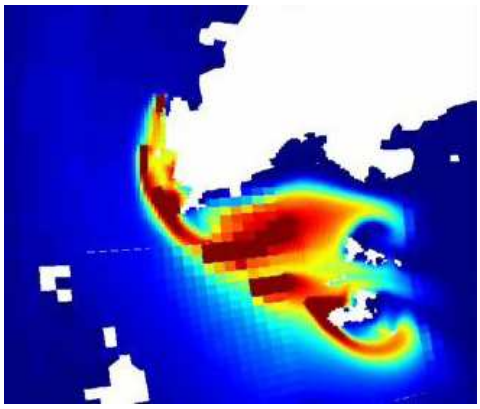
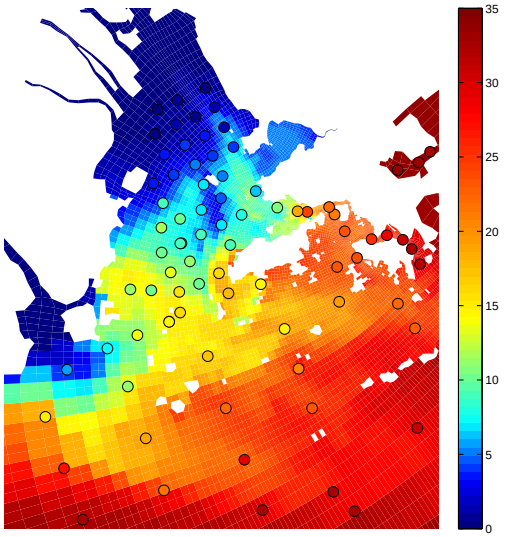


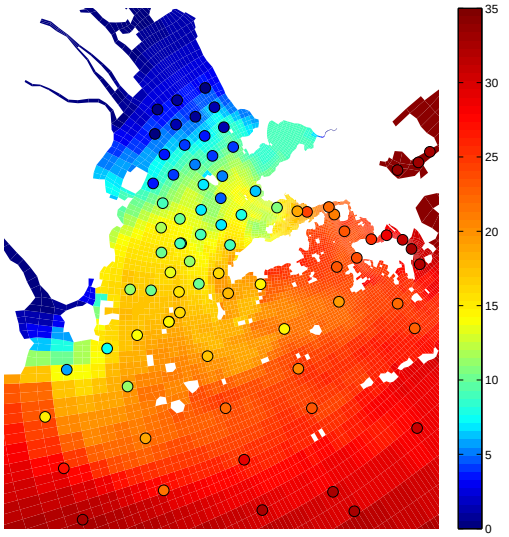
Figure 1.1: Forecast by Delft3D-WAQ of silt plumes near Hong Kong

Fortunately, software is already available for this purpose. Delft3D-WAQ, a simulation program that has been developed by WL | Delft Hydraulics, is a useful tool in forecasting water quality. In particular, it is able to predict the size of silt plumes caused by dredging (see Figure 1.1). Basically, the software approximates the solution of the advection diffusion reaction equation by means of the finite volume method. Since it is often necessary to predict one or two years ahead, large time steps are preferred in order to have limited computing time.

However, there are two aspects that need improvement. First of all, the current schemes are either



(a) Accurate explicit scheme (Scheme 12, see Appendix A), computational time ≈ 176 min.



(b) Robust implicit scheme (Scheme 16, see Appendix A), computational time ≈ 9 min.

Figure 1.2: Simulation by Delft3D-WAQ of salinity in an estuary near Hong Kong. The colors of the cicles indicate measured values.

explicit higher order schemes that are not robust (see Figure 1.2(a)), or implicit first order schemes that are inaccurate (see Figure 1.2(b)). Moreover, the convergence speed of the present iterative solver for linear systems is unsatisfactory for diffusion dominated problems. In other words, in this report, answers to the following question will be sought:

1. *Is it possible to construct a finite volume scheme for the advection diffusion equation that is both accurate and robust?*
2. *Can the convergence speed of the present iterative solver for linear systems be increased for diffusion dominated problems?*

In other words: can the damage done to dolphins and coral reefs be estimated better and faster?

The first part of this thesis, in which the numerical model is constructed, provides an answer to the first question. In Chapter 2, a physical water quality description is translated into a mathematical model that is based on the advection diffusion reaction equation. The solution to this model can be approximated by means of the finite volume method, which is discussed in Chapter 3. Chapter 4 considers accurate explicit flux correcting transport schemes. Robust theta schemes are considered in Chapter 5. In Chapter 6, the local-theta scheme is considered, which attempts to combine the advantages of the previous methods, to obtain a scheme that is both accurate and robust. A summary and recommendations can be found in Chapter 7.

The second part of this thesis describes some basic theory for solving the numerical model, which forms a first step towards the answer of the second question. Implicit variants of the finite volume method require the solution of many large linear systems. In order to solve these systems efficiently, iterative solvers are considered in Chapter 8. Useful tools in improving the performance of iterative schemes are preconditioning (Chapter 9) and reordering of the matrix (Chapter 10). Chapter 11 discusses storage schemes for sparse matrices that can save both memory and time. A summary is given in chapter 12.

Part I

Constructing the numerical model

Chapter 2

Modeling water quality

2.1 Physical water quality model

The quality of water is determined by the concentrations of the substances it contains, such as oxygen, algae, salts, bacteria, viruses, toxic heavy metals, pesticides, and silt. These concentrations can be affected in two ways. Firstly, particles can be transported through the water in several ways. Moreover, water quality processes play an important role. Both phenomena will be discussed briefly below.

2.1.1 Transport

A substance can be transported by diffusion, advection, and by an own movement that is independent of the preceding types of transport.

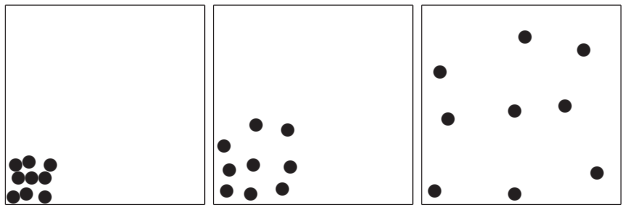


Figure 2.1: Molecular diffusion

Molecular diffusion is the spontaneous spreading of matter due to the random movement of molecules. Figure 2.1 displays a schematic visualisation of this mixing process.

Advection is the transport of a substance due to the motion of the fluid. The flow carries the particles in the downstream direction. Figure 2.2 illustrates how turbulent flow can lead to what is called *turbulent mixing*. Although turbulent mixing is a result of advection, it is often modeled as diffusion. For this reason, it is sometimes referred to as *turbulent diffusion*.

Own movement is any movement that is not caused by advection or diffusion. This kind of movement could be forced by gravity, the substance itself, or the wind. *Gravitational movement* arises when there is a difference between the density of the substance and that of the water. Silt, for example, is heavier than water. Therefore, it will generally have an extra downward motion. *Active movement* only applies to organisms that can 'swim' in some sense. Examples are shrimps, fish, and certain algae that can propel themselves through the water. *Floating movement* is the

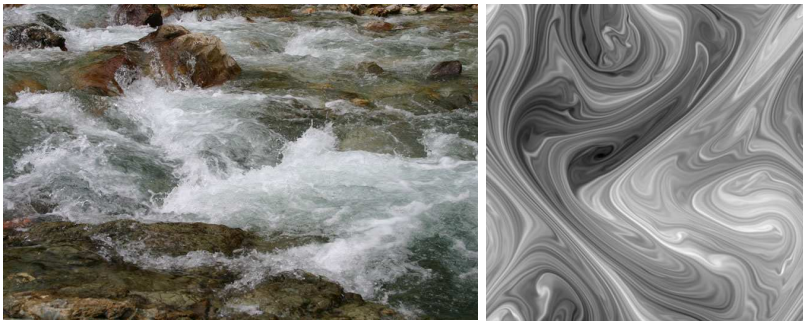


Figure 2.2: Turbulent mixing

motion that a floating substance obtains from the wind. As a result, its concentration is generally higher on the downwind water surface side of the area.

2.1.2 Water quality processes

Apart from transport, water quality processes can have a great effect on the concentration of a substance. These processes involve the interaction between the substances. Examples are photosynthesis, mineralisation, sedimentation, nitrification, and the mortality of bacteria. For a detailed description of these processes, see e.g. [man05, Chapter 8] or [Pos05].

2.2 Mathematical water quality model

According to the physical description above, water quality is affected by transport and water quality processes. Transport due to advection and own movement can be modeled by one advection term. Adding the molecular diffusion and the water quality processes to the model, the mathematical water quality model boils down to the advection diffusion reaction equation, one for each substance that needs to be modeled.

Model 2.1 (**Water quality model**)**:** Consider a substance that is dissolved in flowing water. Let $c(\underline{\mathbf{x}},t)$ denote its (unknown) concentration, $d(\underline{\mathbf{x}},t)$ its molecular diffusion coefficient, and $\underline{\mathbf{u}}(\underline{\mathbf{x}},t)$ its velocity due to advection[1] and own movement. Let $p(\underline{\mathbf{x}},t)$ represent all relevant water quality processes. p may also depend on c or on the concentration of other substances. For this substance, the water quality model reads:

$$\begin{cases} \frac{\partial c}{\partial t}(\underline{\mathbf{x}},t) + \nabla \cdot \big(\underline{\mathbf{u}}(\underline{\mathbf{x}},t)c(\underline{\mathbf{x}},t) - d(\underline{\mathbf{x}},t)\nabla c(\underline{\mathbf{x}},t)\big) &= p(\underline{\mathbf{x}},t), \\ c(\underline{\mathbf{x}},0) &= \mathring{c}\,(\underline{\mathbf{x}}), \\ c|_{\underline{\mathbf{x}}\in\partial\mathcal{D}_1} &= \breve{c}(\underline{\mathbf{x}},t), \\ (\nabla c\cdot\underline{\mathbf{n}})|_{\underline{\mathbf{x}}\in\partial\mathcal{D}_2} &= 0. \end{cases} \tag{2.1}$$

Here, $t\in[0,T]$ and $\underline{\mathbf{x}}\in\mathcal{D}\subset\mathbb{R}^m$ (m=1,2,3). $\mathring{c}\,(\underline{\mathbf{x}})$ is the initial condition. The boundary of $\mathcal{D}$ is partitioned according to $\partial\mathcal{D}=\partial\mathcal{D}_1\cup\partial\mathcal{D}_2$. Here, $\partial\mathcal{D}_2$ is the part of the boundary through which no transport takes place (the shore) which is modeled with the help of a homogeneous *Neumann boundary condition*. On $\partial\mathcal{D}_1$, a *Dirichlet boundary condition* is imposed.[2] $\underline{\mathbf{n}}$ is the outward normal unit vector on $\partial\mathcal{D}$. ⌟

[1] In general the velocity due to advection is conform the velocity profile of the water. However, it is possible that a substance is not being advected, but e.g. lying on the bottom.

[2] Of course, there are other types of boundary conditions, but these are not considered in this report.

2.3 Summary

The quality of water is determined by the concentrations of the substances it contains. These concentrations can be affected by transport and water quality processes. The corresponding mathematical model is the advection diffusion reaction equation, one for each substance that needs to be simulated.

Chapter 3

Finite Volume Method

In general, it is impossible to obtain the analytical solution of the water quality model (Model 2.1). Fortunately, a numerical approximation can be computed by means of the finite volume method. The two main ingredients of the finite volume method are an integral form and a subdevision of the spatial domain into 'finite volumes'. Roughly speaking, the finite volume method approximates the integral form for each of those volumes. After that, the resulting system is solved to obtain an approximation of the solution of the original model. Good references on the finite volume method are, for instance, [BO04], [God96, Chapter 4], and [Krö97, Chapter 3].

3.1 Grid

The first step towards a finite volume approximation is the subdivision of the spatial domain into smaller volumes. These volumes will be referred to as grid cells, although they need not be stacked in any regular way. In each grid cell, the average concentration of a substance is considered, which forms a good approximation for the concentration at the center of the volume.

Definition 3.1 (Cell centered grid): A *cell centered grid* of a spatial domain $\mathcal{D} \subset \mathbb{R}^d$ consists of a set of closed control volumes $\mathcal{V} = \{\mathcal{V}_i \subset \mathcal{D} : i = 1, ..., I\}$ and a set of storage locations $\mathcal{X} = \{\underline{\mathbf{x}}_i \in \mathcal{D} : i = 1, ..., I\}$ such that

1. $\underline{\mathbf{x}}_i$ is at the center of mass of $\mathcal{V}_i$;
2. the volumes cover the entire spatial domain:

$$\mathcal{D} = \bigcup_{i=1}^{I} \mathcal{V}_i;$$

3. the volumes do not overlap in the sense that, for all $i \neq j$, either

$$\mathcal{V}_i \cap \mathcal{V}_j = \varnothing,$$

or, if the volumes are adjacent,

$$\mathcal{V}_i \cap \mathcal{V}_j = \partial\mathcal{V}_i \cap \partial\mathcal{V}_j,$$

where $\partial\mathcal{V}_i$ denotes the boundary of $\mathcal{V}_i$.

The grid is denoted as $G = (\mathcal{V}, \mathcal{X})$. In the one-dimensional case, the grid is chosen so that $x_1 < x_2 < ... < x_I$. ⌟

Remark 3.2 (Grid in WAQ)**:** WAQ often receives the velocity profile from Delft3D-FLOW, another simulation program that has been developed by WL | Delft Hydraulics. This program simulates water flow by computing an approximate solution of the shallow water equations. FLOW can handle two types of structured grids, which are both curvilinear and staggered in the horizontal direction. Figure 3.1 illustrates the different approaches in the vertical direction. The *σ-grid* uses a fixed number of time dependent boundary-fitted layers. The top layer fits the water surface, whereas the bottom layer fits the sea bed. A *z-grid* uses time-independent horizontal layers. Unlike the σ-grid, the number of (active) cells per column may vary.

WAQ's grid results from aggregating adjacent cells of the mesh that is used by FLOW. From the example in Figure 3.2 it becomes clear that this generally leads to an unstructured grid. Some schemes in WAQ require some structure though. In practice, the grid is usually strongly non-uniform. Moreover, the cells may be a thousand times as wide as they are high. ⌟

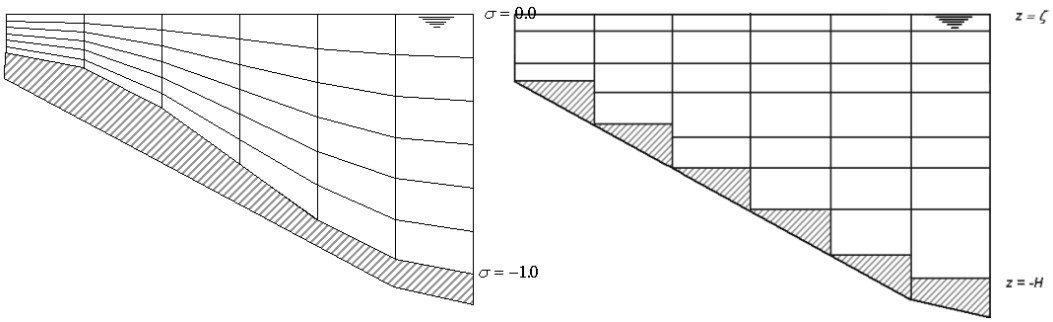


Figure 3.1: Side view of the structured grid types of FLOW: σ-grid (left) and z-grid (right)

3.2 Integral form

The second main component of the finite volume method is an integral form of the model, which basically results from integrating the model equation.

Definition 3.3 (**Integral formulation of the water quality model**)**:** Consider Model 2.1. Let

$$G = (\{\mathcal{V}_1, ..., \mathcal{V}_I\}, \{\underline{\mathbf{x}}_1, ..., \underline{\mathbf{x}}_I\})$$

be a (time-dependent) cell-centered grid for the (time-dependent) space domain $\mathcal{D}$. If K grid cells are adjacent to the open boundary $\partial\mathcal{D}_1$, introduce K adjacent virtual cells $\mathcal{V}_{I+1}, ..., \mathcal{V}_{I+K}$. Let $\mathcal{J}_i$ contain the indices of the neighbors of grid cell $\mathcal{V}_i$:

$$\mathcal{J}_i = \{j \in \{1, ..., I + K\} : \mathcal{V}_i \cap \mathcal{V}_j \neq \varnothing\}.$$

Let $\mathcal{S}_{ij}$ denote the joint boundary of neighboring grid cells $\mathcal{V}_i$ and $\mathcal{V}_j$:

$$\mathcal{S}_{ij} = \partial\mathcal{V}_i \cap \partial\mathcal{V}_j.$$

Let $\underline{\mathbf{n}}_{ij}$ be the unit normal vector on $\mathcal{S}_{ij}$ that points in the direction of $\mathcal{V}_j$. The integral form of Model 2.1 reads

$$\frac{d}{dt}\int_{\mathcal{V}_i} c\, d\underline{\mathbf{x}} + \sum_{j\in\mathcal{J}_i}\int_{\mathcal{S}_{ij}} \left(\underline{\mathbf{u}}c - d\nabla c\right)\cdot\underline{\mathbf{n}}_{ij}\, d\underline{\mathbf{x}} = \int_{\mathcal{V}_i} p\, d\underline{\mathbf{x}}. \tag{3.1}$$

⌟

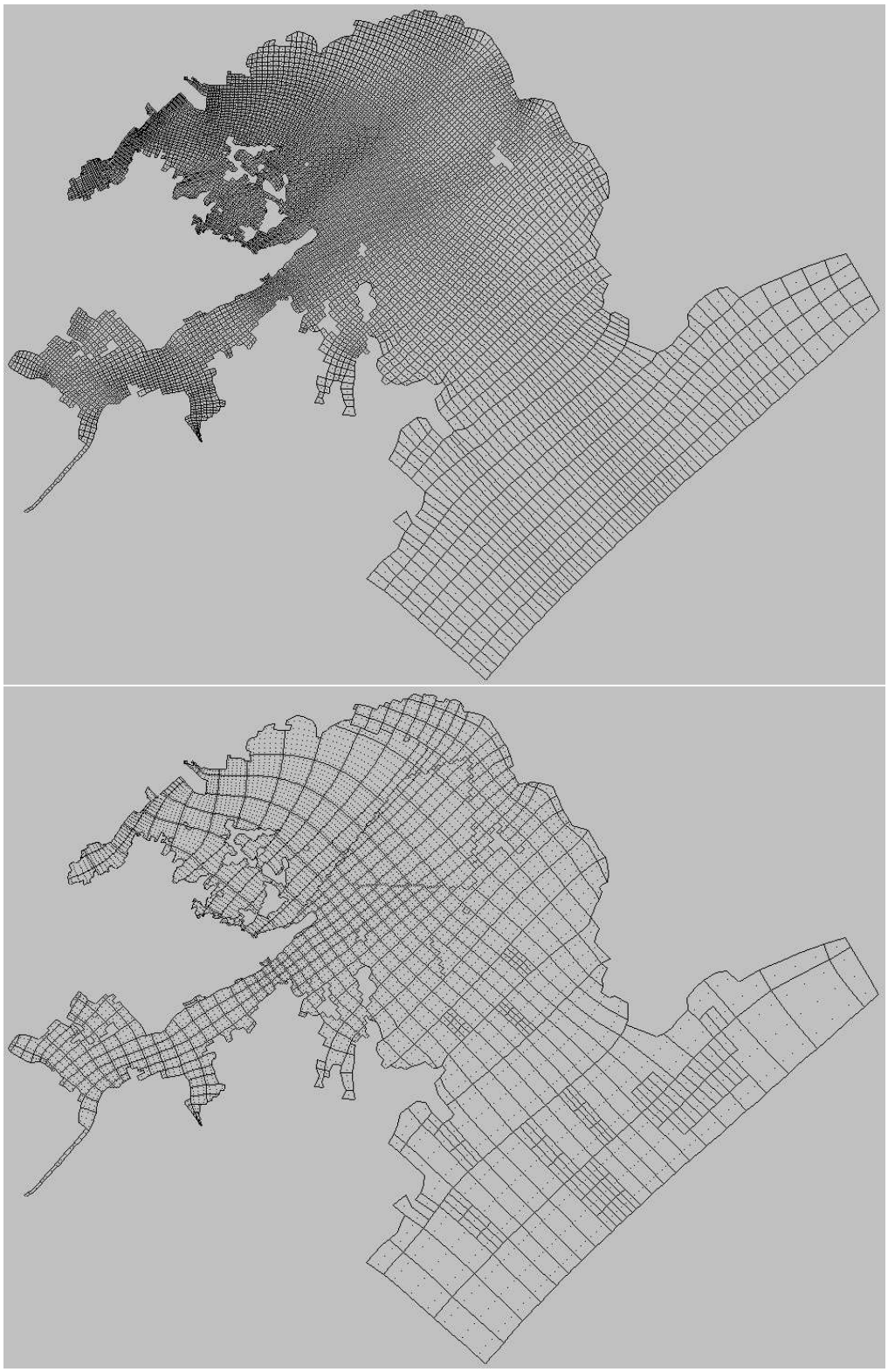


Figure 3.2: An example of a grid in WAQ (bottom), resulting from the aggregation of grid cells of the grid used by FLOW (top)

WAQ works with average values of the velocity on the faces of the grid cells. For this reason, below, the integral form is rewritten in terms of these average velocities, by expressing the velocity as the sum of the average velocity and the deviation of this mean value. After separating these two components in the model, it is assumed that the remaining pure deviation terms can be modeled as an additional diffusion term.

Proposition 3.4: *Consider the integral form that was defined in Definition 3.3. Define the following boundary averages and their deviations:*

$$\begin{aligned}
\overline{c}_{ij}(t) &= \frac{1}{|\mathcal{S}_{ij}|}\int_{\mathcal{S}_{ij}} c(\underline{\mathbf{x}},t)\,d\underline{\mathbf{x}},\\
\overline{\underline{\mathbf{u}}}_{ij}(t) &= \frac{1}{|\mathcal{S}_{ij}|}\int_{\mathcal{S}_{ij}} \underline{\mathbf{u}}(\underline{\mathbf{x}},t)\,d\underline{\mathbf{x}},\\
\tilde{c}_{ij}(\underline{\mathbf{x}},t) &= c(\underline{\mathbf{x}},t)-\overline{c}_{ij}, \qquad \underline{\mathbf{x}}\in\mathcal{S}_{ij},\\
\tilde{\underline{\mathbf{u}}}_{ij}(\underline{\mathbf{x}},t) &= \underline{\mathbf{u}}(\underline{\mathbf{x}},t)-\overline{\underline{\mathbf{u}}}_{ij}, \qquad \underline{\mathbf{x}}\in\mathcal{S}_{ij},
\end{aligned}$$

where $|\mathcal{S}_{ij}|$ denotes the surface area of $\mathcal{S}_{ij}$. Then, $\forall i = 1,...,I$:

$$\frac{d}{dt}\int_{\mathcal{V}_i} c\,d\underline{\mathbf{x}} + \sum_{j\in\mathcal{J}_i}\left(|\mathcal{S}_{ij}|\overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} - \int_{\mathcal{S}_{ij}}\left(d\nabla c - \tilde{c}_{ij}\tilde{\underline{\mathbf{u}}}_{ij}\right)\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}\right) = \int_{\mathcal{V}_i} p\,d\underline{\mathbf{x}}. \tag{3.2}$$

Proof. The advection term can be rewritten according to:

$$\begin{aligned}
\int_{\mathcal{S}_{ij}} c\underline{\mathbf{u}}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}} &= \int_{\mathcal{S}_{ij}} \overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}} + \int_{\mathcal{S}_{ij}} \tilde{c}_{ij}\tilde{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}\\
&\quad + \int_{\mathcal{S}_{ij}} \overline{c}_{ij}\tilde{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}} + \int_{\mathcal{S}_{ij}} \tilde{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}.
\end{aligned}$$

Since the average deviation of the average is zero, this reduces to:

$$\begin{aligned}
\int_{\mathcal{S}_{ij}} c\underline{\mathbf{u}}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}} &= \int_{\mathcal{S}_{ij}} \overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}} + \int_{\mathcal{S}_{ij}} \tilde{c}_{ij}\tilde{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}\\
&= |\mathcal{S}_{ij}|\overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} + \int_{\mathcal{S}_{ij}} \tilde{c}_{ij}\tilde{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}.
\end{aligned}$$

Substitution ends the proof. ■

Assumption 3.5: The term $\frac{1}{|\mathcal{S}_{ij}|}\int_{\mathcal{S}_{ij}} \tilde{c}_{ij}\tilde{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}$ in (3.2) can be interpreted as turbulence on a sub-grid scale [Pos05, p.27]. It is assumed that this term can be modeled as an additional diffusion term, i.e. $\exists\,\underline{\underline{\tilde{D}}} : \mathcal{D}\times[0,T]\to\mathbb{R}^{m\times m}$ such that (3.2) is equivalent to:

$$\frac{d}{dt}\int_{\mathcal{V}_i} c\,d\underline{\mathbf{x}} + \sum_{j\in\mathcal{J}_i}\left(|\mathcal{S}_{ij}|\overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} - \int_{\mathcal{S}_{ij}}\left(\underline{\underline{\tilde{D}}}\,\nabla c\right)\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}\right) = \int_{\mathcal{V}_i} p\,d\underline{\mathbf{x}}. \tag{3.3}$$

Note that the molecular diffusion coefficient d, has been included in $\underline{\underline{\tilde{D}}}$. ⌟

Remark 3.6 (Magnitude of additional diffusion)**:** The order of magnitude of the additional diffusion, that results form the assumption above, depends on size of the grid cells and on the magnitude of the velocity. Typically, WAQ uses an additional diffusion of the following order of magnitude:

m	order of magnitude of the additional diffusion
1	$1000\,\mathrm{m}^2\,\mathrm{s}^{-1}$
2	$10\,\mathrm{m}^2\,\mathrm{s}^{-1}$
3	$1\,\mathrm{m}^2\,\mathrm{s}^{-1}$

Normally, the molecular diffusion coefficient d lies between $0\,\mathrm{m}^2\,\mathrm{s}^{-1}$ and $1\,\mathrm{m}^2\,\mathrm{s}^{-1}$, regardless of the dimension. Therefore, diffusion dominated problems mainly occur in the one- and two-dimensional cases. ⌟

3.3 Finite volume method

The finite volume method uses separate spatial and time discretisation. First, spatial discretisation is applied to approximate the integral form in terms of cell averages and numerical fluxes. After that, the time domain is discretised to solve the resulting system of ordinary differential equations.

Method 3.7 (Finite Volume Method (FVM)): The *Finite volume method* obtains an approximation of the solution of Model 2.1 in the following manner:

1. Approximate (3.3) by

$$\frac{d|\mathcal{V}_i|\overline{c}_i}{dt} + \sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}|\phi_{ij}(\overline{c}_i,\overline{c}_j) = |\mathcal{V}_i|\overline{p}_i, \qquad i=1,...,I, \tag{3.4}$$

   where $|\mathcal{V}_i|$ denotes the volume of $\mathcal{V}_i$, and $\overline{c}_i$ and $\overline{p}_i$ denote the following cell averages:

$$\begin{aligned} \overline{c}_i(t) &= \begin{cases} \frac{1}{|\mathcal{V}_i|}\int_{\mathcal{V}_i} c(\underline{\mathbf{x}},t)\,d\underline{\mathbf{x}}, & i=1,..,I, \\ \frac{1}{|\mathcal{V}_i\cap\partial\mathcal{D}_1|}\int_{\mathcal{V}_i\cap\partial\mathcal{D}_1} \breve{c}(\underline{\mathbf{x}},t)\,d\underline{\mathbf{x}}, & i=I+1,..,I+K, \end{cases} \\ \overline{p}_i(t) &= \frac{1}{|\mathcal{V}_i|}\int_{\mathcal{V}_i} p(\underline{\mathbf{x}},t)\,d\underline{\mathbf{x}}, \qquad i=1,...,I, \end{aligned}$$

   and ϕ_{ij} is a so-called *numerical flux function*[1]:

$$\phi_{ij}(\overline{c}_i,\overline{c}_j) \approx \overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} - \frac{1}{|\mathcal{S}_{ij}|}\int_{\mathcal{S}_{ij}} \left(\underline{\underline{\tilde{D}}}\,\nabla c\right)\cdot\underline{\mathbf{n}}_{ij}\,d\underline{\mathbf{x}}, \tag{3.5}$$

2. Discretise the time domain as follows:

$$0 = t_0 < t_1 < ... < t_N \le T,$$

   and apply an ODE solver (see e.g. [BF01, Chapter 5]) to (3.4) to obtain a system of the form[2]

$$g_i^n(\overline{c}_1^{n-1},...,\overline{c}_{I+K}^{n-1},\overline{c}_1^n,...,\overline{c}_{I+K}^n) = 0, \qquad i=1,...,I;\ n=1,...,N. \tag{3.6}$$

   Here, $\overline{c}_i^n$ is short for $\overline{c}_i(t_n)$. In the remaining of this part of the thesis, a generalisation of this convenient notation will be used, i.e. $q^n := q(t_n)$ for each quantity $q(t)$. The solution of this system approximates the solution of the model:

$$c(\underline{\mathbf{x}}_i,t_n) \approx \overline{c}_i^n, \qquad i=1,...,I;\ n=1,...,N. \qquad \lrcorner$$

Remark 3.8 (Numerical flux in WAQ)**:** As a rule, WAQ uses a central difference approach for the diffusion term and a separate finite difference approach for the advection term. More precisely, the numerical flux function ϕ_{ij} is of the form:

$$\phi_{ij} = \psi_{ij} - \overline{d}_{ij}\frac{\overline{c}_j-\overline{c}_i}{\|\underline{\mathbf{x}}_j-\underline{\mathbf{x}}_i\|_2},$$

where $\overline{d}_{ij}=\overline{d}_{ji}$ represents the total amount of diffusion between $\mathcal{V}_i$ and $\mathcal{V}_j$, and $\psi_{ij}\approx\overline{c}_{ij}\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}$ approximates the advection term by means of a certain finite difference approach. $\lrcorner$

Remark 3.9 (Dealing with nonlinearity)**:** Note that p may depend nonlinearly on the concentration of any substance that is included in the water quality model. In order to avoid the necessity of solving a complex nonlinear system, WAQ uses the following strategy.

[1] ϕ_{ij} may also depend on concentrations other than $\overline{c}_i$ and $\overline{c}_j$. However, such flux functions are not considered in this thesis.

[2] g_i^n may also depend on other times than t_{n-1} and t_n. However, such time-discretisations are not considered in this thesis.

1. Suppose that $\overline{c}_i^{n-1}$ is known. Initially, leave transport out of consideration, and deal with the water quality processes in an explicit manner. In other words, compute intermediate states $\hat{\overline{c}}_i^n$ by means of:

$$\frac{|\mathcal{V}_i^n|\hat{\overline{c}}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} = |\mathcal{V}_i^{n-1}|\overline{p}_i^{n-1}, \qquad i = 1, ..., I.$$

2. After that, compute $\overline{c}_i^n$ by solving

$$\frac{d|\mathcal{V}_i|\overline{c}_i}{dt} = -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}|\phi_{ij}(\overline{c}_i, \overline{c}_j), \qquad i = 1, ..., I,$$

with the help of an ODE solver (see e.g. Method 4.4 or Method 5.1) using $\hat{\overline{c}}_i^n$ instead of $\overline{c}_i^{n-1}$.

Note that this fractional step approach introduces an error. On the other hand, it involves linear systems only, as long as the flux functions are linear. ⌟

The remainder of this part of the thesis will focus on the second step of the strategy above.

3.4 The quality of a finite volume method

The quality of a finite volume scheme is determined by a combination of accuracy and robustness. These topics are discussed briefly below.

3.4.1 Accuracy

Several properties are related to accuracy. First of all, the scheme should not conflict with the law of conservation of mass. Usually, mass conservation is a result of anti-symmetrical flux functions ($\phi_{ij} = -\phi_{ji}$).

Definition 3.10 (**Mass conservation**)**:** Consider Method 3.7 for $p = 0$ (no processes) and $\partial D_1 = \varnothing$ (no open boundary[3]). The scheme is *mass conservative*, if the total amount of mass in the interior grid cells does not change in time:

$$\textstyle\sum_{i=1}^I |\mathcal{V}_i^n|\overline{c}_i^n = \sum_{i=1}^I |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}, \qquad n = 1, ..., N$$ ⌟

Additionally, the method should yield the exact solution for infinitely small grid cells and time steps. In that case, the method is called convergent.

Definition 3.11 (**Global truncation error**)**:** The *global truncation error* of Method 3.7 at time t_n is defined as:

$$e_i^n = c(\underline{\mathbf{x}}_i, t_n) - \overline{c}_i^n, \qquad i = 1, ..., I.$$ ⌟

Definition 3.12 (**Convergence**)**:** Consider the Method 3.7. Let the spatial mesh sizes and the time steps be decreasing functions of a parameter h. The method *converges* at time t_n with respect to some norm $\|.\|$, if

$$\lim_{h\downarrow 0} \|\underline{\mathbf{e}}^n\| = 0, \qquad t_n, \underline{\mathbf{x}}_1, ..., \underline{\mathbf{x}}_I \text{ fixed.}$$

Here, $\underline{\mathbf{e}}^n$ is the vector containing the global truncation errors. ⌟

In practice, the global truncation error is (generally) not equal to zero, since the grid cells and the time steps are not infinitely small. Therefore, it is convenient to have an indication of the accuracy, which is provided below for the one-dimensional equidistant case.

[3] In case of an open boundary, the change in mass should match the transport through the open boundary.

Definition 3.13 (Order of accuracy): Consider the one-dimensional variant ($m = 1$) of Method 3.7 with constant time step Δt and constant cell width Δx. The method is said to be s_1 *order accurate* in time and s_2 order accurate in space with respect to some norm $\|.\|$, if

$$\|\underline{\mathbf{e}}^n\| \quad = \quad O(\Delta t^{s_1}) + O(\Delta x^{s_2}).$$

Here, $\underline{\mathbf{e}}^n$ is the vector containing the global truncation errors. ⌟

Intuitively, convergence can only occur if the local truncation error is small enough. This condition is called consistency.

Definition 3.14 (Local truncation error): The *local truncation error* of Method 3.7 at time t_n is defined as:

$$\tilde{e}_i^n \quad = \quad g_i^n\big(c(\underline{\mathbf{x}}_1, t_{n-1}), ..., c(\underline{\mathbf{x}}_{I+K}, t_{n-1}), c(\underline{\mathbf{x}}_1, t_n), ..., c(\underline{\mathbf{x}}_{I+K}, t_n)\big), \qquad i = 1, ..., I.$$ ⌟

Definition 3.15 (Consistency): Consider Method 3.7. Let the spatial mesh sizes and the time steps be decreasing functions of a parameter h. The method is *consistent* at time t_n with respect to some norm $\|.\|$, if

$$\lim_{h \downarrow 0} \|\tilde{\underline{\mathbf{e}}}^n\| \quad = \quad 0, \qquad t_n, \underline{\mathbf{x}}_1, ..., \underline{\mathbf{x}}_I \text{ fixed.}$$

Here, $\tilde{\underline{\mathbf{e}}}^n$ is the vector containing the local truncation errors. ⌟

3.4.2 Robustness

Next to accuracy, efficiency is of great importance. A method is said to be *robust* if its "efficiency is insensitive to changes in the problem, such as variations in grid point distribution (especially cell aspect ratios)" [Wes01, p. 263], or in the velocity profile. The robustness of a scheme is usually related to conditions that ensure stability, positivity, and non-oscillatory behavior. This section discusses these properties.

Roughly speaking, stability means that a small perturbation of the initial condition should not lead to a completely different solution. A definition of absolute stability is given below. For more information about stability-related topics, see e.g. [Wes01, Chapter 5].

Definition 3.16 (Absolute stability): Method 3.7 is called *absolutely stable* with respect to some norm $\|.\|$, if there exist constants $k, \tau > 0$ (τ may depend on the spatial mesh size) such that, if

$$t_n - t_{n-1} \leq \tau, \qquad \forall n = 1, ..., N,$$

then, for any perturbation $\underline{\mathbf{w}}^0$ of the initial condition $\underline{\mathbf{c}}^0$ that yields a perturbation $\underline{\mathbf{w}}^n$ of $\underline{\mathbf{c}}^n$,

$$\|\underline{\mathbf{w}}^n\| \leq k\|\underline{\mathbf{w}}^0\|, \qquad \forall n = 1, ..., N.$$ ⌟

Next to stability, positivity is a favorable feature, because negative concentrations are unphysical. Positivity preserving schemes guarantee positive results, provided that the initial and boundary conditions are positive.

Definition 3.17 (Positivity preservation): Method 3.7 (for $p = 0$) is *positivity preserving*, if

$$\begin{aligned} \bar{c}_i^0 &\geq 0, \quad i = 1, ..., I, \\ \bar{c}_i^n &\geq 0, \quad i = I+1, ..., I+K;\ n = 0, ..., N, \end{aligned}$$

implies that

$$\bar{c}_i^n \geq 0, \qquad i = 1, .., I;\ n = 1, ..., N.$$ ⌟

Finally, the method should not generate spurious oscillations. In the one-dimensional case, the occurrence of wiggles is unlikely if the scheme is monotonicity-preserving [Wes01, p. 340].

Definition 3.18 (**Monotonicity preservation**)**:** Consider Method 3.7 for $p = 0$ and $m = 1$. The scheme is monotonicity preserving, if, for every non-decreasing (non-increasing) initial condition $\{\overline{c}_i^0\}$, the numerical solution at all later instants $\overline{c}_i^n$ $(n = 1, 2, ..., N)$ is non-decreasing (non-increasing).

In the multi-dimensional unstructured case, the concept of local extremum diminishing schemes is useful, which was introduced by Jameson [Jam93].

Definition 3.19 (**Local Extremum Diminishing (LED)**)**:** Consider Method 3.7 for $p = 0$. The system of ODEs that results from the spatial discretisation is called *local extremum diminishing*, if a local maximum cannot increase and a local minimum cannot decrease. ⌟

Observe that a LED scheme is automatically positivity preserving and L_∞-stable, as the global maximum cannot increase and the global minimum cannot decrease [KT02, p. 531]. However, the nice features of a LED spatial discretisation may still get disturbed by time discretisation. For this purpose, a local and a global discrete maximum principle will be discussed now. The local discrete maximum principle basically states that each concentration $\overline{c}_i^n$ lies between the minimum and maximum of the concentrations that it depends on. The global discrete maximum principle implies that each concentration $\overline{c}_i^n$ lies between the minimum and the maximum of the initial and the boundary conditions.

Definition 3.20 (**Local discrete maximum principle**)**:** Consider Method 3.7 for $p = 0$. Suppose a scheme of the form:

$$a_{ii}^n \overline{c}_i^n + \sum_{j \in \mathcal{A}_i^n} a_{ij}^n \overline{c}_j^n = \sum_{j \in \mathcal{B}_i^n} b_{ij}^n \overline{c}_j^{n-1}, \qquad i = 1, .., I,\ a_{ij}^n \neq 0\,(j \in \mathcal{A}_i^n \cup \{i\}),\ b_{ij}^n \neq 0\,(j \in \mathcal{B}_i^n),$$

where $\mathcal{A}_i \subset \mathcal{J}_i$ and $\mathcal{B}_i \subset \mathcal{J}_i \cup \{i\}$. The scheme admits a *local discrete maximum principle*, if, for any solution to the scheme, either

$$\overline{c}_i^n = \overline{c}_j^n = \overline{c}_k^{n-1}, \qquad j \in \mathcal{A}_i^n, k \in \mathcal{B}_i^n, \tag{3.7}$$

or

$$\min\left\{\min_{j \in \mathcal{A}_i^n} \overline{c}_j^n, \min_{j \in \mathcal{B}_i^n} \overline{c}_j^{n-1}\right\} < \overline{c}_i^n < \max\left\{\max_{j \in \mathcal{A}_i^n} \overline{c}_j^n, \max_{j \in \mathcal{B}_i^n} \overline{c}_j^{n-1}\right\}, \tag{3.8}$$

for all $i = 1, ..., I$ and for all $n = 1, .., N$. ⌟

Definition 3.21 (**Global discrete maximum principle**)**:** Consider Method 3.7 for $p = 0$. Suppose a scheme of the form:

$$a_{ii}^n \overline{c}_i^n + \sum_{j \in \mathcal{A}_i^n} a_{ij}^n \overline{c}_j^n = \sum_{j \in \mathcal{B}_i^n} b_{ij}^n \overline{c}_j^{n-1}, \qquad i = 1, .., I,\ a_{ij}^n \neq 0\,(j \in \mathcal{A}_i^n \cup \{i\}),\ b_{ij}^n \neq 0\,(j \in \mathcal{B}_i^n),$$

where $\mathcal{A}_i \subset \mathcal{J}_i$ and $\mathcal{B}_i \subset \mathcal{J}_i \cup \{i\}$. The scheme satisfies the *global discrete maximum principle*, if, for any solution to the scheme, for each $i = 1, ..., I$ and for each $n = 1, ..., N$, there is a non-decreasing path to the boundary or to to the initial condition in the sense that there exist $j_1, j_2, ..., j_E \in \{1, ..., I + K\}$ and $m_1, m_2, ..., m_E \in \{0, 1, ..., N\}$ such that

- $j_{e+1} \in \mathcal{A}_{j_e}^{m_e} \cup \mathcal{B}_{j_e}^{m_e}$,
- $m_{e+1} \leq m_e$,
- either $m_E = 0$ or $j_E \in \{I + 1, ..., I + K\}$,
- $\overline{c}_i^n \leq \overline{c}_{j_1}^{m_1} \leq ... \leq \overline{c}_{j_E}^{m_E}$,

and if, similarly, a non-increasing path exists. As a consequence, the scheme satisfies:

$$\underbrace{\min\left\{\min_{j=1,\ldots,I}\{\overline{c}_j^0\},\min_{\substack{j=I+1,\ldots,I+K\\n=0,\ldots,N}}\{\overline{c}_j^n\},\right\}}_{=:m}\leq\overline{c}_i^n\leq\underbrace{\max\left\{\max_{j=1,\ldots,I}\{\overline{c}_j^0\},\max_{\substack{j=I+1,\ldots,I+K\\n=0,\ldots,N}}\{\overline{c}_i^n\}\right\}}_{=:M},\tag{3.9}$$

for all $i=1,...,I+K$ and for all $n=0,...,N$. ⌟

Often, the global discrete maximum can be derived by successive application of the local discrete maximum principle. For the fully explicit case it is mentioned in [BO04, p. 13] that the local discrete maximum principle "precludes spurious extrema and $O(1)$ Gibbs-like phenomena". In the general implicit case, the global discrete maximum principle implies that, for each concentration $\overline{c}_i^n$, it is possible to step through the stencil, passing only concentrations that are not larger than $\overline{c}_i^n$, until either a boundary cell ($i=I+1,...,I+K$) or an initial cell ($n=0$) is reached. A similar result holds for non-increasing values. In other words: a local maximum in the interior can only be the result of the transportation of a local maximum in the initial or boundary conditions and, in that sense, can not be spurious. In the remaining of this thesis this will be called *non-oscillatory* behavior. Another convenient consequence of the global discrete maximum principle is that the scheme is positivity preserving and absolutely stable with respect to $\|.\|_\infty$.

Theorem 3.22: *If Method 3.7 satisfies the global discrete maximum principle (3.9), then the scheme is positivity preserving.*

Proof. Trivial. ■

Theorem 3.23: *If Method 3.7 satisfies the global discrete maximum principle (3.9), then the scheme is absolutely stable with respect to* $\|.\|_\infty$.

Proof. (See also [Wes01, p. 170-171].) Let $\underline{\mathbf{w}}^0$ be a perturbation of the initial condition $\overline{\underline{\mathbf{c}}}^0$ that yields a perturbation $\underline{\mathbf{w}}^n$ of $\overline{\underline{\mathbf{c}}}^n$. As both $\overline{\underline{\mathbf{c}}}^n$ and $\overline{\underline{\mathbf{c}}}^n+\underline{\mathbf{w}}^n$ satisfy the (linear) scheme, subtraction yields that $\underline{\mathbf{w}}^n$ also satisfies the scheme. Applying the global maximum principle (3.9), it follows that

$$\min_j\{w_j^0\}\leq w_i^n\leq\min_j\{w_j^0\}$$

Hence,

$$\|\underline{\mathbf{w}}^n\|_\infty\leq\|\underline{\mathbf{w}}^0\|_\infty,$$

As a result, the scheme is absolutely stable with respect to $\|.\|_\infty$ (see also Definition 3.16 for $k=1$). ■

3.5 Summary

The solution of the water quality model can be approximated by means of the finite volume method (FVM). The grid that is used by Delft3D-WAQ is usually three-dimensional, unstructured, and strongly non-uniform. Water quality processes are treated in an explicit manner, in order to avoid the necessity of solving a nonlinear system. The quality of a finite volume scheme is determined by accuracy and robustness. In this respect, the local and the global discrete maximum principle are favorable properties of a FVM because they imply stability, positivity and non-oscillatory behavior.

Chapter 4

Accurate explicit schemes

In the introduction it was mentioned that WAQ's current schemes are either explicit higher order schemes that are not robust, or implicit first order schemes that are inaccurate. This chapter considers the first category.

4.1 Local extremum diminishing flux functions

The flux function that is used in a finite volume scheme determines much of the characteristics of the scheme. The beauty of first order upwind discretisation (4.1) is that it leads to a linear local extremum diminishing (LED) system of ODEs. This is easy to prove with the help of the following theorem.

Theorem 4.1: *Consider Method 3.7 for $p = 0$. Suppose that spatial discretisation has resulted in a system of the form:*

$$\begin{aligned} \frac{d|\mathcal{V}_i|\overline{c}_i}{dt} &= -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}|\phi_{ij} \\ -|\mathcal{S}_{ij}|\phi_{ij} &= -\gamma_{ji}\overline{c}_i + \gamma_{ij}\overline{c}_j, \qquad \sum_{j\in\mathcal{J}_i}(-\gamma_{ji}+\gamma_{ij}) = 0. \end{aligned}$$

If $\gamma_{ij} \geq 0$ for all i and j, then, the system is LED.

Proof. (See also [Jam93, p. 385] or [KT02, p. 531].) Suppose that $\overline{c}_i$ is a local maximum. So $\overline{c}_i \geq \overline{c}_j$ for all $j \in \mathcal{A}_i$. Consequently:

$$\begin{aligned} \frac{d|\mathcal{V}_i|\overline{c}_i}{dt} &= \sum_{j\in\mathcal{J}_i}(-\gamma_{ji}\overline{c}_i + \gamma_{ij}\overline{c}_j) \\ &= \sum_{j\in\mathcal{J}_i} -\gamma_{ji}\overline{c}_i + \sum_{j\in\mathcal{J}_i}\gamma_{ij}(\overline{c}_j - \overline{c}_i) + \sum_{j\in\mathcal{J}_i}\gamma_{ij}\overline{c}_i \\ &= \underbrace{\sum_{j\in\mathcal{J}_i}(-\gamma_{ji}+\gamma_{ij})\,\overline{c}_i}_{=0} + \sum_{j\in\mathcal{J}_i}\underbrace{\gamma_{ij}}_{\geq 0}\underbrace{(\overline{c}_j - \overline{c}_i)}_{\leq 0} \leq 0. \end{aligned}$$

Therefore, the local maximum can not increase. Similarly, it can be shown that a local minimum can not decrease. Hence, the scheme is LED. ■

Proposition 4.2: *Consider Method 3.7 for $p = 0$, after applying first order upwind spatial discretisation to the advection terms and central difference discretisation to the diffusion terms:*

$$\frac{d|\mathcal{V}_i|\overline{c}_i}{dt} = -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\phi_{ij},$$

where ϕ_{ij} is the following flux function:

$$\begin{aligned} -|\mathcal{S}_{ij}|\phi_{ij} &= -\gamma_{ji}\overline{c}_i + \gamma_{ij}\overline{c}_j \\ \gamma_{ij} &= -|\mathcal{S}_{ij}|\left(\min\{0, \overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\} - \frac{\overline{d}_{ij}}{\|\underline{\mathbf{x}}_j - \underline{\mathbf{x}}_i\|_2}\right). \end{aligned} \tag{4.1}$$

If the velocity profile is conservative in the sense that

$$\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}|\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} = 0, \tag{4.2}$$

then, the resulting system is LED.

Proof. First, note that $\gamma_{ij} \geq 0$. Furthermore, observe that

$$\begin{aligned} \sum_{j\in\mathcal{J}_i}(-\gamma_{ji}+\gamma_{ij}) &= -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\left(-\max\{0,\overline{\underline{\mathbf{u}}}_{ji}\cdot\underline{\mathbf{n}}_{ji}\} + \max\{0,\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\}\right) \\ &= -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\left(\min\{0,\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\} + \max\{0,\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\}\right) \\ &= -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} \\ &\overset{(4.2)}{=} 0 \end{aligned}$$

Applying Theorem 4.1 yields that the scheme is LED. ■

A similar argument does not hold for central discretisation (4.3), which is of second order. However, the proposition below shows how central discretisation can be rendered LED by eliminating negative coefficients by adding artificial diffusion. This strategy results in what will be called upwinded central discretisation.

Proposition 4.3: *Consider Method 3.7 for $p = 0$. Apply central discretisation to the advection terms and the central difference scheme to the diffusion terms to obtain:*

$$\frac{d|\mathcal{V}_i|\overline{c}_i}{dt} = -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\tilde{\phi}_{ij},$$

where $\tilde{\phi}_{ij}$ is the following flux function:

$$\begin{aligned} -|\mathcal{S}_{ij}|\tilde{\phi}_{ij} &= -\tilde{\gamma}_{ji}\overline{c}_i + \tilde{\gamma}_{ij}\overline{c}_j \\ \tilde{\gamma}_{ij} &= -|\mathcal{S}_{ij}|\left(\frac{\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}}{2} - \frac{\overline{d}_{ij}}{\|\underline{\mathbf{x}}_j - \underline{\mathbf{x}}_i\|_2}\right). \end{aligned} \tag{4.3}$$

If ϕ_{ij} is the following 'upwinded' version of $\tilde{\phi}_{ij}$:

$$\begin{aligned} -|\mathcal{S}_{ij}|\phi_{ij} &= -\gamma_{ji}\overline{c}_i + \gamma_{ij}\overline{c}_j \\ \gamma_{ij} &= \tilde{\gamma}_{ij} + \max\{0, -\tilde{\gamma}_{ij}, -\tilde{\gamma}_{ji}\}, \end{aligned} \tag{4.4}$$

and if the velocity profile is conservative in the sense that

$$\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}|\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij} = 0, \tag{4.5}$$

then, the system

$$\frac{d|\mathcal{V}_i|\overline{c}_i}{dt} = -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\phi_{ij},$$

is LED.

Proof. First, note that $\gamma_{ij} \geq 0$. Furthermore, observe that

$$\begin{aligned}\sum_{j\in\mathcal{J}_i}(-\gamma_{ji}+\gamma_{ij}) &= \sum_{j\in\mathcal{J}_i}(-\tilde{\gamma}_{ji}+\tilde{\gamma}_{ij})\\ &= -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}|\overline{\underline{\mathbf{u}}}_{ij}\cdot\underline{\mathbf{n}}_{ij}\\ &\overset{(4.5)}{=} 0\end{aligned}$$

Applying Theorem 4.1 yields that the scheme is LED. ■

According to Kuzmin et al. [KMT, p. 4-5], who proposed this strategy, the physical diffusion is "automatically detected and the amount of artificial diffusion is reduced accordingly." For diffusion dominated problems, upwinded central discretisation is identical to central discretisation, "since the coefficients are nonnegative from the outset". For the one-dimensional advection equation, upwinded central discretisation is equivalent to first order upwind discretisation.

4.2 Explicit schemes

Method 4.4 (Explicit (upwind) scheme): The *explicit scheme* is a FVM (Method 3.7) that reads (for p=0):

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1},$$

where ϕ_{ij} is a flux function. If ϕ_{ij} is as in (4.1), the scheme is known as the *explicit upwind scheme*. ⌟

Figure 4.1 displays the results of the explicit upwind scheme for the following test case.

Test case 4.5 (One-dimensional advection equation): This test case is the one-dimensional advection equation with periodic boundary conditions:

$$\begin{cases} \frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0, & x\in[0,10], t\geq 0,\\ c(x,0) = 1_{[2,4]}(x)\frac{1}{2}\big(1-\cos(\pi x)\big) + 1_{[6,8]}(x), & x\in[0,10],\\ c(0,t) = c(10,t), & t\geq 0. \end{cases}$$

Note that the (exact) solution is periodic with a period of 10. ⌟

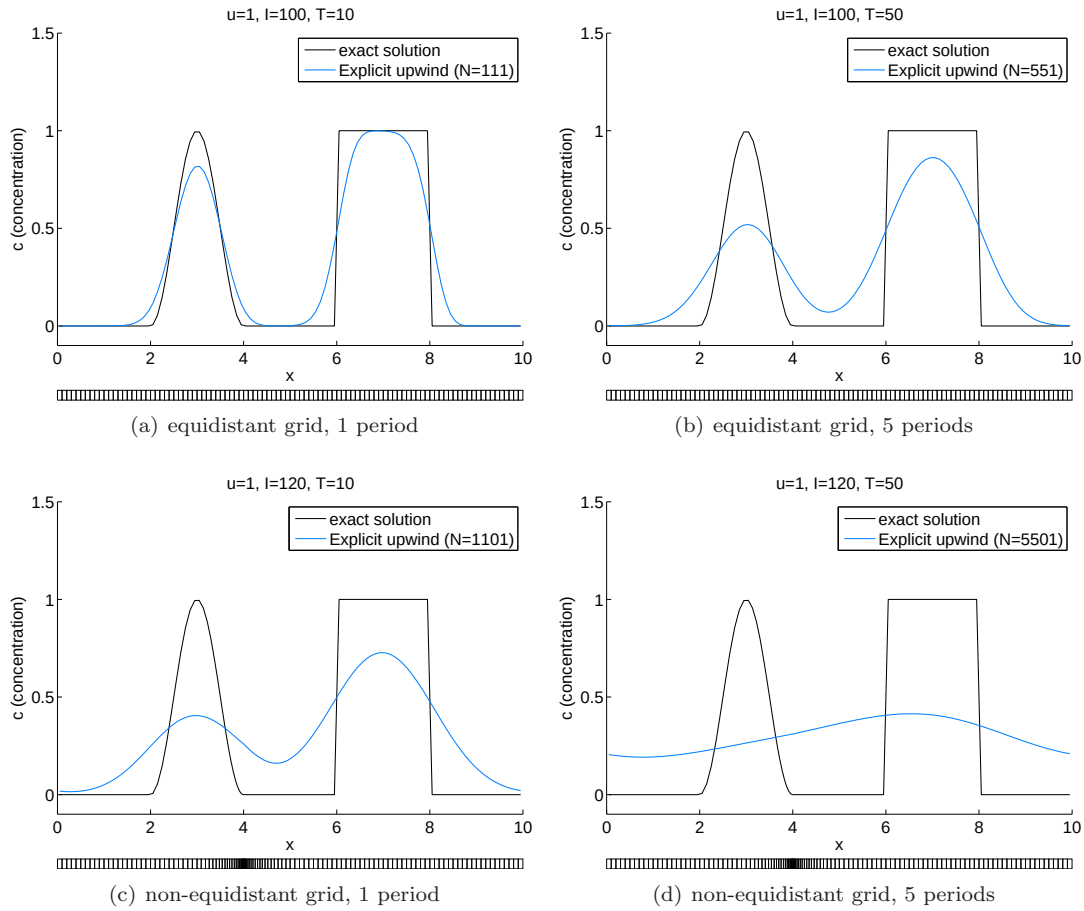
For this test case, the explicit upwind scheme reads:

$$\frac{\Delta x_i\overline{c}_i^n - \Delta x_i\overline{c}_i^{n-1}}{\Delta t} = -u\overline{c}_i^{n-1} + u\overline{c}_{i-1}^{n-1}, \tag{4.6}$$

where $\Delta t = t_n - t_{n-1}$ is the constant time step and $\Delta x_i = |\mathcal{V}_i^{n-1}| = |\mathcal{V}_i^n|$ is the cell width. The scheme will be stable, positivity preserving and non-oscillatory if

$$\frac{\Delta x_i}{u\Delta t} \leq 1, \qquad i = 1,...,I, \tag{4.7}$$

which results from (5.2) for $\theta = 0$. This condition is also known as the *Courant-Friedrichs-Lewy (CFL) condition* [LeV02, Section 4.4]. Note that the smallest grid cell is responsible for the number of time steps. This explains why the number of time steps N is much larger in Figures 4.1(c) and 4.1(d) than in Figures 4.1(a) and 4.1(b). Apparently, the scheme is not robust. Furthermore, the explicit upwind scheme is not very accurate for this test case. The numerical solution is smeared out compared to the exact solution. Observe that this effect is larger in Figures 4.1(c) and 4.1(d)



(a) equidistant grid, 1 period

(b) equidistant grid, 5 periods

(c) non-equidistant grid, 1 period

(d) non-equidistant grid, 5 periods

Figure 4.1: Explicit upwind scheme (Method 4.4) for Test case 4.5

than in Figures 4.1(a) and 4.1(b). This is explained by the fact that, for a one-dimensional equidistant grid, the explicit upwind scheme introduces the following additional artificial diffusion coefficient:

$$\frac{u\Delta x}{2}\left(1-\frac{u\Delta t}{\Delta x}\right). \tag{4.8}$$

This follows from Proposition 5.2 for $\theta = 0$. A simple calculation shows that the numerical diffusion in Figures 4.1(a) and 4.1(b) reads 0.0198, whereas it is equal to 0.1818 (in the largest cells) in Figures 4.1(c) and 4.1(d). The next section discusses a strategy to diminish this numerical diffusion.

4.3 Flux corrected transport

Theorem 4.6 (Godunov's barrier theorem)**:** *Consider the one-dimensional advection equation (Model 2.1 for $m = 1$, $d = 0$, $p = 0$, and u constant):*

$$\frac{\partial c}{\partial t} + u\frac{\partial c}{\partial x} = 0.$$

Suppose that the model is solved by means of Method 3.7 with a constant mesh width Δx and a constant time step Δt. Assume that the scheme can be written in the form:

$$\overline{c}_i^n \quad = \quad \sum_j \gamma_j \overline{c}_{i+j}^{n-1}.$$

If the scheme is second-order in the sense that the exact solution is reproduced if the initial condition is a polynomial of second degree, then, the scheme cannot be monotonicity preserving, unless $\frac{|u|\Delta t}{\Delta x} \in \mathbb{N}$.

Proof. See [Wes01, Theorem 9.2.2]. ■

Roughly speaking, Godunov's barrier theorem implies that a linear method either is relatively inaccurate (remember the artificial diffusion of the explicit upwind scheme), or it has a tendency to generate spurious oscillations [BO04, p.19]. The Flux Correcting Transport (FCT) algorithm attempts to combine a non-oscillatory first order flux function with an accurate higher order flux function by means of a nonlinear limiter. Roughly speaking, the algorithm uses a convex combination of the two flux functions, instead of just one of them. This strategy can also be described as updating the first order flux with a limited correction flux.

Method 4.7 (Explicit FCT scheme): The *explicit FCT scheme* is a FVM (Method 3.7) of the following form (for $p = 0$):

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} \quad = \quad -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^{n-1}|\Big(\hat{\phi}_{ij}^{n-1} + l_{ij}^n\big(\tilde{\phi}_{ij}^{n-1} - \hat{\phi}_{ij}^{n-1}\big)\Big). \tag{4.9}$$

Here, $\hat{\phi}_{ij}$ is a non-oscillatory first order numerical flux function of the form (3.5), and $\tilde{\phi}_{ij}$ is a higher order one. The difference $\tilde{\phi}_{ij} - \hat{\phi}_{ij}$ can be interpreted as a correction term, which is limited by l_{ij}. If $\hat{\phi}_{ij}$ is as in (4.1), the *explicit upwind FCT scheme* is obtained. ⌟

Note that choosing $l_{ij}^n = 0$ corresponds to applying the first order method. On the other hand, setting $l_{ij}^n = 1$ is equivalent to using the higher order method. The limiter that is used by scheme 12 of WAQ is a generalised version of the one-dimensional limiter that was proposed by Boris & Book [BB73], the founders of the FCT algorithm. This limiter has been extended to the multi-dimensional case on a structured grid by Zalesak [Zal79]. Below, it is further generalised for an unstructured grid. The limiter allows as much correction as possible, provided that it generates no new local extrema.

Method 4.8 (Explicit FCT scheme à la Boris & Book): This method results from the explicit FCT scheme (Method 4.7) by using the following limiter:

1. Suppose that $\overline{c}_i^{n-1}$ is known from the previous time step. Compute a first order approximation $\hat{\overline{c}}_i^n$ by means of (4.9) with $l_{ij}^n = 0$, which is equivalent to applying the explicit scheme (Method 4.4) with the lower order flux function $\hat{\phi}_{ij}$:

$$\frac{|\mathcal{V}_i^n|\hat{\overline{c}}_i^n - |\mathcal{V}_i^{n-1}|\hat{\overline{c}}_i^{n-1}}{t_n - t_{n-1}} \quad = \quad -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^{n-1}|\hat{\phi}_{ij}^{n-1}.$$

2. Define the flux correction $\Delta\phi_{ij}^n$ as follows:

$$\Delta\phi_{ij}^n \quad = \quad \tilde{\phi}_{ij}^{n-1} - \hat{\phi}_{ij}^{n-1}$$

3. Since $\Delta\phi_{ij}^n$ often corrects the numerical diffusion of the first order flux, it is sometimes referred to as anti-diffusion. Because an anti-diffusion term should not behave as a diffusion term, put $\Delta\phi_{ij}^n = 0$ if

$$\Delta\phi_{ij}^n(\hat{\overline{c}}_i^n - \hat{\overline{c}}_j^n) > 0.$$

If this prelimiting step is not performed, the flux correction may smooth the low-order solution, or it may cause small-scale numerical ripples [KT02, p.541]. Since it is unphysical for an anti-diffusion term to be directed from a higher concentration to a lower concentration, the effect of the adjustment above is minimal in practice [Zal79, p. 342].

4. Construct an upper and a lower bound for $\overline{c}_i^n$ by means of the first order approximation $\hat{\overline{c}}_i^n$:

$$\begin{aligned} \overline{c}_i^{\max} &= \max_{j\in\mathcal{J}_i\cup\{i\}}\{\hat{\overline{c}}_j^n\}, \\ \overline{c}_i^{\min} &= \min_{j\in\mathcal{J}_i\cup\{i\}}\{\hat{\overline{c}}_j^n\}. \end{aligned}$$

5. The amount of mass that flows into cell $\mathcal{V}_i$ as a result of the flux correction (without the limiter) reads:

$$\lambda_i^+ = \sum_{j\in\mathcal{J}_i}(t_n - t_{n-1})|\mathcal{S}_{ij}^n|\max\{0, -\Delta\phi_{ij}^n\}.$$

The allowed mass increase is, however:

$$\mu_i^+ = |\mathcal{V}_i|\left(\overline{c}_i^{\max} - \hat{\overline{c}}_i^n\right).$$

Thus, the fraction of mass that is allowed to flow into the cell is given by:

$$\nu_i^+ = \begin{cases} \min\left\{1, \frac{\mu_i^+}{\lambda_i^+}\right\}, & \lambda_i^+ > 0, \\ 0, & \lambda_i^+ = 0, \end{cases}$$

Introduce analogue quantities for mass decrease:

$$\begin{aligned} \lambda_i^- &= \sum_{j\in\mathcal{J}_i}(t_n - t_{n-1})|\mathcal{S}_{ij}^n|\max\{0, \Delta\phi_{ij}^n\}, \\ \mu_i^- &= |\mathcal{V}_i|\left(\hat{\overline{c}}_i^n - \overline{c}_i^{\min}\right), \\ \nu_i^- &= \begin{cases} \min\left\{1, \frac{\mu_i^-}{\lambda_i^-}\right\}, & \lambda_i^- > 0, \\ 0, & \lambda_i^- = 0. \end{cases} \end{aligned}$$

6. The limiter is the mass fraction that is allowed by both adjacent cells:

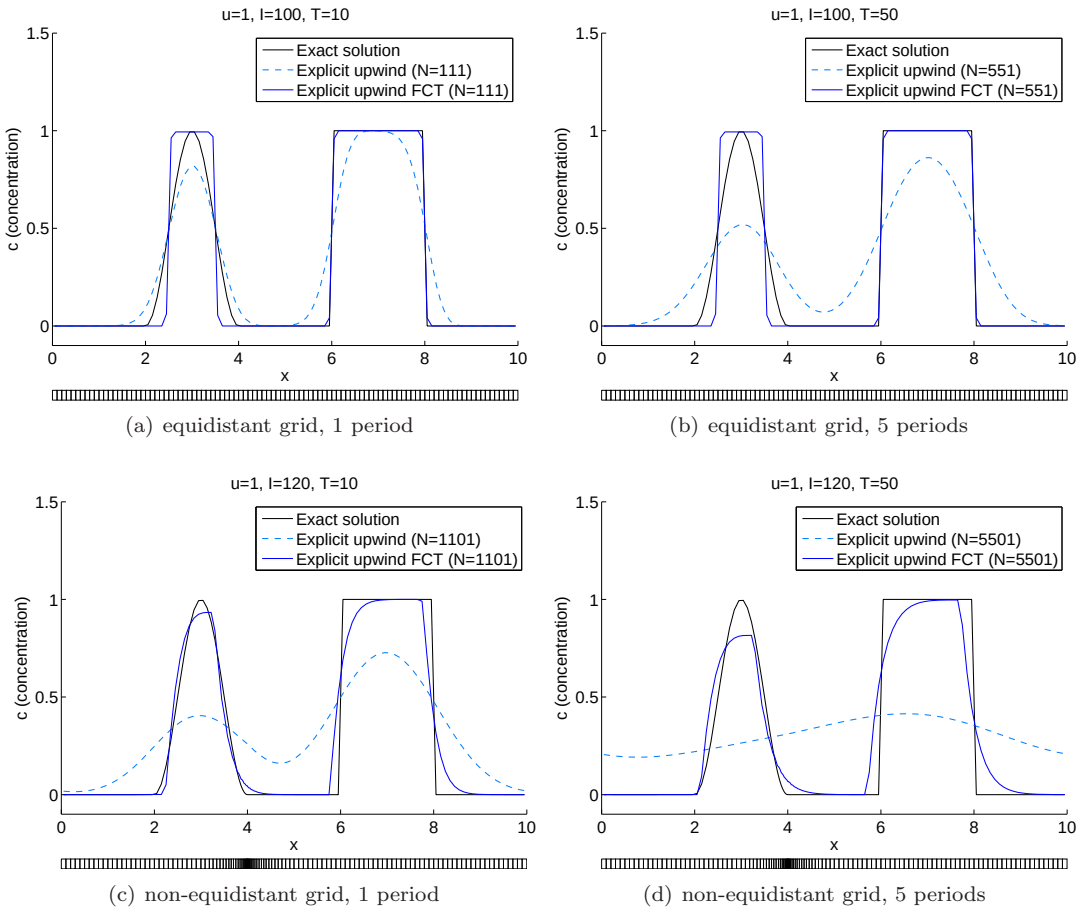
$$l_{ij}^n = \begin{cases} \min\{\nu_j^+, \nu_i^-\}, & \Delta\phi_{ij}^n \geq 0, \\ \min\{\nu_i^+, \nu_j^-\}, & \Delta\phi_{ij}^n < 0. \end{cases}$$

⌟

Remark 4.9: Alternatively, next to the first order approximation $\hat{\overline{c}}_i^n$, the previous solution estimation $\overline{c}_i^{n-1}$ could be used to determine the bounds $\overline{c}_i^{\max}$ and $\overline{c}_i^{\min}$:

$$\begin{aligned} \overline{c}_i^{\max} &= \max_{j\in\mathcal{J}_i\cup\{i\}}\left\{\max\{\overline{c}_j^{n-1}, \hat{\overline{c}}_j^n\}\right\}, \\ \overline{c}_i^{\min} &= \min_{j\in\mathcal{J}_i\cup\{i\}}\left\{\min\{\overline{c}_j^{n-1}, \hat{\overline{c}}_j^n\}\right\}. \end{aligned}$$

In general, this leads to a larger limiter, which means that more flux correction is applied that diminishes numerical diffusion. However, this choice is less safe because the birth of new local extremes is no longer excluded (see also [Zal79, p. 342]). ⌟



(a) equidistant grid, 1 period

(b) equidistant grid, 5 periods

(c) non-equidistant grid, 1 period

(d) non-equidistant grid, 5 periods

Figure 4.2: Explicit upwind FCT scheme (Method 4.8) with central flux correction (4.3) for Test case 4.5

Remark 4.10 (TVD limiters)**:** After Boris and Book proposed the FCT algorithm, many alternative limiting strategies have been developed. A popular class is formed by the Total Variation Diminishing (TVD) limiters, which includes minmod, Van Leer, MC, and superbee. These limiters are considered for multi-dimensional unstructured grids in [KT, Sections 2 and 10]. ⌟

Figure 4.2 displays the results of the explicit upwind FCT scheme (Method 4.8) with central flux correction (4.3). Compared to the explicit upwind scheme, the numerical diffusion has been reduced drastically. However, the anti-diffusion is too large in some portions of the domain, especially in Figures 4.2(a) and 4.2(b). This is explained by the fact that, for a one-dimensional equidistant grid, the explicit scheme with central fluxes introduces the following additional artificial diffusion coefficient:

$$-\frac{1}{2}u^2\Delta t.$$

This follows from Proposition 5.6 for $\theta = 0$. Because negative diffusion is unphysical, central fluxes are unsuitable to serve as flux correction in the explicit FCT scheme. An alternative flux correction is provided by the Lax-Wendroff scheme.

Method 4.11 (**Lax-Wendroff scheme**)**:** The *Lax-Wendroff* scheme is a fully discrete FVM (see

Method 3.7 for notational aspects) of the following form (for $p = 0$ and $\underline{\underline{\tilde{D}}} = 0$):

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1},$$

where

$$\phi_{ij}^{n-1} = \hat{\phi}_{ij}^{n-1} + \left(\frac{\Delta\xi_{ij}^{n-1}}{\|\underline{\mathbf{x}}_j^{n-1} - \underline{\mathbf{x}}_i^{n-1}\|_2} - \frac{\underline{\mathbf{u}}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1}(t_n - t_{n-1})}{2\|\underline{\mathbf{x}}_j^{n-1} - \underline{\mathbf{x}}_i^{n-1}\|_2}\right)\underline{\mathbf{u}}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1}\left(\overline{c}_j^{n-1} - \overline{c}_i^{n-1}\right).$$

Here, $\hat{\phi}_{ij}^{n-1}$ is the first order upwind flux:

$$\hat{\phi}_{ij}^{n-1} = \max\{\overline{\mathbf{u}}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1}, 0\}\overline{c}_i^{n-1} + \min\{\overline{\mathbf{u}}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1}, 0\}\overline{c}_j^{n-1},$$

and $\Delta\xi_{ij}^{n-1}$ is the distance[1] between $\underline{\mathbf{x}}_i^{n-1}$ and $\mathcal{S}_{ij}^{n-1}$, if $\overline{u}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1} \geq 0$, and minus the distance between $\underline{\mathbf{x}}_j^{n-1}$ and $\mathcal{S}_{ij}^{n-1}$, if $\overline{u}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1} < 0$.

Derivation. First, reduce the model to the one-dimensional advection equation with constant velocity $u > 0$, constant cell width $\Delta x > 0$, and constant time step $\Delta t > 0$. Consider the following Taylor expansion around $t = t_{n-1}$:

$$c(x, t_n) \approx c(x, t_{n-1}) + \Delta t\frac{\partial c}{\partial t}(x, t_{n-1}) + \frac{\Delta t^2}{2}\frac{\partial^2 c}{\partial t^2}(x, t_{n-1}).$$

This expression can be rewritten to obtain:

$$\frac{c(x, t_n) - c(x, t_{n-1})}{\Delta t} \approx \frac{\partial c}{\partial t}(x, t_{n-1}) + \frac{\Delta t}{2}\frac{\partial^2 c}{\partial t^2}(x, t_{n-1}).$$

Note that:

$$\begin{aligned}\frac{\partial c}{\partial t} &= -u\frac{\partial c}{\partial x},\\ \frac{\partial^2 c}{\partial t^2} &= u^2\frac{\partial^2 c}{\partial x^2}.\end{aligned}$$

Hence:

$$\frac{c(x, t_n) - c(x, t_{n-1})}{\Delta t} = -u\frac{\partial c}{\partial x}(x, t_{n-1}) + \frac{\Delta t}{2}u^2\frac{\partial^2 c}{\partial x^2}(x, t_{n-1}).$$

Applying central differences yields:

$$\frac{\overline{c}_i^n - \overline{c}_i^{n-1}}{\Delta t} = -u\frac{\overline{c}_{i+1}^{n-1} - \overline{c}_{i-1}^{n-1}}{2\Delta x} + \frac{\Delta t}{2}u^2\frac{\overline{c}_{i-1}^{n-1} - 2\overline{c}_i^{n-1} + \overline{c}_{i+1}^{n-1}}{\Delta x^2}.$$

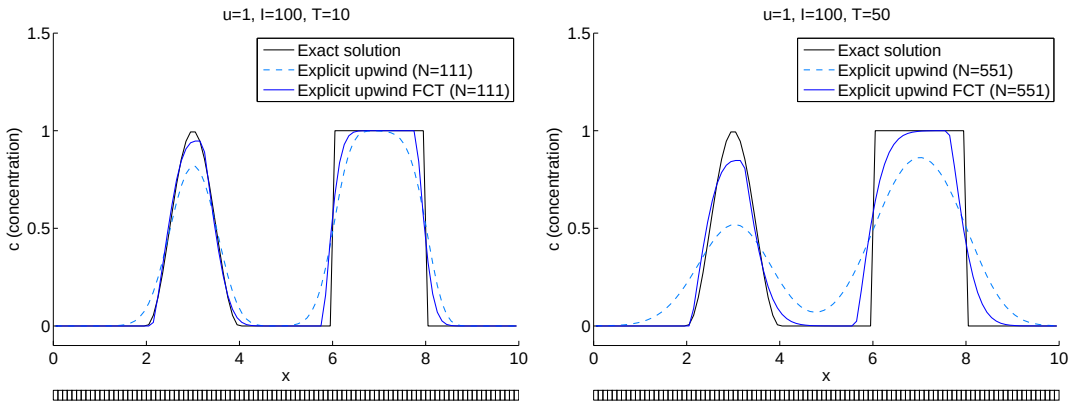
Rewriting gives:

$$\begin{aligned}\Delta x\frac{\overline{c}_i^n - \overline{c}_i^{n-1}}{\Delta t} &= -u\overline{c}_i^{n-1} + u\overline{c}_{i-1}^{n-1}\\ &\quad -\frac{1}{2}\left(1 - \frac{u\Delta t}{\Delta x}\right)u(\overline{c}_{i+1}^{n-1} - \overline{c}_i^{n-1}) - \frac{1}{2}\left(1 - \frac{u\Delta t}{\Delta x}\right)u(\overline{c}_{i-1}^{n-1} - \overline{c}_i^{n-1}).\end{aligned}$$

Finally, the derivation of the generalisation to non-equidistant grids can be found in [LeV02, Section 6.17]. ■

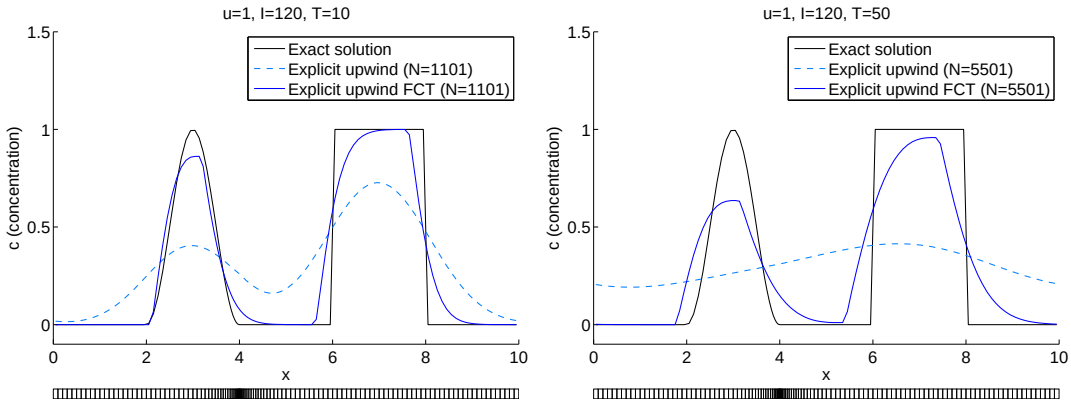
Figure 4.3 displays the results of the explicit upwind FCT scheme (Method 4.8) with Lax-Wendroff flux correction (Method 4.11) for Test case 4.5. Again, compared to the explicit upwind scheme, the numerical diffusion has been reduced. Where the central scheme caused too much anti-diffusion (see Figure 4.2), the Lax-Wendroff scheme seems to add a more appropriate amount. Proposition 5.7 (for $\theta = 0$) shows that, on a one-dimensional equidistant grid, the numerical diffusion of the Lax-Wendroff scheme is equal to zero. Scheme 12 of WAQ uses Lax-Wendroff flux correction.

[1] Of course, there is more than one way to define this distance. In WAQ, the distance between $\underline{\mathbf{x}}_i$ and $\mathcal{S}_{ij}$ is an approximation of $\|\underline{\mathbf{x}}_i - \underline{\mathbf{y}}_{ij}\|_2$. Here y_{ij}^n is the point where the line through $\underline{\mathbf{x}}_i$ and $\underline{\mathbf{x}}_j$ intersects $\mathcal{S}_{ij}$.



(a) equidistant grid, 1 period

(b) equidistant grid, 5 periods



(c) non-equidistant grid, 1 period

(d) non-equidistant grid, 5 periods

Figure 4.3: Explicit upwind FCT scheme (Method 4.8) with Lax-Wendroff flux correction (Method 4.11) for Test case 4.5

4.4 Summary

The explicit upwind scheme is neither robust nor accurate. The former is caused by the fact that the time step is limited to ensure stability, positivity and non-oscillatory behavior. The inaccuracy is a a result of numerical diffusion, which can be diminished by means of the flux corrected transport (FCT) algorithm. Central flux correction leads to an anti-diffusion error. For this reason, the Lax-Wendroff scheme is more suitable to serve as flux corrector, as its numerical diffusion is equal to zero. Although the accuracy of the explicit FCT scheme is generally high, the robustness of the explicit scheme remains unchanged, which can result in long computational times. Scheme 12 of WAQ is an explicit FCT scheme with Lax-Wendroff flux correction.

Chapter 5

Robust implicit theta schemes

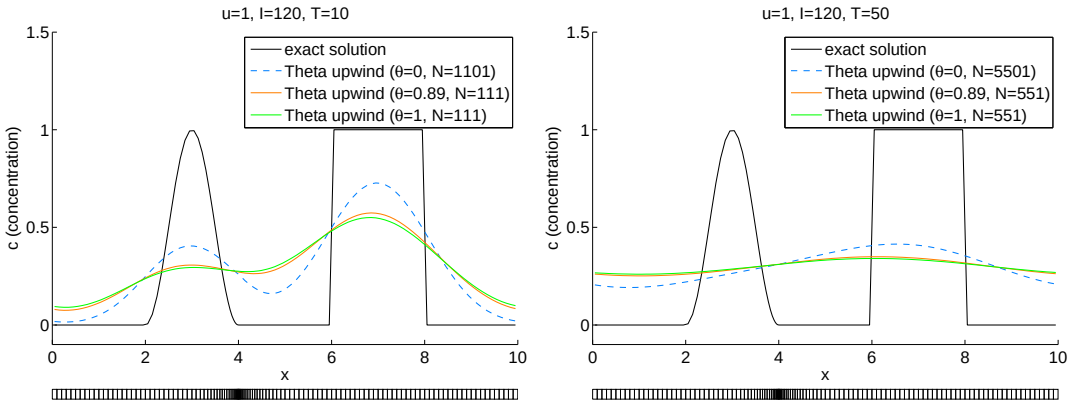
In the introduction it was mentioned that WAQ's current schemes are either explicit higher order schemes that are not robust, or implicit first order schemes that are inaccurate. This chapter considers the second category.

5.1 Theta scheme

Method 5.1 (Theta (upwind) scheme): The *theta scheme* is a FVM (Method 3.7) that reads (for p=0):

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} \quad = \quad -\sum_{j\in\mathcal{J}_i} \left((1-\theta)|\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1} + \theta|\mathcal{S}_{ij}^{n}|\phi_{ij}^{n}\right),$$

where ϕ_{ij} is a flux function. If ϕ_{ij} is as in (4.1), the scheme *theta upwind scheme* is obtained. ⌟



(a) 1 period (b) 5 periods

Figure 5.1: Theta upwind scheme (5.1) for Test case 4.5

Figure 5.1 displays the results of the theta upwind scheme for Test case 4.5. For this test case, the theta upwind scheme reads:

$$\frac{\Delta x_i \overline{c}_i^n - \Delta x_i \overline{c}_i^{n-1}}{\Delta t} \quad = \quad -(1-\theta)u(\overline{c}_i^{n-1} - \overline{c}_{i-1}^{n-1}) - \theta u(\overline{c}_i^{n} - \overline{c}_{i-1}^{n}), \tag{5.1}$$

where $\Delta t = t_n - t_{n-1}$ is the constant time step and $\Delta x_i = |\mathcal{V}_i^{n-1}| = |\mathcal{V}_i^n|$ is the cell width. The scheme will be stable, positivity preserving and non-oscillatory if

$$\theta \geq 1 - \frac{\Delta x_i}{u \Delta t}, \qquad i = 1, ..., I, \tag{5.2}$$

which follows from (6.10) for constant θ. The test was performed for three different pairs of θ and Δt that satisfy (5.2). In the fully explicit case ($\theta = 0$, blue), this has resulted in a relatively large number of time steps. In the fully implicit case ($\theta = 1$, red), any number of time steps is allowed, because (5.2) is unconditionally satisfied. This is the value that is used by scheme 16 of WAQ. Finally, observe that the numerical solution is more smeared out in comparison with the exact solution as θ becomes larger. In the following proposition, an expression for the numerical diffusion coefficient is derived with the help of the modified equation.

Proposition 5.2: *Consider the theta upwind scheme (Method 5.1) for the one-dimensional advection equation with constant velocity $u > 0$, constant cell width $\Delta x > 0$, and constant time step $\Delta t > 0$:*

$$\frac{\overline{c}_i^{n+1} - \overline{c}_i^n}{\Delta t} + \theta u \frac{\overline{c}_i^{n+1} - \overline{c}_{i-1}^{n+1}}{\Delta x} + (1-\theta) u \frac{\overline{c}_i^n - \overline{c}_{i-1}^n}{\Delta x} = 0.$$

Let $\eta(x,t)$ be a 'sufficiently differentiable' function such that $\eta(x_i, t_n) = \overline{c}_i^n$. Then, the corresponding modified equation *reads*

$$\frac{\partial \eta}{\partial t}(x_i,t_n) + u \frac{\partial \eta}{\partial x}(x_i,t_n) \approx \underbrace{\frac{u\Delta x}{2}\left(1 - (1-2\theta)\frac{u\Delta t}{\Delta x}\right)}_{\textit{Numerical diffusion coefficient}} \frac{\partial^2 \eta}{\partial x^2}(x_i,t_n).$$

Proof. Because $\eta(x_i, t_n) = \overline{c}_i^n$,

$$\begin{aligned} &\frac{\eta(x_i,t_{n+1}) - \eta(x_i,t_n)}{\Delta t} \\ &+\theta u \frac{\eta(x_i,t_{n+1}) - \eta(x_{i-1},t_{n+1})}{\Delta x} \\ &+(1-\theta) u \frac{\eta(x_i,t_n) - \eta(x_{i-1},t_n)}{\Delta x} = 0. \end{aligned}$$

Using a Taylor expansion of η around t_n results in (higher order terms that will be neglected later are colored):

$$\begin{aligned} &\frac{\partial \eta}{\partial t}(x_i,t_n) + \frac{\Delta t}{2}\frac{\partial^2 \eta}{\partial t^2}(x_i,t_n) + \frac{\Delta t^2}{6}\frac{\partial^3 \eta}{\partial t^3}(x_i,\tau_1) \\ &+\theta u \left(\frac{\eta(x_i,t_n) + \Delta t \frac{\partial \eta}{\partial t}(x_i,t_n) + \frac{\Delta t^2}{2}\frac{\partial^2 \eta}{\partial t^2}(x_i,t_n) + \frac{\Delta t^3}{6}\frac{\partial^3 \eta}{\partial t^3}(x_i,\tau_2)}{\Delta x} \right. \\ &\left. - \frac{\eta(x_{i-1},t_n) + \Delta t \frac{\partial \eta}{\partial t}(x_{i-1},t_n) + \frac{\Delta t^2}{2}\frac{\partial^2 \eta}{\partial t^2}(x_{i-1},t_n) + \frac{\Delta t^3}{6}\frac{\partial^3 \eta}{\partial t^3}(x_{i-1},\tau_3)}{\Delta x} \right) \\ &+(1-\theta) u \frac{\eta(x_i,t_n) - \eta(x_{i-1},t_n)}{\Delta x} = 0, \end{aligned}$$

for certain $\tau_1, \tau_2, \tau_3 \in [t_n, t_{n+1}]$. Using a Taylor expansion of η around x_i yields:

$$\begin{aligned}
&\frac{\partial \eta}{\partial t}(x_i,t_n) + \frac{\Delta t}{2}\frac{\partial^2 \eta}{\partial t^2}(x_i,t_n) + \frac{\Delta t^2}{6}\frac{\partial^3 \eta}{\partial t^3}(x_i,\tau_1) \\
+\theta u\Bigg(&\frac{\partial \eta}{\partial x}(x_i,t_n) - \frac{\Delta x}{2}\frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) + \frac{\Delta x^2}{6}\frac{\partial^3 \eta}{\partial x^3}(\xi_1,t_n) \\
&+\Delta t \frac{\partial}{\partial x}\frac{\partial \eta}{\partial t}(x_i,t_n) - \frac{\Delta t \Delta x}{2}\frac{\partial^2}{\partial x^2}\frac{\partial \eta}{\partial t}(\xi_2,t_n) \\
&+\frac{\Delta t^3}{6\Delta x}\frac{\partial^3 \eta}{\partial t^3}(x_i,\tau_2) - \frac{\Delta t^3}{6\Delta x}\frac{\partial^3 \eta}{\partial t^3}(\xi_3,\tau_3)\Bigg) \\
+(1-\theta)u\left(\frac{\partial \eta}{\partial x}(x_i,t_n) - \frac{\Delta x}{2}\frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) + \frac{\Delta x^2}{6}\frac{\partial^3 \eta}{\partial x^3}(\xi_4,t_n)\right) &= 0,
\end{aligned}$$

for certain $\xi_1, \xi_2, \xi_3, \xi_4 \in [x_{i-1}, x_i]$. Rewriting gives:

$$\begin{aligned}
\frac{\partial \eta}{\partial t}(x_i,t_n) + u\frac{\partial \eta}{\partial x}(x_i,t_n) &= \frac{u\Delta x}{2}\frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) \\
&\quad -\frac{\Delta t}{2}\frac{\partial^2 \eta}{\partial t^2}(x_i,t_n) - \theta u \Delta t \frac{\partial}{\partial x}\frac{\partial \eta}{\partial t}(x_i,t_n) \\
&\quad -\frac{\Delta t^2}{6}\frac{\partial^3 \eta}{\partial t^3}(x_i,\tau_1) \\
&\quad -\theta u\Bigg(\frac{\Delta x^2}{6}\frac{\partial^3 \eta}{\partial x^3}(\xi_1,t_n) - \frac{\Delta t \Delta x}{2}\frac{\partial^2}{\partial x^2}\frac{\partial \eta}{\partial t}(\xi_2,t_n) \\
&\quad +\frac{\Delta t^3}{6\Delta x}\frac{\partial^3 \eta}{\partial t^3}(x_i,\tau_2) - \frac{\Delta t^3}{6\Delta x}\frac{\partial^3 \eta}{\partial t^3}(\xi_3,\tau_3)\Bigg) \\
&\quad -(1-\theta)u\frac{\Delta x^2}{6}\frac{\partial^3 \eta}{\partial x^3}(\xi_4,t_n).
\end{aligned}$$

Note that

$$\begin{aligned}
\frac{\partial \eta}{\partial t}(x_i,t_n) &= -u\frac{\partial \eta}{\partial x}(x_i,t_n) + O(\Delta t) + O(\Delta x) \\
\frac{\partial^2 \eta}{\partial t^2}(x_i,t_n) &= u^2\frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) + O(\Delta t) + O(\Delta x).
\end{aligned}$$

Substitution yields:

$$\begin{aligned}
\frac{\partial \eta}{\partial t}(x_i,t_n) + u\frac{\partial \eta}{\partial x}(x_i,t_n) &= \frac{u\Delta x}{2}\frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) \\
&\quad -\frac{u^2\Delta t}{2}\frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) + \theta u^2 \Delta t \frac{\partial^2 \eta}{\partial x^2}(x_i,t_n) \\
&\quad +O\left(\Delta t^2\right) + O\left(\Delta x^2\right) + O(\Delta x \Delta t).
\end{aligned}$$

Rewriting and neglecting second order terms ends the proof. ■

The proposition above implies that, on an equidistant grid, the numerical diffusion coefficient of Test case 4.5 reads:

$$\frac{u\Delta x}{2}\left(1-(1-2\theta)\frac{u\Delta t}{\Delta x}\right). \tag{5.3}$$

This is illustrated in Figure 5.2 by means of a contour plot of $1-(1-2\theta)\frac{u\Delta t}{\Delta x}$ as a function of the CFL-number $\frac{u\Delta t}{\Delta x}$. The magenta line indicates the border of the region where the numerical diffusion is nonnegative:

$$\theta \geq \frac{1}{2}\left(1-\frac{\Delta x}{u\Delta t}\right). \tag{5.4}$$

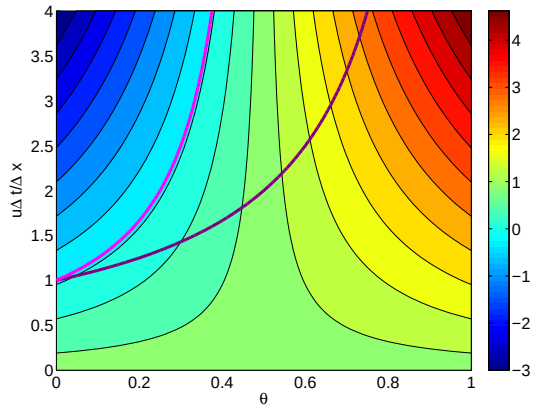


Figure 5.2: Contour plot of $1-(1-2\theta)\frac{u\Delta t}{\Delta x}$, the scaled numerical diffusion coefficient (5.3); the magenta border corresponds to (5.4), the (dark) purple border corresponds to (5.2)

The (dark) purple line indicates the border of the region where (5.2) is satisfied. Apparently, if (5.2) is satisfied, which implies stable, positive, and non-oscillatory behavior, the numerical diffusion coefficient must be strictly positive (unless $\theta = 0$ and $\frac{u\Delta t}{\Delta x} = 1$). At the same time, a strictly positive numerical diffusion coefficient causes artificial smearing of the solution (see Figure 5.1). Observe that the numerical diffusion grows with θ. Altogether, it is best to satisfy (5.2) with the smallest possible θ.

5.2 Theta FCT scheme

Section 4.3 described the explicit FCT scheme (Method 4.7), which is accurate, but not robust due to a severe stability criterion for the time step. The question is: will an implicit variant of such a scheme lead to a method that is both accurate and robust? For this purpose, the explicit FCT scheme will be generalized to the theta FCT scheme.

Method 5.3 (Theta (upwind) FCT scheme): The *theta FCT scheme* is a FVM (Method 3.7) of the following form (for $p = 0$):

$$\begin{aligned}\frac{|\mathcal{V}_i^n|\bar{c}_i^n - |\mathcal{V}_i^{n-1}|\bar{c}_i^{n-1}}{t_n - t_{n-1}} &= -\sum_{j\in\mathcal{J}_i}|\mathcal{S}_{ij}^n|\Big((1-\theta)\Big(\hat{\phi}_{ij}^{n-1} + l_{ij}^{n,n-1}\big(\tilde{\phi}_{ij}^{n-1} - \hat{\phi}_{ij}^{n-1}\big)\Big) \\ &\quad +\theta\Big(\hat{\phi}_{ij}^{n} + l_{ij}^{n,n}\big(\tilde{\phi}_{ij}^{n} - \hat{\phi}_{ij}^{n}\big)\Big)\Big) \end{aligned} \tag{5.5}$$

Again, $\hat{\phi}_{ij}$ is a non-oscillatory first order numerical flux function of the form (3.5), and $\tilde{\phi}_{ij}$ is a higher order one. The difference $\tilde{\phi}_{ij} - \hat{\phi}_{ij}$ can be interpreted as a correction term, which is limited by l_{ij}. If $\hat{\phi}_{ij}$ is as in (4.1), the *theta upwind FCT scheme* is obtained. ⌟

Note that, if $\theta = 0$, the method corresponds to the explicit FCT scheme (Method 4.7). In the implicit case ($\theta > 0$), Method 5.3 requires the solution of a nonlinear system, if the limiter depends nonlinearly on the concentration at the new time t_n. The latter is usually the case because of Godunov's barrier theorem (Theorem 4.6). The following method avoids this difficulty by approximating the flux corrections at the new time with the help of the first order solution approximation. The limiter is based on the one described in Method 4.8.

Method 5.4 ((Approximate) theta (upwind) FCT scheme à la Boris & Book): This method consists of the following steps, which involve linear systems only (see Method 5.3 for notational aspects):

1. Suppose that $\overline{c}_i^{n-1}$ is known from the previous time step. Compute a first order approximation $\hat{\overline{c}}_i^n$ by means of (5.5) with $l_{ij}^{n,n-1} = l_{ij}^{n,n} = 0$, which is equivalent to applying the theta scheme (Method 5.1) with the lower order flux function $\hat{\phi}_{ij}$:

$$\frac{|\mathcal{V}_i^n|\hat{\overline{c}}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^n| \left((1-\theta)\hat{\phi}_{ij}^{n-1} + \theta\hat{\phi}_{ij}^n\right).$$

2. Define the total flux correction $\Delta\phi_{ij}^n$ as follows:

$$\begin{aligned}\Delta\phi_{ij}^n &= (1-\theta)\left(\tilde{\phi}_{ij}^{n-1} - \hat{\phi}_{ij}^{n-1}\right) + \theta\left(\tilde{\phi}_{ij}^n - \hat{\phi}_{ij}^n\right)\\ &= (1-\theta)\left(\tilde{\phi}_{ij}(\overline{c}_i^{n-1}, \overline{c}_j^{n-1}) - \hat{\phi}_{ij}(\overline{c}_i^{n-1}, \overline{c}_j^{n-1})\right) + \theta\left(\tilde{\phi}_{ij}(\overline{c}_i^n, \overline{c}_j^n) - \hat{\phi}_{ij}(\overline{c}_i^n, \overline{c}_j^n)\right).\end{aligned}$$

   Since $\overline{c}_i^n$ is unknown, approximate $\Delta\phi_{ij}^n$ with the help of the first order approximation[1] $\hat{\overline{c}}_i^n$:

$$\Delta\phi_{ij}^n \approx (1-\theta)\left(\tilde{\phi}_{ij}(\overline{c}_i^{n-1}, \overline{c}_j^{n-1}) - \hat{\phi}_{ij}(\overline{c}_i^{n-1}, \overline{c}_j^{n-1})\right) + \theta\left(\tilde{\phi}_{ij}(\hat{\overline{c}}_i^n, \hat{\overline{c}}_j^n) - \hat{\phi}_{ij}(\hat{\overline{c}}_i^n, \hat{\overline{c}}_j^n)\right).$$

3. Apply steps 2-6 of Method 4.8 to obtain the limiter à la Boris & Book $l_{ij}^n = l_{ij}^{n,n-1} = l_{ij}^{n,n}$.

4. Compute the solution estimation at time t_n by approximating (5.5) as follows[2]:

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^n|\hat{\overline{c}}_i^n}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^n| l_{ij}^n \Delta\phi_{ij}^n. \tag{5.6}$$

⌟

Remark 5.5: Alternatively, an iterative theta FCT scheme could be formulated similar to the iterative FEM-FCT scheme that was proposed by Kuzmin et al. in [KMT, Sections 5 and 6]. This probably leads to higher accuracy, although extra computational costs also need to be taken into account. ⌟

Figure 5.3 illustrates the performance of the approximate theta FCT scheme if central fluxes are used to correct the numerical diffusion of the first order upwind scheme. The test was performed for the same pairs of θ and Δt as in Figure 5.1. The explicit case that was obtained before is displayed again in blue, to demonstrate the difference with implicit cases. Unfortunately, the results show a lot of smearing. From the proposition below it becomes clear that the numerical diffusion coefficient of the theta scheme with central fluxes reads

$$\left(\theta - \frac{1}{2}\right) u^2 \Delta t.$$

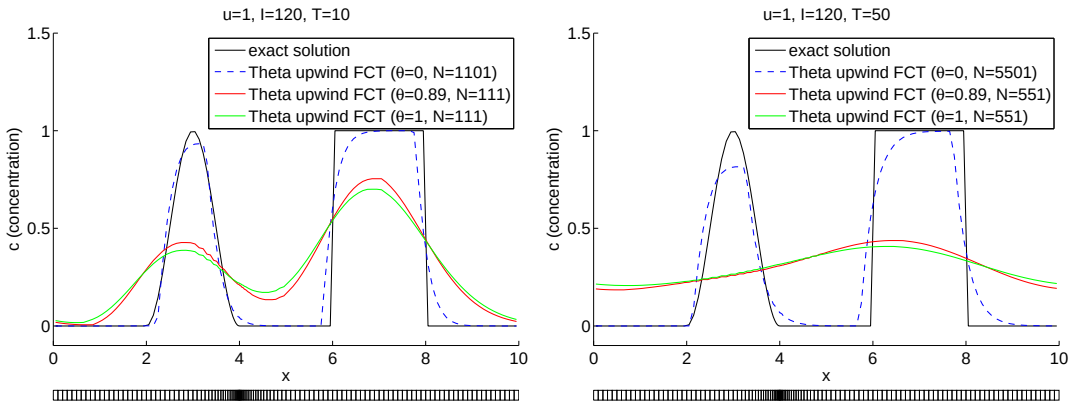
Apparently, central fluxes are only suitable to serve as flux corrector in the theta FCT scheme if θ is sufficiently close to $\frac{1}{2}$. Unfortunately, this condition often conflicts with (5.2) (unless the time step or the grid is altered which would mean that the scheme is not robust), which ensures stability, positivity, and non-oscillatory behavior of the theta upwind scheme.

Proposition 5.6: *Consider the theta scheme (Method 5.1) with central fluxes (4.3) for the one-dimensional advection equation with constant velocity $u > 0$, constant cell width $\Delta x > 0$, and constant time step $\Delta t > 0$:*

$$\Delta x \frac{\overline{c}_i^n - \overline{c}_i^{n-1}}{\Delta t} + u\frac{\overline{c}_{i+1}^n - \overline{c}_{i-1}^n}{2} + (1-\theta)u\frac{\overline{c}_{i+1}^{n-1} - \overline{c}_{i-1}^{n-1}}{2} = 0$$

[1] Alternatively, the higher order approximation, which can be obtained by means of (5.5) with $l_{ij}^{n,n-1} = l_{ij}^{n,n} = 1$, could be used. However, this would require the solution of an extra linear system. This strategy has not been tested.

[2] Instead of (5.6), (5.5) could be used with $l_{ij}^{n,n-1} = l_{ij}^{n,n} = l_{ij}^n$. However, this would not guarantee the absence of new local extremes, because the limiter is now applied to a different correction flux than it was originally designed for.



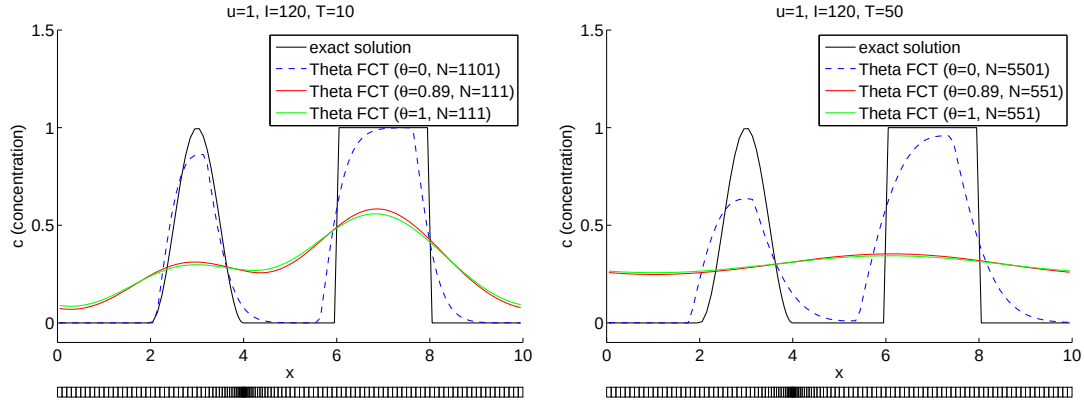
(a) 1 period (b) 5 period

Figure 5.3: Approximate theta upwind FCT scheme (Method 5.4) with central flux correction (4.3) for Test case 4.5

Let $\eta(x,t)$ be a 'suffciently differentiable' function such that $\eta(x_i,t_n) = \overline{c}_i^n$ (for all $i = 1,...,I$, for all $n = 1,...,N$). Then for all $i = 1,...,I$ and for all $n = 1,...,N$:

$$\frac{\partial \eta}{\partial t}(x_i,t_n) + u\frac{\partial \eta}{\partial x}(x_i,t_n) \approx \underbrace{\left(\theta - \frac{1}{2}\right)u^2\Delta t}_{\textit{Numerical diffusion coefficient}} \frac{\partial^2 \eta}{\partial x^2}(x_i,t_n)$$

Proof. The proof is similar to the proof of Proposition 5.2. ■



(a) 1 period (b) 5 periods

Figure 5.4: Approximate theta upwind FCT scheme (Method 5.4) with Lax-Wendroff flux correction (Method 4.11) for Test case 4.5

Figure 5.4 illustrates the effect of using second order Lax-Wendroff fluxes once more to correct the numerical diffusion of the theta upwind scheme. Again, the results show a lot of smearing. From the proposition below it becomes clear that the numerical diffusion coefficient of the theta scheme with Lax-Wendroff fluxes reads

$$\theta u^2 \Delta t.$$

Apparently, Lax-Wendroff fluxes are only suitable to serve as flux corrector in the theta FCT scheme if θ is sufficiently close to 0. Unfortunately, this condition often conflicts with (5.2).

Proposition 5.7: *Consider the theta scheme (Method 5.1) with Lax-Wendroff fluxes (see Method 4.11) for the one-dimensional advection equation with constant velocity $u > 0$, constant cell width $\Delta x > 0$, and constant time step $\Delta t > 0$:*

$$\begin{aligned} \Delta x \frac{\overline{c}_i^n - \overline{c}_i^{n-1}}{\Delta t} + \theta \left(u \frac{\overline{c}_{i+1}^n - \overline{c}_{i-1}^n}{2} - \frac{u^2 \Delta t}{2} \frac{\overline{c}_{i-1}^n - 2\overline{c}_i^n + \overline{c}_{i+1}^n}{\Delta x} \right) \\ + (1-\theta) \left(u \frac{\overline{c}_{i+1}^{n-1} - \overline{c}_{i-1}^{n-1}}{2} - \frac{u^2 \Delta t}{2} \frac{\overline{c}_{i-1}^{n-1} - 2\overline{c}_i^{n-1} + \overline{c}_{i+1}^{n-1}}{\Delta x} \right) &= 0 \end{aligned} \tag{5.7}$$

Let $\eta(x,t)$ be a 'sufficiently differentiable' function such that $\eta(x_i, t_n) = \overline{c}_i^n$ (for all $i = 1, ..., I$, for all $n = 1, ..., N$). Then for all $i = 1, ..., I$ and for all $n = 1, ..., N$:

$$\frac{\partial \eta}{\partial t}(x_i, t_n) + u \frac{\partial \eta}{\partial x}(x_i, t_n) \approx \underbrace{\theta u^2 \Delta t}_{\text{Numerical diffusion coefficient}} \frac{\partial^2 \eta}{\partial x^2}(x_i, t_n)$$

Proof. The proof is similar to the proof of Proposition 5.2. ■

Actually, it is rather peculiar to combine the theta scheme with Lax-Wendroff fluxes, because the Lax-Wendroff fluxes include time discretisation already. For this reason, now, the implicit counterpart of the Lax-Wendroff scheme will be considered, which can be derived in a similar manner as the (explicit) Lax-Wendroff scheme (Method 4.11). As a matter of fact, except for one single minus sign, it leads to the exact same scheme.

Method 5.8 (Implicit Lax-Wendroff scheme): The implicit Lax-Wendroff scheme is a fully discrete FVM (see Method 3.7 for notational aspects) of the following form (for $p = 0$ and $\underline{\underline{\tilde{D}}} = 0$):

$$\frac{|\mathcal{V}_i^n| \overline{c}_i^n - |\mathcal{V}_i^{n-1}| \overline{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j \in \mathcal{J}_i} |\mathcal{S}_{ij}^n| \phi_{ij}^n,$$

where

$$\phi_{ij}^n = \hat{\phi}_{ij}^n + \left(\frac{\Delta \xi_{ij}^n}{\|\underline{\mathbf{x}}_j^n - \underline{\mathbf{x}}_i^n\|_2} + \frac{\underline{\mathbf{u}}_{ij}^n \cdot \underline{\mathbf{n}}_{ij}^n (t_n - t_{n-1})}{2 \|\underline{\mathbf{x}}_j^n - \underline{\mathbf{x}}_i^n\|_2} \right) \underline{\mathbf{u}}_{ij}^n \cdot \underline{\mathbf{n}}_{ij}^n \left(\overline{c}_j^n - \overline{c}_i^n \right).$$

Here, $\hat{\phi}_{ij}^n$ is the first order upwind flux:

$$\hat{\phi}_{ij}^n = \max\{\underline{\overline{\mathbf{u}}}_{ij}^n \cdot \underline{\mathbf{n}}_{ij}^n, 0\} \overline{c}_i^n + \min\{\underline{\overline{\mathbf{u}}}_{ij}^n \cdot \underline{\mathbf{n}}_{ij}^n, 0\} \overline{c}_j^n,$$

and $\Delta \xi_{ij}^n$ is as in Method 4.11.

Derivation. First, reduce the model to the one-dimensional advection equation with constant velocity $u > 0$, constant cell width $\Delta x > 0$, and constant time step $\Delta t > 0$. Consider the following Taylor expansion around $t = t_n$:

$$c(x, t_{n-1}) \approx c(x, t_n) - \Delta t \frac{\partial c}{\partial t}(x, t_n) + \frac{\Delta t^2}{2} \frac{\partial^2 c}{\partial t^2}(x, t_n).$$

Now, the same strategy that led to the explicit Lax-Wendroff scheme (Method 4.11) can be used to obtain the implicit Lax-Wendroff scheme. ■

A convex combination of the (explicit) Lax-Wendroff scheme and the implicit Lax-Wendroff scheme leads to the theta Lax-Wendroff scheme:

Method 5.9 (Theta Lax-Wendroff scheme): The *theta Lax-Wendroff scheme* is a FVM (Method 3.7) that is the following convex combination of the original Lax-Wendroff scheme (Method 4.11) and the implicit Lax-Wendroff scheme (Method 5.8) (for $p = 0$ and $\underline{\underline{\tilde{D}}} = 0$):

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i}\left((1-\theta)|\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1} + \theta|\mathcal{S}_{ij}^n|\psi_{ij}^n\right),$$

where $\theta \in [0,1]$ is a constant, and

$$\begin{aligned}\phi_{ij}^{n-1} &= \hat{\phi}_{ij}^{n-1} + \left(\frac{\Delta\xi_{ij}^{n-1}}{\|\underline{\mathbf{x}}_j^{n-1} - \underline{\mathbf{x}}_i^{n-1}\|_2} - \frac{\underline{\mathbf{u}}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1}(t_n - t_{n-1})}{2\|\underline{\mathbf{x}}_j^{n-1} - \underline{\mathbf{x}}_i^{n-1}\|_2}\right)\underline{\mathbf{u}}_{ij}^{n-1}\cdot\underline{\mathbf{n}}_{ij}^{n-1}\left(\overline{c}_j^{n-1} - \overline{c}_i^{n-1}\right),\\ \psi_{ij}^n &= \hat{\phi}_{ij}^n + \left(\frac{\Delta\xi_{ij}^n}{\|\underline{\mathbf{x}}_j^n - \underline{\mathbf{x}}_i^n\|_2} + \frac{\underline{\mathbf{u}}_{ij}^n\cdot\underline{\mathbf{n}}_{ij}^n(t_n - t_{n-1})}{2\|\underline{\mathbf{x}}_j^n - \underline{\mathbf{x}}_i^n\|_2}\right)\underline{\mathbf{u}}_{ij}^n\cdot\underline{\mathbf{n}}_{ij}^n\left(\overline{c}_j^n - \overline{c}_i^n\right).\end{aligned}$$

Here $\hat{\phi}_{ij}^n$ is the first order upwind flux:

$$\hat{\phi}_{ij}^n = \max\{\underline{\overline{\mathbf{u}}}_{ij}^n\cdot\underline{\mathbf{n}}_{ij}^n, 0\}\overline{c}_i^n + \min\{\underline{\overline{\mathbf{u}}}_{ij}^n\cdot\underline{\mathbf{n}}_{ij}^n, 0\}\overline{c}_j^n,$$

and $\Delta\xi_{ij}^n$ is as in Method 4.11. ⌟

Unfortunately, the theta Lax-Wendroff scheme is unstable for $\theta > \frac{1}{2}$, as becomes clear from the following proposition.

Proposition 5.10: *Consider the theta Lax-Wendroff scheme (Method 5.9) for the one-dimensional advection equation with constant velocity $u > 0$, constant cell width $\Delta x > 0$, and constant time step $\Delta t > 0$:*

$$\begin{aligned}&\Delta x\frac{\overline{c}_i^n - \overline{c}_i^{n-1}}{\Delta t} + \theta\left(u\frac{\overline{c}_{i+1}^n - \overline{c}_{i-1}^n}{2} + \frac{u^2\Delta t}{2}\frac{\overline{c}_{i-1}^n - 2\overline{c}_i^n + \overline{c}_{i+1}^n}{\Delta x}\right)\\ &+(1-\theta)\left(u\frac{\overline{c}_{i+1}^{n-1} - \overline{c}_{i-1}^{n-1}}{2} - \frac{u^2\Delta t}{2}\frac{\overline{c}_{i-1}^{n-1} - 2\overline{c}_i^{n-1} + \overline{c}_{i+1}^{n-1}}{\Delta x}\right) = 0.\end{aligned} \tag{5.8}$$

If $\theta \in (\frac{1}{2}, 1]$, then this scheme is unstable for all $\nu := \frac{u\Delta t}{\Delta x} \in (0,1)$.

Proof. Rewrite the scheme to obtain:

$$\begin{aligned}&\theta\left(-\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)\overline{c}_{i-1}^n + \left(1-\theta\nu^2\right)\overline{c}_i^n + \theta\left(\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)\overline{c}_{i+1}^n =\\ &\quad(1-\theta)\left(\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)\overline{c}_{i-1}^{n-1} + \left(1-(1-\theta)\nu^2\right)\overline{c}_i^{n-1} + (1-\theta)\left(-\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)\overline{c}_{i+1}^{n-1}.\end{aligned}$$

Substitution of $c_i^n = \gamma_\xi^n e^{ij\xi}$, in which j is the imaginary unit, yields:

$$\begin{aligned}&\left(\theta\left(-\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{-j\xi} + \left(1-\theta\nu^2\right) + \theta\left(\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{j\xi}\right)\gamma_\xi^n e^{ij\xi} =\\ &\quad\left((1-\theta)\left(\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{-j\xi} + \left(1-(1-\theta)\nu^2\right) + (1-\theta)\left(-\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{j\xi}\right)\gamma_\xi^{n-1}e^{ij\xi}.\end{aligned}$$

Define the amplification factor g as:

$$\begin{aligned}g(\xi,\nu,\theta) &:= \frac{\gamma_\xi^n}{\gamma_\xi^{n-1}}\\ &= \frac{(1-\theta)\left(\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{-j\xi} + \left(1-(1-\theta)\nu^2\right) + (1-\theta)\left(-\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{j\xi}}{\theta\left(-\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{-j\xi} + \left(1-\theta\nu^2\right) + \theta\left(\frac{1}{2}\nu + \frac{1}{2}\nu^2\right)e^{j\xi}}.\end{aligned}$$

Substitution of $e^{j\xi} = \cos(\xi) + j\sin(\xi)$ yields:

$$g(\xi,\nu,\theta) \quad = \quad \frac{1-(1-\theta)\nu^2+(1-\theta)\nu^2\cos(\xi)-j(1-\theta)\nu\sin(\xi)}{1-\theta\nu^2+\theta\nu^2\cos(\xi)+j\theta\nu\sin(\xi)}.$$

The Von Neumann condition reads:

$$\forall\xi\in\mathbb{R}:\ |g(\xi,\nu,\theta)|\leq 1.$$

This is a necessary and sufficient condition for absolute stability (see [Wes01, p. 175]). Rewriting gives that

$$\left|g\left(\frac{1}{2}\pi,\nu,\theta\right)\right|^2 = \frac{1+\nu^4-\nu^2+(\nu^4+\nu^2)\theta^2-2\nu^4\theta}{1+(\nu^4+\nu^2)\theta^2-2\nu^2\theta} = 1+\frac{(\nu^4-\nu^2)(1-2\theta)}{(1-\nu^2\theta)^2+\nu^2\theta^2} > 1$$

for all $\theta\in(\frac{1}{2},1]$ and $\nu\in(0,1)$. Indeed, the Von Neumann condition is not satisfied. Illustrations of $|g(\xi,0.5,\theta)|$ and $|g(\xi,0.9,\theta)|$ can be found in Figure 5.5. The red area contains all values that are larger than 1. ■

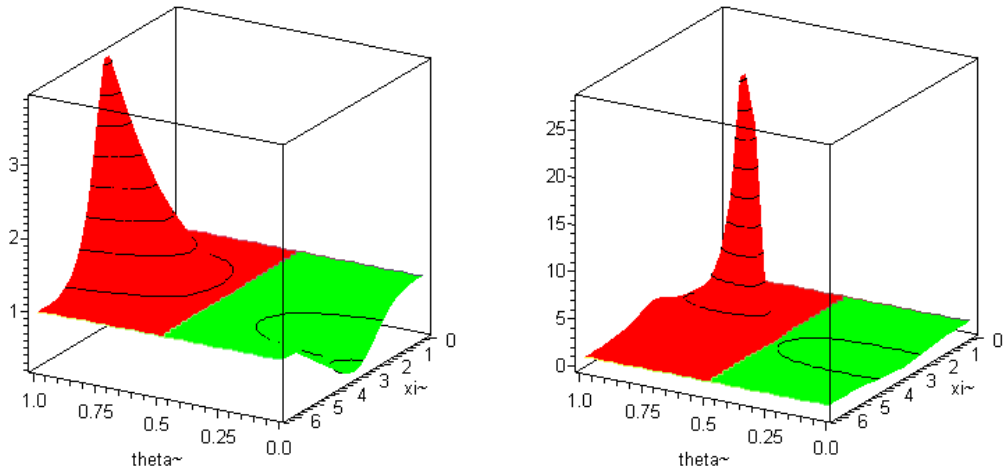
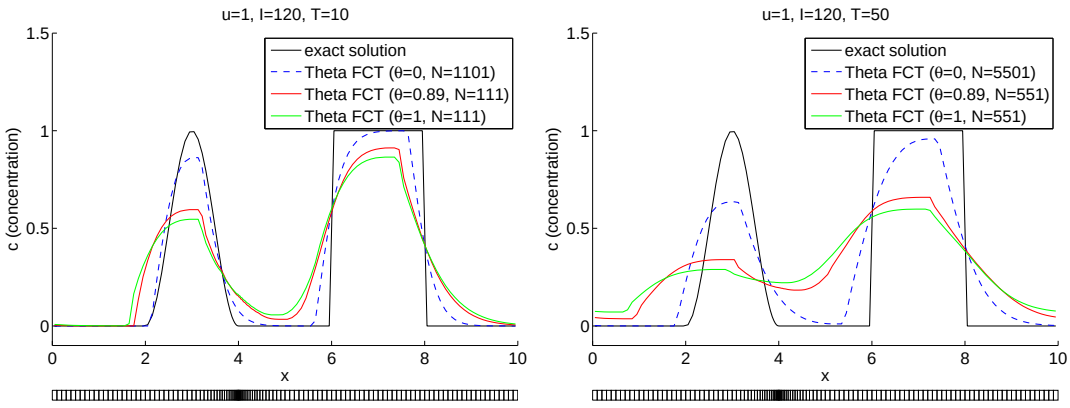


Figure 5.5: Illustration of $|g(\xi,0.5,\theta)|$ (left) and $|g(\xi,0.9,\theta)|$ (right)

Although the theta Lax-Wendroff scheme is unstable for $\theta\in(\frac{1}{2},1]$, it can still be used in the approximate FCT scheme. The reason is that the limiter, which does not allow new local extremes, will render the scheme stable in any case. However, the unstability may blow up the error that is introduced as a result of the fact that the flux corrections at the new time are approximated. This may lead to a severe limiter, i.e. a limiter with a value close to zero.

Figure 5.6 illustrates the effect of using the theta Lax-Wendroff scheme to correct the numerical diffusion of the theta upwind scheme. Indeed, the numerical diffusion has reduced a little, but not as much as in the explicit case. The main reason is that, as θ becomes larger, the theta upwind scheme needs more flux correction, as it suffers more from numerical diffusion (see Proposition 5.2 and Figure 5.2), whereas the flux correction becomes less accurate, because the flux corrections at the new time are approximations. Altogether, the theta Lax-Wendroff scheme does not lead to satisfactory accuracy if it serves as flux corrector in the approximated theta upind FCT scheme.



(a) 1 period (b) 5 periods

Figure 5.6: Approximate theta upwind FCT scheme (Method 5.4) with theta Lax-Wendroff flux correction (Method 5.9) for Test case 4.5

5.3 Summary

The theta upwind scheme is robust, but inaccurate due to numerical diffusion, which grows with θ. The robustness results from the fact that the scheme is stable, positivity preserving and non-oscillatory, provided that θ is sufficiently large. Scheme 16 of WAQ is the theta upwind scheme for $\theta = 1$. As in the explicit case, it is possible to attempt to improve the accuracy by applying the flux corrected transport algorithm. For this purpose, the explicit FCT scheme has been generalised to the theta FCT scheme. Implicit nonlinear systems can be avoided by approximating the flux corrections at the new time by means of the first order solution estimation. Because Lax-Wendroff fluxes are not suitable as flux correctors if θ is not close to zero, and central fluxes are unsuitable if θ is not close to $\frac{1}{2}$, the theta Lax-Wendroff scheme has been considered. Unfortunately, the approximated theta FCT scheme did not lead to satisfactory accuracy with theta Lax-Wendroff flux correction. The main reason is that, as θ becomes larger, the theta upwind scheme needs more flux correction, as it suffers more from numerical diffusion, whereas the flux correction becomes less accurate, because the flux corrections at the new time are approximations.

Chapter 6

Local-theta scheme

Chapter 5 described the robust theta upwind scheme, which is inaccurate due to numerical diffusion. This is strongly related to the time-discretisation, since the numerical diffusion grows with θ (see Proposition 5.2 and Figure 5.2). For this reason, this chapter investigates whether a minimal *local* value for theta will lead to better accuracy, without loss of robustness.

6.1 Local-theta scheme

Below, the theta scheme is generalised to the local-theta scheme.

Method 6.1 (**Local-theta (upwind) scheme**)**:** The *local-theta scheme* is a FVM (Method 3.7) of the following form:

$$\frac{|\mathcal{V}_i^n|\overline{c}_i^n - |\mathcal{V}_i^{n-1}|\overline{c}_i^{n-1}}{t_n - t_{n-1}} \quad = \quad -\sum_{j\in\mathcal{J}_i}\Big(\left(1-\theta_{ij}^n\right)|\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1} + \theta_{ij}^n|\mathcal{S}_{ij}^n|\phi_{ij}^n\Big).$$

Here, ϕ_{ij} is a numerical flux function, and $\theta_{ij}^n \in [0,1]$ is still free to be chosen such that the scheme has nice properties (see below). If ϕ_{ij} is as in (4.1), the *local-theta upwind scheme* is obtained. ⌟

On the one hand, the local theta coeffients θ_{ij}^n should be as small as possible, to minimize the amount of numerical diffusion. On the other hand they need to be large enough to ensure that the scheme is stable, positivity preserving, and non-oscillatory. Sufficient conditions for these properties are constructed below, by deriving a local and a global discrete maximum principle with the help of the following lemma.

Lemma 6.2: *Let γ_j and c_j (j=1,...,J) be reals satisfying*

$$\sum_{j=1}^{J}\gamma_j c_j = 0, \qquad \sum_{j=1}^{J}\gamma_j = 0,\ \gamma_2,...\gamma_J < 0,\ J \geq 2.$$

Then, either

$$c_1 = c_2 = ... = c_J$$

or

$$\min_{j=2,...,J} c_j < c_1 < \max_{j=2,...,J} c_j.$$

Proof. (See also [Wes01, Lemma 4.4.1]) First of all, note that $\gamma_1 = -\sum_{j=2}^{J} \gamma_j > 0$. Hence, $\sum_{j=1}^{J} \gamma_j c_j = 0$ can be rewritten to obtain:

$$c_1 \quad = \quad \sum_{j=2}^{J} \underbrace{\frac{-\gamma_j}{\gamma_1}}_{>0} c_j.$$

Using $\sum_{j=1}^{J} \gamma_j = 0$ once more, it follows that

$$\sum_{j=2}^{J} \frac{-\gamma_j}{\gamma_1} = \frac{\gamma_1}{\gamma_1} = 1.$$

Apparently, c_1 is a convex combination of $c_2, .., c_J$. As a result,

$$c_{\min} := \min_{j=2,...,J} c_j \leq c_1 \leq \max_{j=2,...,J} c_j =: c_{\max}.$$

Now, there are two options.

1. If $c_{\min} = c_{\max}$, then, $c_1 = c_2 = ... = c_J$.
2. If $c_{\min} < c_{\max}$, then there exist $j_{\min}, j_{\max} \geq 2$ such that

   $$c_{j_{\min}} = c_{\min} < c_{\max} = c_{j_{\max}}.$$

   As a result,

   $$c_1 = \sum_{j=2}^{J} \frac{-\gamma_j}{\gamma_1} c_j \leq \sum_{j=2, j\neq j_{min}}^{J} \frac{-\gamma_j}{\gamma_1} c_{\max} + \frac{-\gamma_{j_{\min}}}{\gamma_1} c_{\min} < \sum_{j=2}^{J} \frac{-\gamma_j}{\gamma_1} c_{\max} = c_{\max}$$

   Similarly, it can be shown that $c_1 > c_{\min}$. ■

Theorem 6.3: *Consider the local-theta scheme (Method (6.1)) for flux functions of the form:*

$$-|\mathcal{S}_{ij}|\phi_{ij} \quad = \quad -\gamma_{ji}\bar{c}_i + \gamma_{ij}\bar{c}_j, \qquad \gamma_{ij} \geq 0.$$

Let $\theta_{ij} \in [0,1]$ be symmetrical:

$$\theta_{ij} \quad = \quad \theta_{ji}. \tag{6.1}$$

Moreover, let the system be consistent in the sense that

$$\frac{|\mathcal{V}_i^n| - |\mathcal{V}_i^{n-1}|}{t_n - t_{n-1}} - \sum_{j\in\mathcal{J}_i} \Big(\left(1-\theta_{ij}^n\right)\left(-\gamma_{ji}^{n-1} + \gamma_{ij}^{n-1}\right) + \theta_{ij}^n \left(-\gamma_{ji}^n + \gamma_{ij}^n\right) \Big) \quad = \quad 0. \tag{6.2}$$

Moreover, assume that the following generalistaion of the CFL-condition holds:

$$|\mathcal{V}_i^{n-1}| - \sum_{j\in\mathcal{J}_i} (t_n - t_{n-1})(1-\theta_{ij}^n)\gamma_{ji}^{n-1} \quad \geq \quad 0. \tag{6.3}$$

If, furthermore, for all $i = 1, ..., I$, either $(1-\theta_{ij}^n)\gamma_{ij}^{n-1} > 0$ for some $j \in \mathcal{J}_i$, or (6.3) is satisfied strictly[1], then, the scheme

I. is mass-conservative,

II. satisfies the local discrete maximum principle (see Definition 3.20),

[1] This condition is not necesary and can be neglected in practice.

III. admits the global discrete maximum principle (see Definition 3.21).

Proof.

I. First of all, observe that the scheme reads:

$$\frac{|\mathcal{V}_i^n|\bar{c}_i^n - |\mathcal{V}_i^{n-1}|\bar{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i} \underbrace{\left((1-\theta_{ij}^n)|\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1} + \theta_{ij}^n|\mathcal{S}_{ij}^n|\phi_{ij}^n\right)}_{=:\psi_{ij}^n}. \tag{6.4}$$

Furthermore, note that the flux function ϕ_{ij} is anti-symmetric ($\phi_{ij} = -\phi_{ji}$):

$$-|\mathcal{S}_{ij}|\phi_{ij} = -\gamma_{ji}\bar{c}_i + \gamma_{ij}\bar{c}_j = |\mathcal{S}_{ji}|\phi_{ji} = |\mathcal{S}_{ij}|\phi_{ji}. \tag{6.5}$$

Hence, ψ_{ij} is also anti symmetric:

$$\begin{array}{rcll} \psi_{ji}^n & & (1-\theta_{ji}^n)|\mathcal{S}_{ji}^{n-1}|\phi_{ji}^{n-1} & +\theta_{ji}^n|\mathcal{S}_{ji}^n|\phi_{ji}^n \\ & \overset{(6.1)}{=} & (1-\theta_{ij}^n)|\mathcal{S}_{ji}^{n-1}|\phi_{ji}^{n-1} & +\theta_{ij}^n|\mathcal{S}_{ji}^n|\phi_{ji}^n \\ & \overset{(6.5)}{=} & -(1-\theta_{ij}^n)|\mathcal{S}_{ij}^{n-1}|\phi_{ij}^{n-1} & -\theta_{ij}^n|\mathcal{S}_{ij}^n|\phi_{ij}^n \quad = \quad -\psi_{ij}^n. \end{array} \tag{6.6}$$

As a result, in the absence of an open boundary ($\partial D_1 = \varnothing$),

$$\sum_{i=1}^{I} \frac{|\mathcal{V}_i^n|\bar{c}_i^n - |\mathcal{V}_i^{n-1}|\bar{c}_i^{n-1}}{t_n - t_{n-1}} \overset{(6.4)}{=} -\sum_{i=1}^{I}\sum_{j\in\mathcal{J}_i} \psi_{ij}^n \overset{(6.6)}{=} 0,$$

where it has been used that each ψ_{ij}^n is canceled by ψ_{ji}^n, since $j \in \mathcal{J}_i \Leftrightarrow i \in \mathcal{J}_j$. As a consequence, the scheme is mass conservative.

II. Note that the scheme can be written in the form:

$$a_{ii}^n\bar{c}_i^n + \sum_{j\in\mathcal{A}_i^n} a_{ij}^n\bar{c}_j^n \quad = \quad \sum_{j\in\mathcal{B}_i^n} b_{ij}^n\bar{c}_j^{n-1}, \qquad i = 1,..,I,$$

where

$$\begin{aligned} a_{ii}^n &= |\mathcal{V}_i^n| + (t_n - t_{n-1})\sum_{j\in\mathcal{J}_i}\theta_{ij}^n\gamma_{ji}^n, \\ a_{ij}^n &= -(t_n - t_{n-1})\theta_{ij}^n\gamma_{ij}^n, \\ b_{ii}^n &= |\mathcal{V}_i^{n-1}| - (t_n - t_{n-1})\sum_{j\in\mathcal{J}_i}(1-\theta_{ij}^n)\gamma_{ji}^{n-1}, \\ b_{ij}^n &= (t_n - t_{n-1})(1-\theta_{ij}^n)\gamma_{ij}^{n-1}, \end{aligned}$$

and

$$\begin{aligned} \mathcal{A}_i^n &= \{j \in \mathcal{J}_i : a_{ij} \neq 0\}, \\ \mathcal{B}_i^n &= \{j \in \mathcal{J}_i \cup \{i\} : b_{ij} \neq 0\}. \end{aligned}$$

Observe that

1. $a_{ii}^n > 0$ and $a_{ij}^n < 0$ ($j \in \mathcal{A}_i^n$), since $\gamma_{ij}^n \geq 0$ (and $\theta_{ij}^n \geq 0$) for all i and j,
2. $b_{ij}^n > 0$ ($j \in \mathcal{B}_i^n$), because of (6.3) and $\gamma_{ij}^{n-1} \geq 0$ (and $1 - \theta_{ij}^{n-1} \geq 0$) for all i and j,
3. the sum of the coefficients is zero in the sense that

$$a_{ii}^n + \sum_{j\in\mathcal{A}_i^n} a_{ij}^n + \sum_{j\in\mathcal{B}_i^n} -b_{ij}^n = 0,$$

as a result of consistency (6.2).

Now, the local discrete maximum principle follows immediately from Lemma 6.2 for $\gamma_1 = a_{ii} > 0$ and $c_1 = \overline{c}_i^n$.

III. To obtain the global maximum principle, observe that, by definition, the claim is true for $n = 0$. Now suppose that the claim is true, i.e. a path with non-decreasing values exists for each concentration (see Definition 3.21), up to time t_{n-1}. Next, consider $\overline{c}_i^n$ for some $i \in \{1, ..., I\}$. Now, the local maximum principle leads to one of the following options:

1. (3.7) holds. Because either (6.3) is satisfied strictly, or $(1 - \theta_{ij}^n)\gamma_{ij}^{n-1} > 0$ for some $j \in \mathcal{J}_i$, $\mathcal{B}_i^n$ is nonempty. Hence, $\overline{c}_i^n = \overline{c}_j^{n-1}$ for some $j \in \mathcal{B}_i^n$. As a path with non-decreasing values exists for $\overline{c}_j^{n-1}$, the same path can be used for $\overline{c}_i^n$ and the claim is true.
2. (3.8) holds. Again there are two options
   i. $\overline{c}_i^n < \overline{c}_j^{n-1}$ for some $j \in \mathcal{B}_i^n$. As a path with non-decreasing values exists for $\overline{c}_j^{n-1}$, the same path can be used for $\overline{c}_i^n$ and the claim is true.
   ii. $\overline{c}_i^n < \overline{c}_j^n$ for some $j \in \mathcal{A}_i^n$. Once more, there are two options.
      a. $j \in \{I + 1, ..., I + K\}$ indicates a boundary element. This implies that an non-decreasing path to the boundary has been found, in only one step.
      b. $j \in \{1, ..., I\}$ indicates an interior element. In this case, the argument can be repeated without visiting this index again, because $\overline{c}_i^n < \overline{c}_j^n$ is a strict inequality.

As a consequence a path with increasing values can be found for all $i = 1, ..., I + K$ and all $n = 0, ..., N$. Similarly, a path with non-increasing values can be found. ■

A direct consequence of the local and the global discrete maximum principle is that the scheme is absolutely stable with respect to $\|.\|_\infty$, positivity preserving, and non-oscillatory (see Section 3.4.2).

For the local-theta scheme, (6.2) and (6.3) become

$$\frac{|\mathcal{V}_i^n| - |\mathcal{V}_i^{n-1}|}{t_n - t_{n-1}} = -\sum_{j \in \mathcal{J}_i} \left((1 - \theta_{ij}^n)|\mathcal{S}_{ij}^{n-1}|\underline{\overline{\mathbf{u}}}_{ij}^{n-1} \cdot \underline{\mathbf{n}}_{ij}^{n-1} + \theta_{ij}^n |\mathcal{S}_{ij}^n| \underline{\overline{\mathbf{u}}}_{ij}^n \cdot \underline{\mathbf{n}}_{ij}^n \right), \tag{6.7}$$

and

$$|\mathcal{V}_i^{n-1}| - (t_n - t_{n-1}) \sum_{j \in \mathcal{J}_i} (1 - \theta_{ij}^n)|\mathcal{S}_{ij}^{n-1}| \left(\max\{0, \underline{\overline{\mathbf{u}}}_{ij}^{n-1} \cdot \underline{\mathbf{n}}_{ij}^{n-1}\} + \frac{\overline{d}_{ij}^{n-1}}{\|\underline{\mathbf{x}}_j^{n-1} - \underline{\mathbf{x}}_i^{n-1}\|_2} \right) \geq 0. \tag{6.8}$$

The consistency condition (6.7) is always satisfied in WAQ. Therefore, this condition does not have to be taken into account during the implementation. A practical strategy for obtaining (nearly) optimal values of θ_{ij}^n reads as follows.

Example 6.4: Consider the local-theta upwind scheme (Method 6.1). Assume that (6.7) is satisfied. Theorem 6.3 (practically) holds if the coefficients θ_{ij}^n are chosen as follows:

1. Define the following auxiliary coefficient for each grid cell:

$$\theta_i^n = 1 - \frac{|V_i^{n-1}|}{(t_n - t_{n-1}) \sum_{j \in \mathcal{J}_i} |\mathcal{S}_{ij}^{n-1}| \left(\max\{0, \underline{\overline{\mathbf{u}}}_{ij}^{n-1} \cdot \underline{\mathbf{n}}_{ij}^{n-1}\} + \frac{\overline{d}_{ij}^{n-1}}{\|\underline{\mathbf{x}}_j^{n-1} - \underline{\mathbf{x}}_i^{n-1}\|_2} \right)}, \qquad i = 1, ..., I.$$

   Note that (6.8) is satisfied for grid cell i, if $\theta_{ij}^n = \max\{0, \theta_i^n\}$ for all $j \in \mathcal{J}_i$.

2. Now choose

$$\theta_{ij}^n = \max\left\{0, \theta_i^n, \theta_j^n\right\}.$$

   Note that this ensures that both (6.8) and the symmetry condition (6.1) are satisfied. ⌟

Remark 6.5: In the example above, θ_i^n 'blames' each face of cell i equally if (6.8). Alternatively, only the face with the largest incoming flux could be blamed, by using a relatively large nonzero local theta coefficient for that face only. After that, the coefficients need to be updated to ensure symmetry. Another strategy is to use weighted coefficients in accordance with the size of the incoming flux. ⌟

Remark 6.6: In order to ensure that, for all $i = 1, ..., I$, either $(1-\theta_{ij}^n)\gamma_{ij}^{n-1} > 0$ for some $j \in \mathcal{J}_i$, or (6.8) is satisfied strictly, whenever necesary, the auxiliary coefficients θ_i^n could be raised a little in order to satisfy (6.8) strictly. However, this option has not been implemented in WAQ. First of all, because the condition is not a necesary (yet sufficient) condition to prove the results of Theorem (6.3). Moreover, the modification is only necesary in highly exceptional cases to satisfy the condition. In addition, the absence of this modification did not lead to unstable, negative, or oscillatory results during the testcases. ⌟

Figure 6.1 displays the performance of the local-theta upwind scheme for Test case 4.5. For this test case, the local-theta upwind scheme reads:

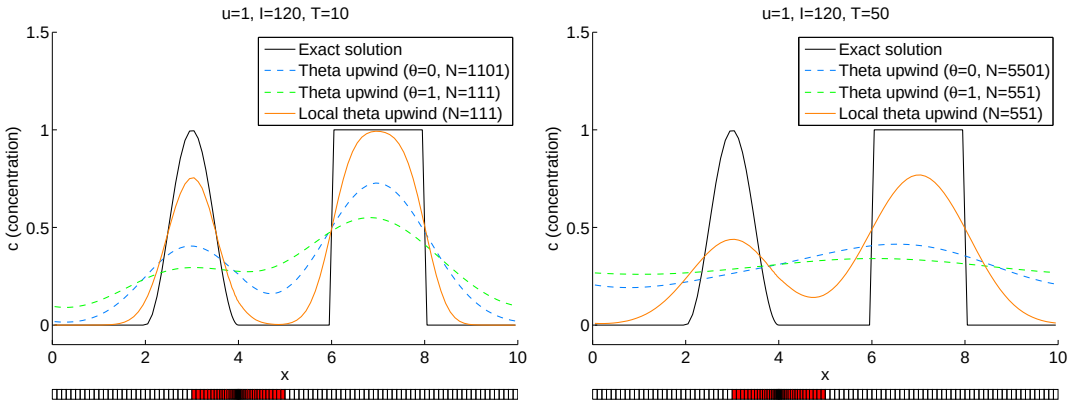
$$\frac{\Delta x_i \overline{c}_i^n - \Delta x_i \overline{c}_i^{n-1}}{\Delta t} = -(1-\theta_{i,i+1}^n)u\overline{c}_i^{n-1} + (1-\theta_{i,i-1}^n)u\overline{c}_{i-1}^{n-1} - \theta_{i,i+1}^n u\overline{c}_i^n + \theta_{i,i-1}^n u\overline{c}_{i-1}^n, \qquad (6.9)$$

where $\Delta t = t_n - t_{n-1}$ is the constant time step and $\Delta x_i = |\mathcal{V}_i^{n-1}| = |\mathcal{V}_i^n|$ is the cell width. The scheme will be stable, positivity preserving and non-oscillatory if

$$\theta_{i+1,i}^n = \theta_{i,i+1}^n = 1 - \frac{\Delta x_i}{u\Delta t}, \qquad i = 1, ..., I, \qquad (6.10)$$

which follows from simplifying (6.8) and Example 6.4.

Figure 6.1 illustrates that, without loss of robustness, the local-theta upwind scheme suffers less from numerical diffusion than the theta upwind scheme.



(a) 1 period (b) 5 periods

Figure 6.1: Local-theta upwind scheme ((6.9) and (6.10)) compared to the theta upwind scheme (5.1) for Test case 4.5

6.2 Flux corrected transport

In the previous section, the numerical diffusion of the theta upwind scheme was reduced by switching from a global theta to a local value. Once more, it is possible to use the flux corrected transport

algorithm to reduce the numerical diffusion even more. The strategy is quite similar to the methods described in Section 5.2. As a matter of fact, the local-theta FCT scheme simply results from replacing θ by θ_{ij}^n in the theta FCT scheme (Method 5.3).

Method 6.7 (Local-theta (upwind) FCT scheme): The *local-theta FCT scheme* is a FVM (Method 3.7) of the following form (for $p = 0$):

$$\begin{aligned}\frac{|\mathcal{V}_i^n|\bar{c}_i^n - |\mathcal{V}_i^{n-1}|\bar{c}_i^{n-1}}{t_n - t_{n-1}} &= -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^n| \Big((1-\theta_{ij}^n)\Big(\hat{\phi}_{ij}^{n-1} + l_{ij}^{n,n-1}\big(\tilde{\phi}_{ij}^{n-1} - \hat{\phi}_{ij}^{n-1}\big)\Big) \\ &\quad + \theta_{ij}^n \Big(\hat{\phi}_{ij}^{n} + l_{ij}^{n,n}\big(\tilde{\phi}_{ij}^{n} - \hat{\phi}_{ij}^{n}\big)\Big)\Big) \end{aligned} \tag{6.11}$$

Again, $\hat{\phi}_{ij}$ is a non-oscillatory first order numerical flux function of the form (3.5), and $\tilde{\phi}_{ij}$ is a higher order one. The difference $\tilde{\phi}_{ij} - \hat{\phi}_{ij}$ can be interpreted as a correction term, which is limited by l_{ij}. If $\hat{\phi}_{ij}$ is as in (4.1), the *local-theta upwind FCT scheme* is obtained. ⌟

Again, the problem of the nonlinear limiter comes up, which calls for the solution of a nonlinear system in the implicit case. This problem can be tackled very similar to Method 5.4.

Method 6.8 ((Approximate) local-theta (upwind) FCT scheme à la Boris & Book): This method consists of the following steps, which involve linear systems only (see Method 6.7 for notational aspects):

1. Suppose that $\bar{c}_i^{n-1}$ is known from the previous time step. Compute a first order approximation $\hat{\bar{c}}_i^n$ by means of (6.11) with $l_{ij}^{n,n-1} = l_{ij}^{n,n} = 0$, which is equivalent to applying the local-theta scheme (Method 6.1) with the first order flux function $\hat{\phi}_{ij}$:

$$\frac{|\mathcal{V}_i^n|\hat{\bar{c}}_i^n - |\mathcal{V}_i^{n-1}|\bar{c}_i^{n-1}}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^n| \left((1-\theta_{ij}^n)\hat{\phi}_{ij}^{n-1} + \theta_{ij}^n \hat{\phi}_{ij}^n \right).$$

2. Define the total flux correction $\Delta\phi_{ij}^n$ as follows:

$$\begin{aligned}\Delta\phi_{ij}^n &= (1-\theta_{ij}^n)\left(\tilde{\phi}_{ij}^{n-1} - \hat{\phi}_{ij}^{n-1}\right) + \theta_{ij}^n \left(\tilde{\phi}_{ij}^{n} - \hat{\phi}_{ij}^{n}\right) \\ &= (1-\theta_{ij}^n)\left(\tilde{\phi}_{ij}(\bar{c}_i^{n-1}, \bar{c}_j^{n-1}) - \hat{\phi}_{ij}(\bar{c}_i^{n-1}, \bar{c}_j^{n-1})\right) + \theta_{ij}^n \left(\tilde{\phi}_{ij}(\bar{c}_i^{n}, \bar{c}_j^{n}) - \hat{\phi}_{ij}(\bar{c}_i^{n}, \bar{c}_j^{n})\right).\end{aligned}$$

   Since $\bar{c}_i^n$ is unknown, approximate $\Delta\phi_{ij}^n$ with the help of the first order approximation $\hat{\bar{c}}_i^n$:

$$\Delta\phi_{ij}^n \approx (1-\theta_{ij}^n)\left(\tilde{\phi}_{ij}(\bar{c}_i^{n-1}, \bar{c}_j^{n-1}) - \hat{\phi}_{ij}(\bar{c}_i^{n-1}, \bar{c}_j^{n-1})\right) + \theta_{ij}^n \left(\tilde{\phi}_{ij}(\hat{\bar{c}}_i^{n}, \hat{\bar{c}}_j^{n}) - \hat{\phi}_{ij}(\hat{\bar{c}}_i^{n}, \hat{\bar{c}}_j^{n})\right).$$

3. Apply steps 2-6 of Method 4.8 to obtain the limiter à la Boris & Book $l_{ij}^{n,n-1} = l_{ij}^{n,n} = l_{ij}^n$.

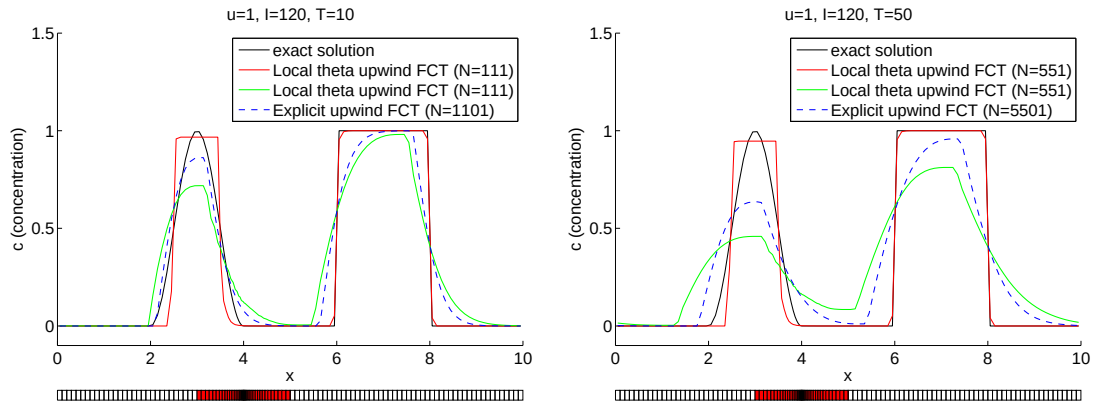
4. Compute the solution estimation at time t_n by approximating (6.11) as follows:

$$\frac{|\mathcal{V}_i^n|\bar{c}_i^n - |\mathcal{V}_i^n|\hat{\bar{c}}_i^n}{t_n - t_{n-1}} = -\sum_{j\in\mathcal{J}_i} |\mathcal{S}_{ij}^n| l_{ij}^n \Delta\phi_{ij}^n. \tag{6.12}$$

⌟

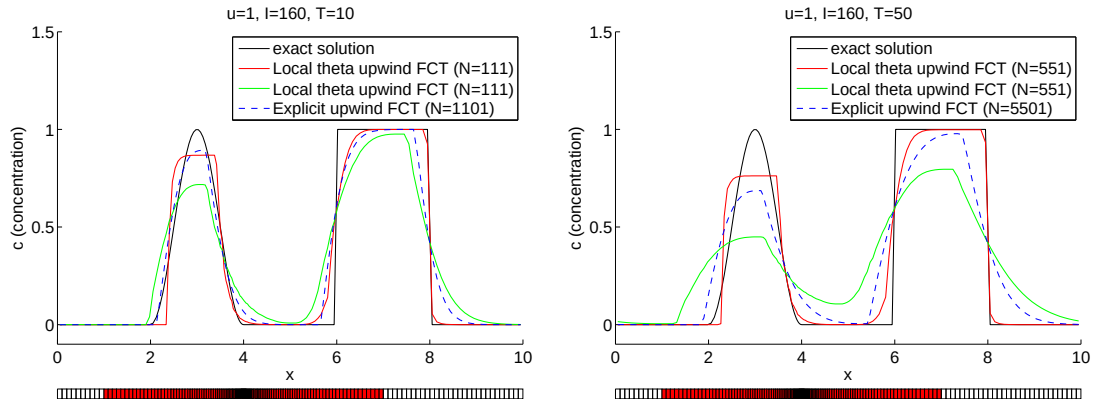
Now that the approximate local-theta FCT scheme has been formulated, several flux correction strategies can be tested. First of all, central flux correction is considered.

Method 6.9: This method is the local-theta upwind FCT scheme (Method 6.8) with central flux correction (4.3). ⌟



(a) small 'implicit' domain, 1 period

(b) small 'implicit' domain, 5 periods



(c) large 'implicit' domain, 1 period

(d) large 'implicit' domain, 5 periods

Figure 6.2: Local-theta upwind FCT scheme (Method 6.9 (red) and Method 6.10 (green)) compared to the local-theta upwind scheme ((6.9) and the explicit upwind FCT scheme (Method 4.8) with Lax-Wendroff flux correction (Method 4.11) for Test case 4.5

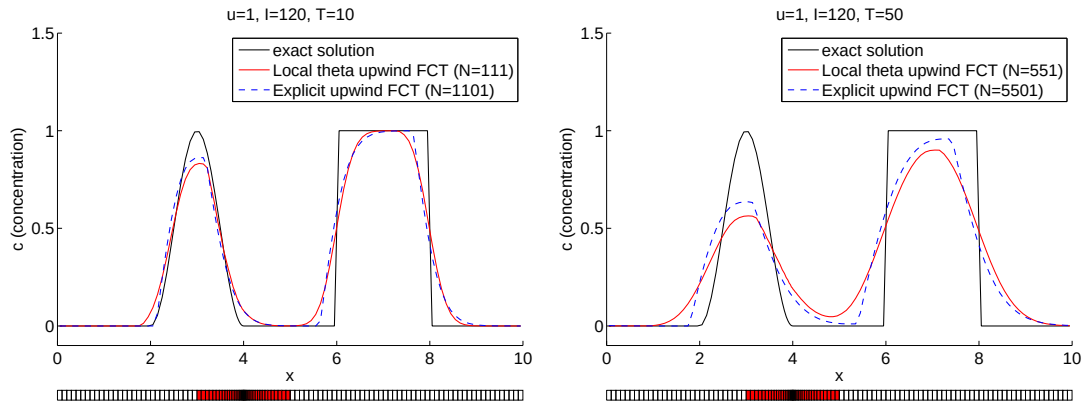
Method 6.9 (Figure 6.2, red) shows hardly any numerical diffusion. As a matter of fact, the numerical diffusion has been replaced by anti-diffusion. A similar result was obtained earlier in the explicit case (Figure 4.2). Proposition 5.6 reveals the amount of unphysical anti-diffusion that is introduced in the (larger) grid cells for which $\theta_{ij}^n \in [0, \frac{1}{2})$. The same proposition shows that this anti-diffusion can be eliminated by raising the local theta coefficients to a minimal value of $\frac{1}{2}$, which leads the following method.

Method 6.10: This method is the local-theta upwind FCT scheme (Method 6.8) with central flux correction (4.3), where all θ_{ij}^n has been chosen such that $\theta_{ij}^n \geq \frac{1}{2}$. ⌟

A negative side-effect of Method 6.10 (Figure 6.2, green) is that an extra amount of numerical diffusion is introduced by the local-theta upwind scheme (see also Proposition 5.2). In other words: Method 6.10 replaces the anti-diffusion error of Method 6.9 with a diffusion error.

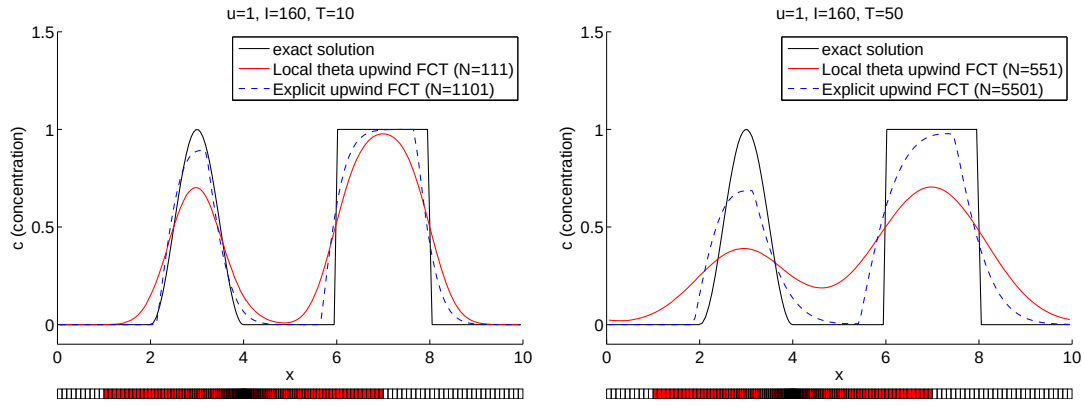
Alternatively, as in the explicit case, the anti-diffusion error can be avoided by switching from central flux correction to Lax-Wendroff correction, because the latter has zero numerical diffusion (see Proposition 5.7 for $\theta = 0$).

Method 6.11: This method is the local-theta upwind FCT scheme (Method 6.8) with Lax-Wendroff flux correction (Method 4.11) for explicit faces ($\theta_{ij}^n = 0$) and no flux correction for



(a) small 'implicit' domain, 1 period

(b) small 'implicit' domain, 5 periods



(c) large 'implicit' domain, 1 period

(d) large 'implicit' domain, 5 periods

Figure 6.3: Local-theta upwind FCT scheme (Method 6.11) compared to the local-theta upwind scheme ((6.9) and the explicit upwind FCT scheme (Method 4.8) with Lax-Wendroff flux correction (Method 4.11) for Test case 4.5

implicit faces ($\theta_{ij}^n > 0$). ⌟

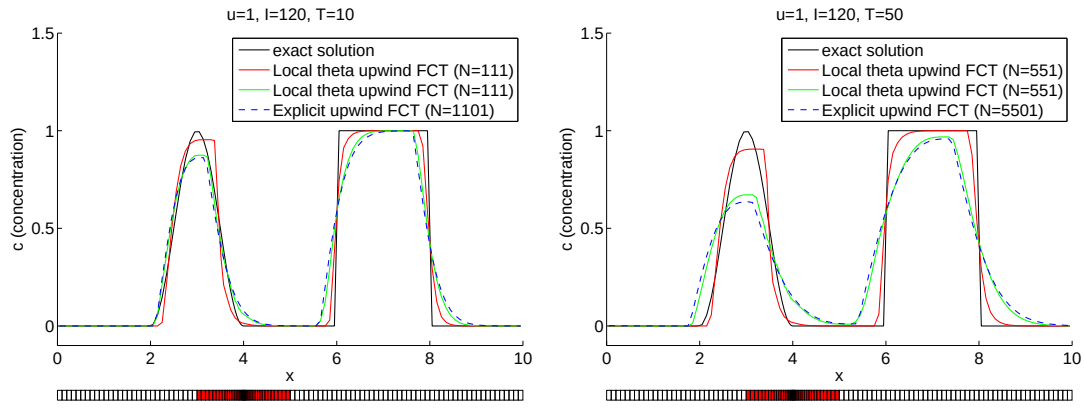
Figures 6.3(a) and 6.3(b) illustrate that, as long as the implicit part of the spatial domain is large, the accuracy of Method 6.11 (red) is comparable to the accuracy of the explicit FCT scheme (blue). Because flux correction is only applied in the explicit part of the spatial domain, the amount of numerical diffusion increases as the implicit domain becomes larger (Figures 6.3(c) and 6.3(d)).

By combining the advantages of Methods 6.9 and 6.11, the following method is obtained.

Method 6.12: This method is the local-theta upwind FCT scheme (Method 6.8) with Lax-Wendroff flux correction (Method 4.11) for explicit faces ($\theta_{ij}^n = 0$) and central flux correction (4.3) for implicit faces ($\theta_{ij}^n > 0$). ⌟

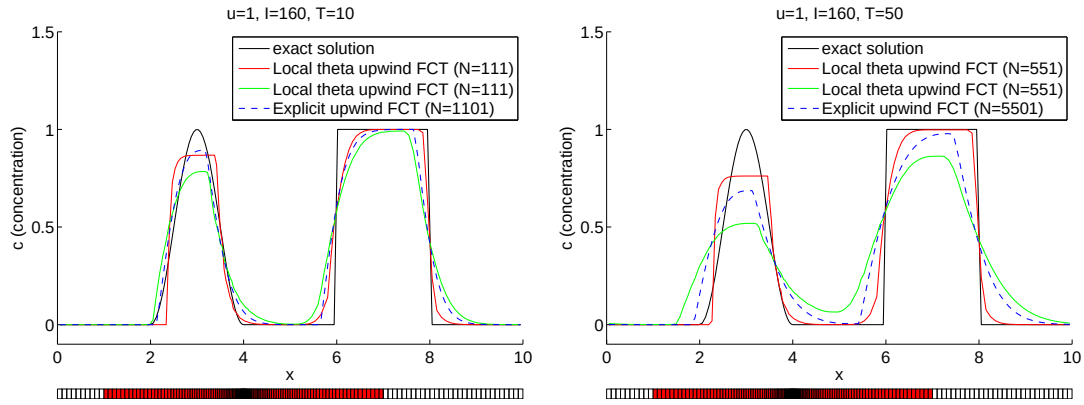
Similar to Method 6.9, Method 6.12 (Figure 6.4, red) introduces unphysical anti-diffusion, but only for the implicit grid cells for which $\theta_{ij}^n \in (0, \frac{1}{2})$. As a consequence, Method 6.12 shows less anti-diffusion than Method 6.9 (compare e.g. Figures 6.2(a) and 6.4(a)). Moreover, this explains why a larger implicit domain can lead to more anti-diffusion (compare e.g. Figures 6.4(a) and 6.4(c)). Again, it is possible to eliminate the anti-diffusion, by raising the local theta coefficients to a minimal value of $\frac{1}{2}$, which leads to the following method.

Method 6.13: This method is the local-theta upwind FCT scheme (Method 6.8) with Lax-



(a) small 'implicit' domain, 1 period

(b) small 'implicit' domain, 5 periods



(c) large 'implicit' domain, 1 period

(d) large 'implicit' domain, 5 periods

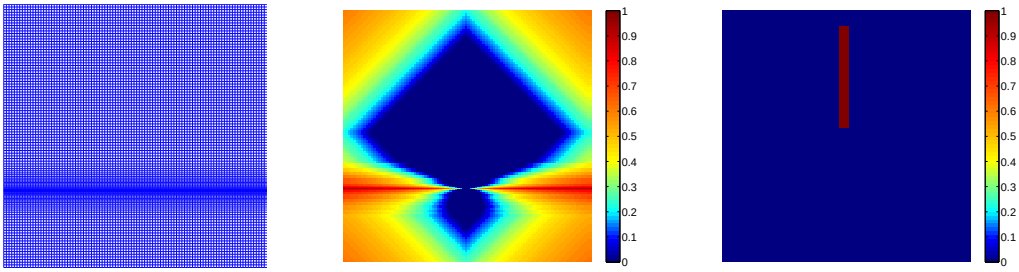
Figure 6.4: Local-theta upwind FCT scheme (Method 6.12 (red) and Method 6.13 (green)) compared to the local-theta upwind scheme ((6.9) and the explicit upwind FCT scheme (Method 4.8) with Lax-Wendroff flux correction (Method 4.11) for Test case 4.5

Wendroff flux correction (Method 4.11) for explicit faces ($\theta_{ij}^n = 0$) and central flux correction (4.3) flux correction for implicit faces ($\theta_{ij}^n > 0$), where θ_{ij}^n has been chosen such that either $\theta_{ij}^n = 0$ of $\theta_{ij}^n \geq \frac{1}{2}$. ⌟

Similar to Method 6.10, Method 6.13 (Figure 6.4, green) introduces an extra amount of numerical diffusion of the theta upwind scheme. Because this strategy is not applied in the explicit domain, this extra diffusion error is small in comparison with Method 6.10. Finally, note that, for this test case, the accuracy of Method 6.13 is higher than the accuracy of Method 6.11 (compare for example Figures 6.3(d) and 6.4(d)).

6.3 Molenkamp problem

Now, the local-theta FCT scheme will be tested for the Molenkamp problem. This testcase is the two-dimensional advection equation (Model 2.1 for $d = 0$, $p = 0$, and $m = 2$) with a constant angular velocity, which has been chosen such that the exact solution is periodic with a period of 4 hours. An illustration of the initial condition, the grid, and the local θ coefficients (for $\Delta t = 60\,\mathrm{s}$) can be found in Figure 6.5.



(a) grid

(b) Local θ coefficients ($\Delta t = 60\,\mathrm{s}$)

(c) initial condition

Figure 6.5: Molenkamp problem

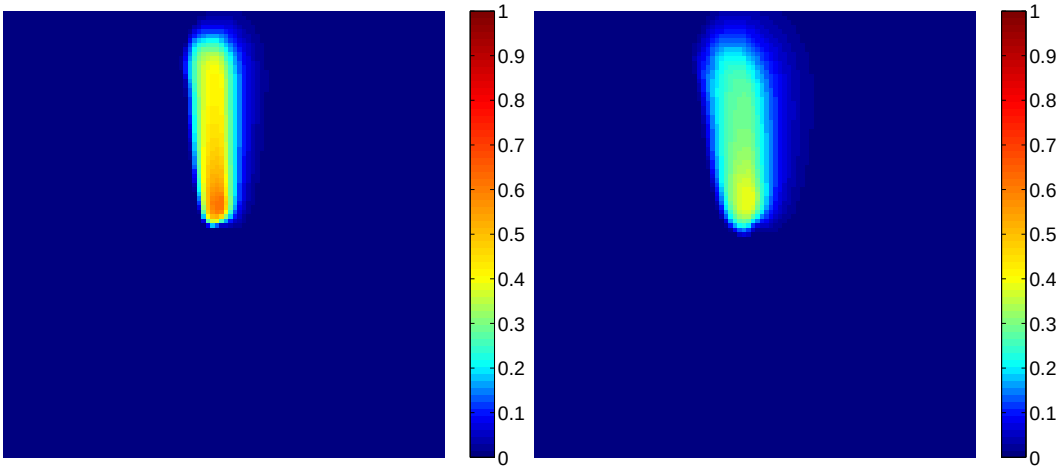
Figures 6.6 and 6.7 display the performance of several schemes for the Molenkamp test. As in the one-dimensional case (Figure 6.1), the local-theta scheme (Figure 6.6(c)) shows less numerical diffusion than scheme 16 of WAQ (Figure 6.6(b)), which is the theta upwind scheme with $\theta = 1$. Nevertheless, flux correction appears to be essential.

At the center of the spatial domain, the local-theta FCT schemes (Figures 6.7(a), 6.7(b), and 6.7(c)) all perform better than scheme 12 of WAQ (Figure 6.6(a)), which is an explicit FCT scheme. This can be explained as follows. In the explicit cells, the four schemes coincide, except that scheme 12 uses a smaller time step. As a consequence, scheme 12 needs to correct more numerical diffsusion to obtain the same accuracy (see also Proposition 5.2 for $\theta = 0$).

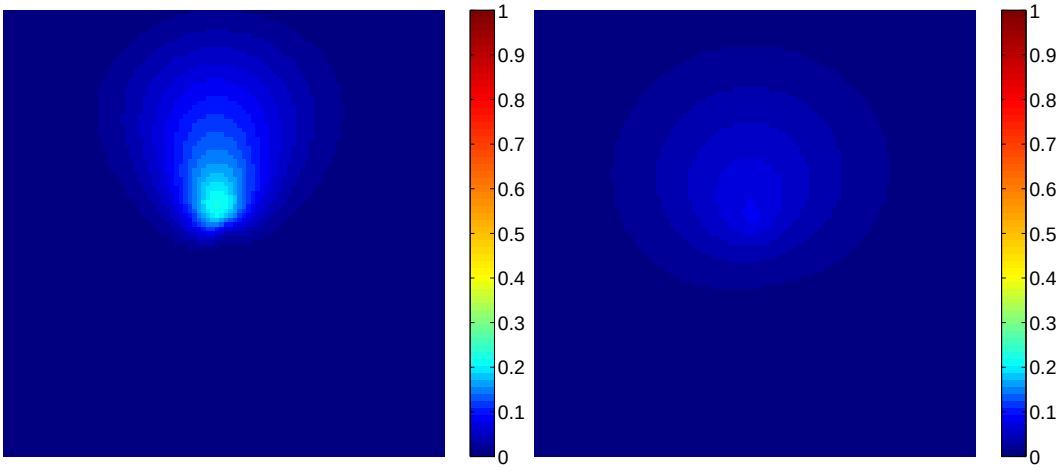
Near the boundary, where the number of explicit cells is small (see Figure 6.5(b)), the differences of the schemes become visible. Method 6.11 (Figure 6.7(a)) shows more numerical diffusion than scheme 12 near the boundary, which makes sense, as Method 6.11 only applies flux correction in the explicit part of the domain. Method 6.12 (Figure 6.7(b)) shows hardly any numerical diffusion. However, as mentioned before, it may introduce unphysical anti-diffusion, which can not be observed in this test case, because of the nature of the initial condition. Moreover, this method shows a dispersive error which appears to grow with θ. This error could be investigated theoretically by deriving a more accurate version of the modified equation that is considered in proposition 5.6. As observed before, Method 6.13 (Figure 6.7(c)) guarantees the absence of an anti-diffusion error, at the cost of extra numerical diffusion. Similar to Method 6.12, this method shows a dispersive error, which has been smeared out as a result of the extra numerical diffusion. Nonetheless, Method 6.13 is more accurate that Method 6.11 in the implicit area.

6.4 Revisiting Hong Kong

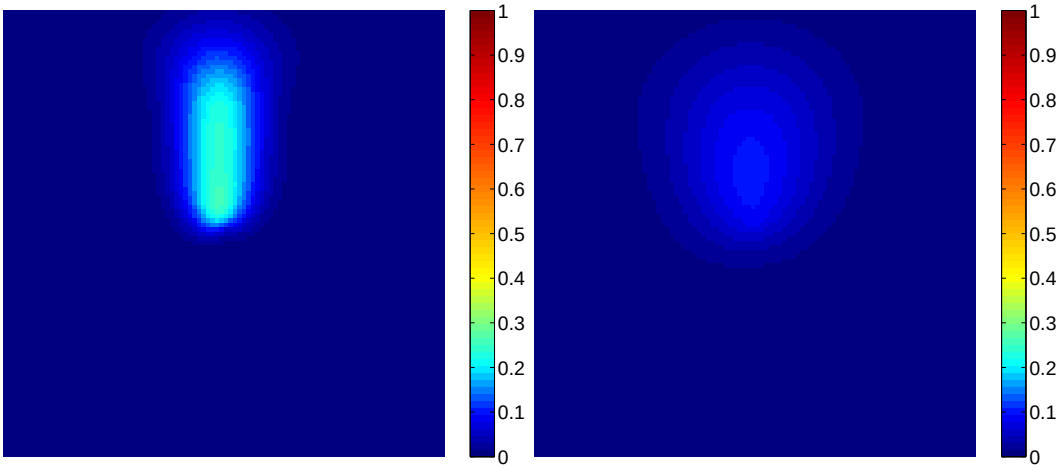
Now, it is time to revisit the Hong Kong model that illustrated the problem in the introduction of this thesis (Figure 1.2). Figures 6.8 and 6.9 show the performance of Methods 6.12 and 6.13 for this problem, compared to schemes 12 and 16 of WAQ. First of all, note the difference in computational time between scheme 12 (176 minutes) and the local-theta upwind schemes (14 minutes). Figure 6.10 illustrates that the local theta coefficients vary strongly in space and time during the simulation. Even though the implicit part of the spatial domain appears to be large, the accuracy of the local theta FCT schemes is comparable to the accuracy of scheme 12. Furthermore, note that the difference between Methods 6.12 and 6.13 is hardly visible.



(a) Scheme 12 of WAQ (see Appendix A) ($\Delta t = 6\,\mathrm{s}$)

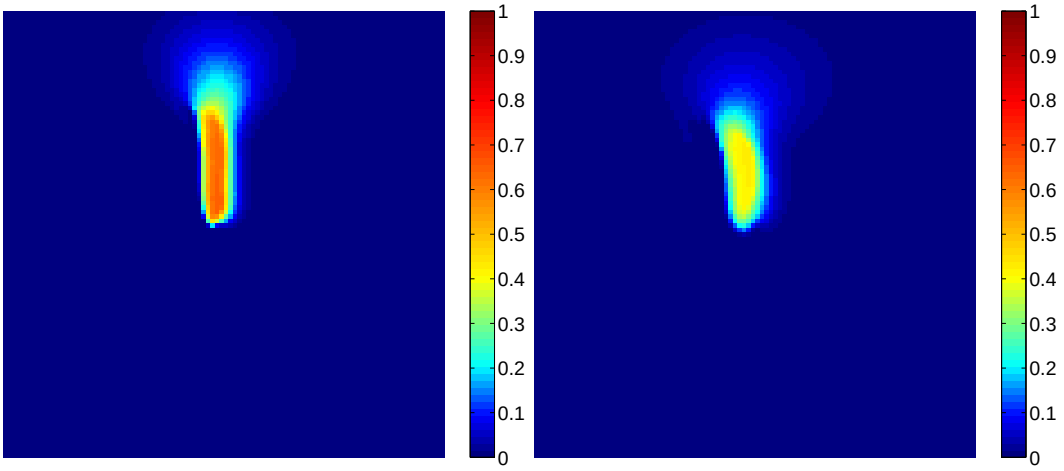


(b) Scheme 16 of WAQ (see Appendix A) ($\Delta t = 60\,\mathrm{s}$)

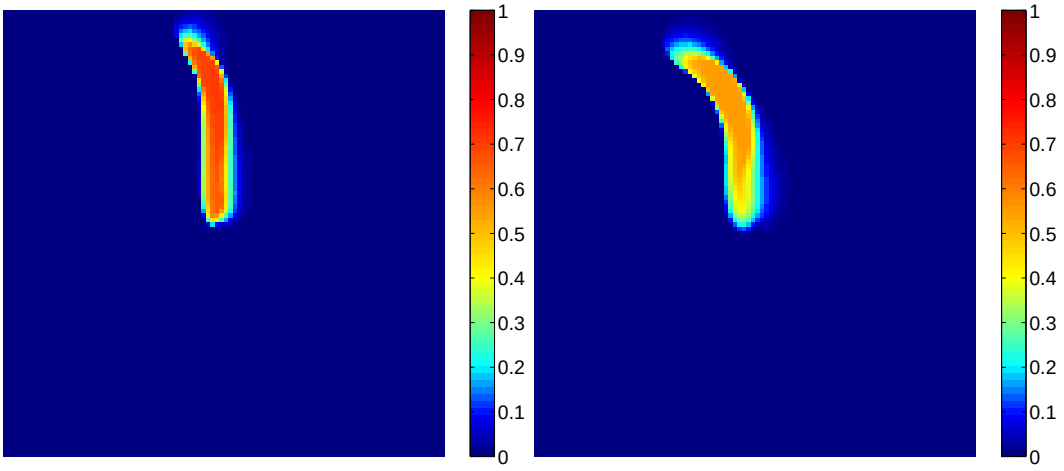


(c) Local-theta upwind scheme (Method 6.1) ($\Delta t = 60\,\mathrm{s}$)

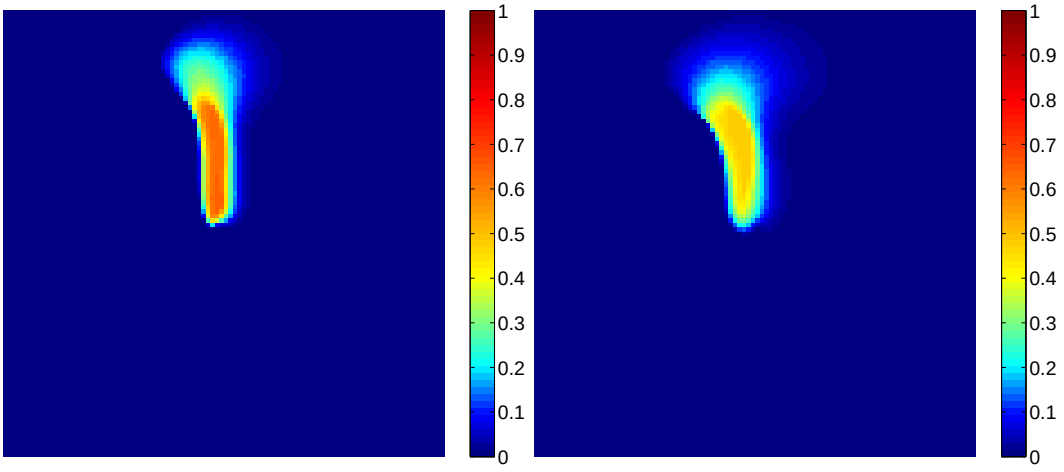
Figure 6.6: Molenkamp problem after one (left) and four (right) rotations



(a) Local-theta upwind FCT scheme (Method 6.11) ($\Delta t = 60\,\mathrm{s}$)



(b) Local-theta upwind FCT scheme (Method 6.12) ($\Delta t = 60\,\mathrm{s}$)



(c) Local-theta upwind FCT scheme (Method 6.13) ($\Delta t = 60\,\mathrm{s}$)

Figure 6.7: Molenkamp problem after one (left) and four (right) rotations

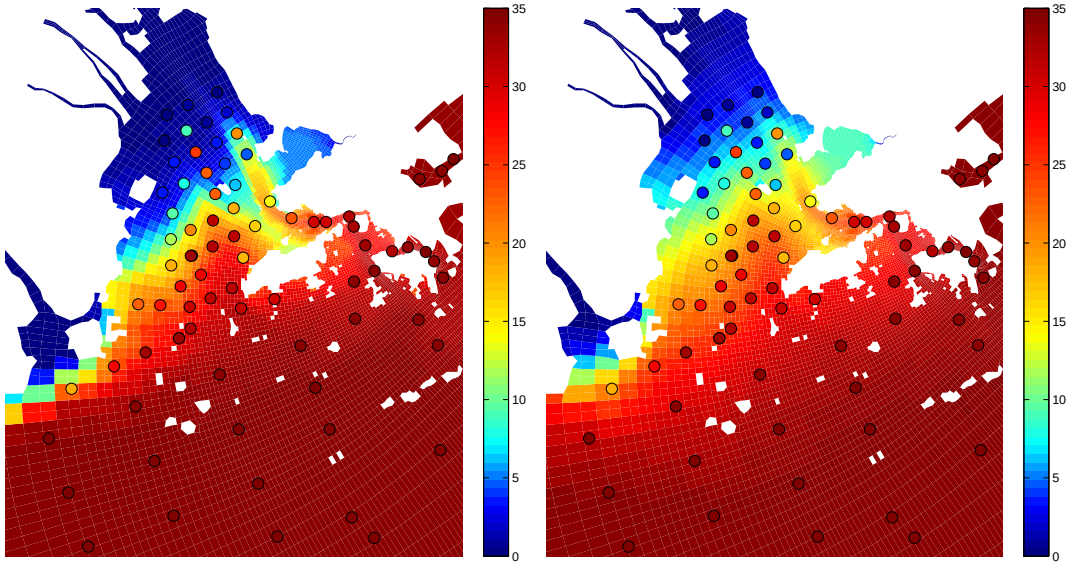
6.5 Final implementation in WAQ

Several test cases have illustrated that the local-theta FCT schemes 6.12 and 6.13 are not only robust, but also accurate. Therefore, these schemes have both been added to the current schemes of WAQ.

What has not been put forward so far, is the fact that both schemes 12 and 16 use central discretisation in the vertical direction (see also Appendix A), which may cause oscillatory or negative results. For this reason, a Forrester filter has been implemented in WAQ, which is optional (this option was used during the testcase). The Forrester filter renders the solution monotone in the vertical direction, eliminating buth spurious and non-spurious oscillations. Moreover, in the current implementation, positivity of the solution is not guaranteed, with and without the Forrester filter. These two relevant drawbacks of schemes 12 and 16 do not apply for the local-theta schemes. Alternative versions of schemes 12 and 16, that do not suffer from the aforementioned disadvantages, are schemes 14 and 15, which use diffusive implicit upwind discretisation in the vertical direction (see also Appendix A). The latter schemes have a much lower level of accuracy than the local-theta FCT schemes.

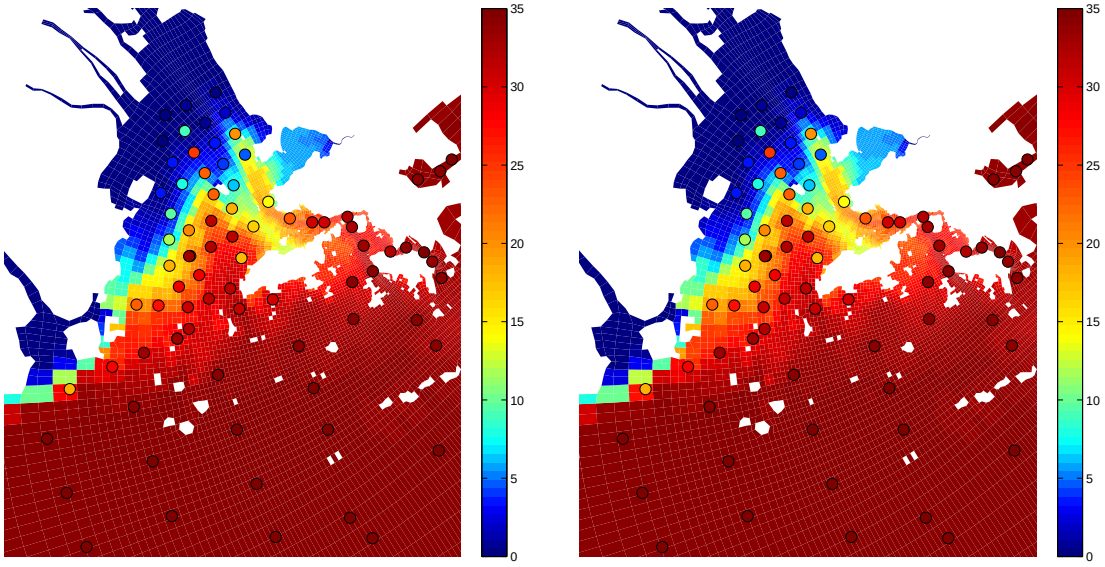
6.6 Summary

The theta scheme can be generalised to the local-theta scheme, which uses an optimal local θ rather than a constant value. The coefficients are called optimal, if they are as small as possible, to minimise the amount of numerical diffusion, yet large enough to ensure that the scheme is stable, positivity preserving and non-oscillatory. This way, the accuracy of the theta upwind scheme is improved, without loss of robustness. Once more, the flux corrected transport algorithm can be used to enhance the accuracy even more. If Lax-Wendroff flux correction is applied in the explicit part of the spatial domain, i.e. the faces where the local-theta coefficients are zero, the scheme can compete with the explicit FCT scheme as long as the explicit domain is sufficiently large. To obtain good accuracy for larger time steps as well, central flux correction can be applied in the implicit part of the domain in addition. During a Molenkamp test, this strategy led to a replacement of the diffusion error with a dispersion error, that grew with θ. Another flaw of this approach is that unphysical anti-diffusion may be introduced, which can be avoided by raising the local theta coefficients to a minimum of $\frac{1}{2}$ in the implicit part of the domain. A negative side effect of this remedy is an increase of the numerical diffusion of the local-theta upwind scheme. However, the overall accuracy is acceptable. Altogether, the local-theta FCT schemes 6.12 and 6.13 are both accurate and robust and have both been implemented in the source code of Delft3D-WAQ.



(a) Scheme 12 of WAQ (see Appendix A) (Δt = 1 min. , cpu time $\approx$ 176 min.)

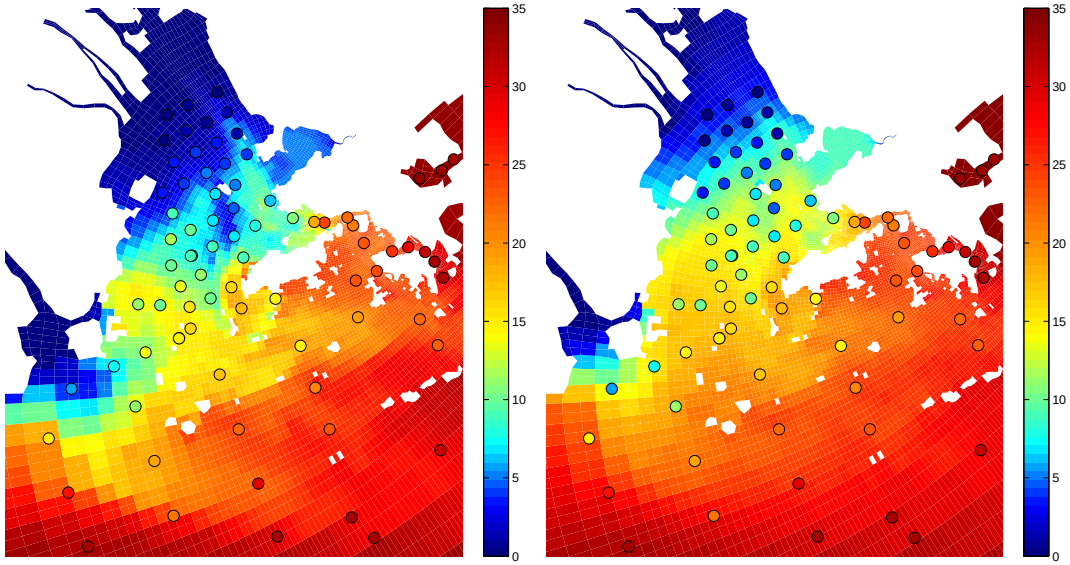
(b) Scheme 16 of WAQ (see Appendix A) (Δt = 60 min. , cpu time $\approx$ 9 min.)



(c) Local-theta upwind FCT scheme (Method 6.12) ($\Delta t = 60$ min. , cpu time $\approx$ 14 min.)

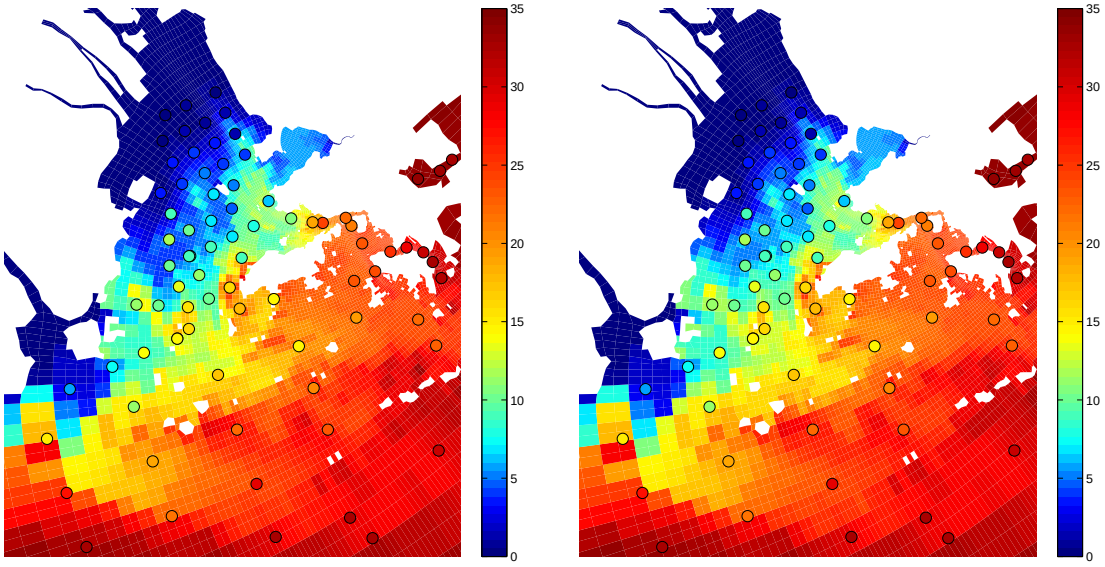
(d) Local-theta upwind FCT scheme (Method 6.13) ($\Delta t = 60$ min. , cpu time $\approx$ 14 min.)

Figure 6.8: Simulation of salinity (after $\pm\frac{1}{2}$ year) at the bottom of an estuary near Hong Kong by means of Delft3D-WAQ. The colors of the cicles indicate measured values.



(a) Scheme 12 of WAQ (see Appendix A) (Δt = 1 min. , cpu time $\approx$ 176 min.)

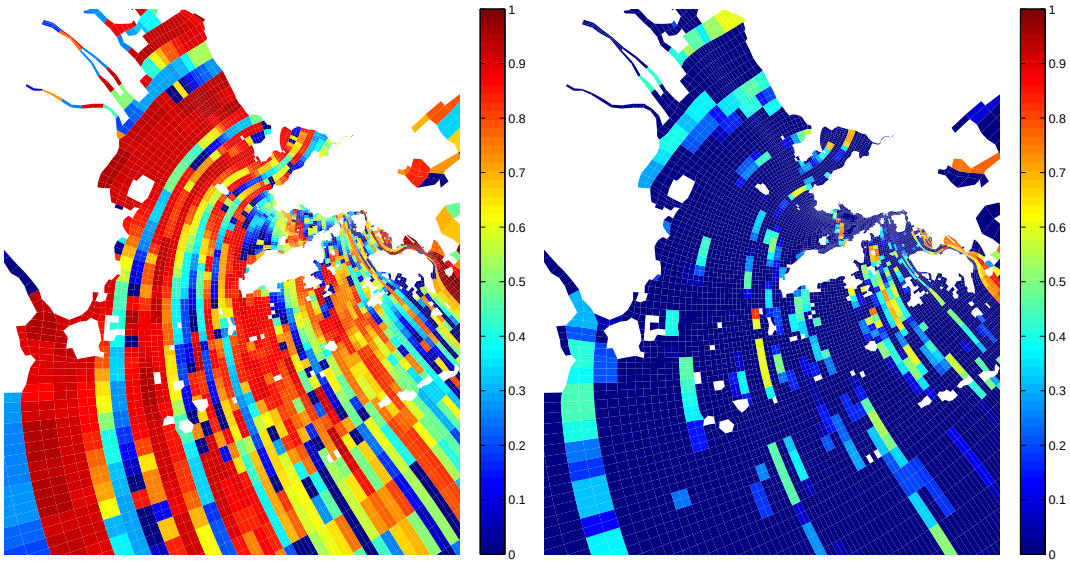
(b) Scheme 16 of WAQ (see Appendix A) (Δt = 60 min. , cpu time $\approx$ 9 min.)



(c) Local-theta upwind FCT scheme (Method 6.12) ($\Delta t = 60$ min. , cpu time $\approx$ 14 min.)

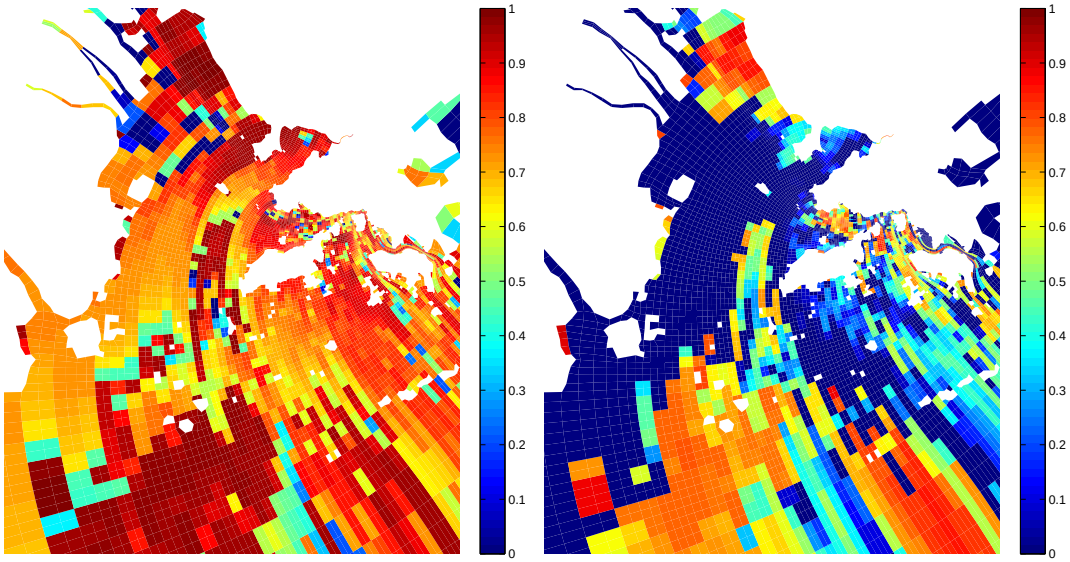
(d) Local-theta upwind FCT scheme (Method 6.13) ($\Delta t = 60$ min. , cpu time $\approx$ 14 min.)

Figure 6.9: Simulation of salinity (after $\pm\frac{1}{2}$ year) at the surface of an estuary near Hong Kong by means of Delft3D-WAQ. The colors of the cicles indicate measured values.



(a) Bottom layer, at the beginnnig

(b) Bottom layer, at the end (after $\frac{1}{2}$ year)



(c) Surface layer, at the beginning

(d) Surface layer, at the end (after $\frac{1}{2}$ year)

Figure 6.10: Local theta coefficients for the Hong model for $\Delta t = 60\,\text{min}$.

Chapter 7

Summary & Recommendations

Water quality is determined by the concentrations of the substances it contains. These concentrations be affected by transport and water quality processes. The corresponding mathematical model is the advection diffusion reaction equation, one for each substance that needs to be simulated.

The solution of the water quality model can be approximated by means of the finite volume method (FVM). The grid that is used by Delft3D-WAQ is usually three-dimensional, unstructured, and strongly non-uniform. Water quality processes are treated in an explicit manner, in order to avoid the necessity of solving a nonlinear system. The quality of a finite volume scheme is determined by accuracy and robustness. In this respect, the local and the global discrete maximum principle are favorable properties of a FVM because they imply stability, positivity and non-oscillatory behavior.

The explicit upwind scheme is neither robust nor accurate. The former is caused by the fact that the time step is limited to ensure stability, positivity and non-oscillatory behavior. The inaccuracy is a a result of numerical diffusion, which can be diminished by means of the flux corrected transport (FCT) algorithm. Central flux correction leads to an anti-diffusion error. For this reason, the Lax-Wendroff scheme is more suitable to serve as flux corrector, as its numerical diffusion is equal to zero. Although the accuracy of the explicit FCT scheme is generally high, the robustness of the explicit scheme remains unchanged, which can result in long computational times. Scheme 12 of WAQ is an explicit FCT scheme with Lax-Wendroff flux correction.

The theta upwind scheme is robust, but inaccurate due to numerical diffusion, which grows with θ. The robustness results from the fact that the scheme is stable, positivity preserving and non-oscillatory, provided that θ is sufficiently large. As in the explicit case, it is possible to attempt to improve the accuracy by applying the flux corrected transport algorithm. For this purpose, the explicit FCT scheme has been generalised to the theta FCT scheme. Implicit nonlinear systems can be avoided by approximating the flux corrections at the new time by means of the first order solution estimation. Because Lax-Wendroff fluxes are not suitable as flux correctors if θ is not close to zero, and central fluxes are unsuitable if θ is not close to $\frac{1}{2}$, the theta Lax-Wendroff scheme has been considered. Unfortunately, the approximated theta FCT scheme did not lead to satisfactory accuracy with theta Lax-Wendroff flux correction. The main reason is that, as θ becomes larger, the theta upwind scheme needs more flux correction, as it suffers more from numerical diffusion, whereas the flux correction becomes less accurate, because the flux corrections at the new time are approximations.

The theta scheme can be generalised to the local-theta scheme, which uses an optimal local θ rather than a constant value. The coefficients are called optimal, if they are as small as possible, to minimise the amount of numerical diffusion, yet large enough to ensure that the scheme is stable, positivity preserving and non-oscillatory. This way, the accuracy of the theta upwind

scheme is improved, without loss of robustness. Once more, the flux corrected transport algorithm can be used to enhance the accuracy even more. If Lax-Wendroff flux correction is applied in the explicit part of the spatial domain, i.e. the faces where the local-theta coefficients are zero, the scheme can compete with the explicit FCT scheme as long as the explicit domain is sufficiently large. To obtain good accuracy for larger time steps as well, central flux correction can be applied in the implicit part of the domain in addition. During a Molenkamp test, this strategy led to a replacement of the diffusion error with a dispersion error, that grew with θ. Another flaw of this approach is that unphysical anti-diffusion may be introduced, which can be avoided by raising the local theta coefficients to a minimum of $\frac{1}{2}$ in the implicit part of the domain. A negative side effect of this remedy is an increase of the numerical diffusion of the local-theta upwind scheme. However, the overall accuracy is acceptable. Altogether, the local-theta FCT schemes 6.12 and 6.13 are both accurate and robust and have both been implemented in the source code of Delft3D-WAQ.

To answer the question that was asked in the introduction of this thesis (Chapter 1):

> *It is possible to construct a finite volume scheme for the advection diffusion equation that is both accurate and robust by constructing an optimal local blend of the accurate explicit FCT scheme and the robust theta method.*

7.1 Recommendations

Apart from flux correction, the accuracy of the local theta scheme can be improved in two ways. Firstly, the first order upwind flux could be replaced by, for instance, the upwinded central flux (4.4). Regardless of the time discretisation, this probably leads to a decrease in numerical diffusion. Moreover, the local theta coefficients could altered. Two suggestions can be found in Remark 6.5. The flux correction strategy can be improved in three ways. First of all, an alternative higher order flux function could lead to better flux correction. Figure 6.7(b) calls for an investigation of the dispersion error of the theta scheme with central discretisation. This could be executed by deriving of a more accurate version of the modified equation. Furthermore, a different limiter, such as a TVD limiter [KT, Sections 2 and 10], could lead to an improvement. Perhaps it is possible to eliminate the anti-diffusion error of the central scheme by changing the limiter, rather than by raising the local theta coefficients at the cost of extra numerical diffusion? Finally, an iterative local-theta FCT scheme could be formulated similar to the iterative FEM-FCT scheme that was proposed by Kuzmin et al. in [KMT, Sections 5 and 6]. This probably leads to higher accuracy, although extra computational costs also need to be taken into account.

Part II

Solving the numerical model

Chapter 8

Solution methods for linear systems

In the previous part, the finite volume method was discussed to solve the water quality model (Model 2.1). Since it is often needed to predict water quality several years ahead, large time steps are desirable. Therefore, implicit methods are preferable to explicit schemes, as the latter do not allow arbitrary time steps without becoming unstable. Implicit methods require the solution of many large sparse[1] linear systems. To obtain these solutions, efficient solvers are discussed in this chapter.

8.1 Direct methods

A direct method computes a theoretically exact solution using a finite number of operations. A useful measure for the amount of work is the number of floating point operations (*flops*).

Example 8.1: Consider a full matrix $\underline{\underline{A}} \in \mathbb{R}^{n \times n}$ and a vector $\underline{\mathbf{x}} \in \mathbb{R}^n$. Then the matrix vector product $\underline{\underline{A}}\,\underline{\mathbf{x}}$ needs $2n^2$ flops. ⌟

8.1.1 Triangular matrices

Triangular systems are easily solved by means of backward or forward substitution.

Method 8.2 (Forward substitution): Let $\underline{\underline{L}} \in \mathbb{R}^{n \times n}$ be a lower triangular matrix. *Forward substitution* solves the linear system $\underline{\underline{L}}\,\underline{\mathbf{x}} = \underline{\mathbf{b}}$ by means of the following algorithm:

1. for $i = 1, ..., n$:
2. $\quad x_i = b_i$
3. $\quad$ for $j = 1, ..., i-1$:
4. $\quad\quad x_i = x_i - l_{ij} x_j$
5. $\quad$ end
6. end

For a full matrix, the computational costs amount to n^2 flops. In case the matrix has at most q nonzero off-diagonal elements per row, approximately $2qn$ flops are needed. ⌟

Method 8.3 (Backward substitution): Let $\underline{\underline{U}} \in \mathbb{R}^{n \times n}$ be an upper triangular matrix. *Backward substitution* solves the linear system $\underline{\underline{U}}\,\underline{\mathbf{x}} = \underline{\mathbf{b}}$ by means of the following algorithm:

1. for $i = n, ..., 1$:
2. $\quad x_i = b_i$

[1] A matrix is sparse if it contains 'many' zero elements

3. for $j = i+1, ..., n$:
4. $x_i = x_i - u_{ij} x_j$
5. end
6. $x_i = \frac{x_i}{u_{ii}}$
7. end

For a full matrix, the computational costs amount to n^2 flops. In case the matrix has at most q nonzero off-diagonal elements per row, approximately $2qn$ flops are needed. ⌟

8.1.2 General square matrices

A general square system can be solved by means of Gaussian elimination. This method reduces a linear system to two triangular systems, which can be solved by backward or forward substitution. The triangular systems are obtained by constructing an LU factorisation of the matrix.

Definition 8.4 (LU factorisation): An *LU factorisation* of a matrix $\underline{\underline{A}}$ consists of a unit lower triangular matrix $\underline{\underline{L}}$ and an upper triangular matrix $\underline{\underline{U}}$ such that

$$\underline{\underline{A}} \quad = \quad \underline{\underline{L}}\,\underline{\underline{U}}.$$ ⌟

Method 8.5 (LU factorisation): The following algorithm generates an LU factorisation for a matrix $\underline{\underline{A}} \in \mathbb{R}^{n \times n}$, provided that the pivots, u_{kk}, are nonzero.

1. for $i = 1, ..., n$:
2. $w = a_{i*}$
3. for $k = 1, ..., i-1$:
4. $w_k = \frac{w_k}{u_{kk}}$
5. $w = w - w_k u_{k*}$
6. end
7. $l_{ij} = w_j$ for $j = 1, ..., i-1$
8. $u_{ij} = w_j$ for $j = i, ..., n$
9. end

a_{i*} denotes row i of $\underline{\underline{A}}$. For a full matrix, the computational costs of the factorisation amount to $\frac{2}{3}n^3$ flops. ⌟

More detailed information about LU factorizations can be found in [GL96, Section 3.2].

Method 8.6 (Gaussian elimination): *Gaussian elimination* solves an $n \times n$ linear system $\underline{\underline{A}}\,\underline{\mathbf{x}} = \underline{\mathbf{b}}$ in three steps:

1. Construct an LU factorisation of $\underline{\underline{A}}$. This can by done by means of Method 8.5.
2. Solve $\underline{\mathbf{y}}$ from $\underline{\underline{L}}\,\underline{\mathbf{y}} = \underline{\mathbf{b}}$ by means of forward substitution (Method 8.2)
3. Solve $\underline{\mathbf{x}}$ from $\underline{\underline{U}}\,\underline{\mathbf{x}} = \underline{\mathbf{y}}$ by applying backward substitution (Method 8.3) ⌟

A disadvantage of Gaussian elimination for sparse matrices is that $\underline{\underline{L}}$ and $\underline{\underline{U}}$ are generally less sparse than $\underline{\underline{A}}$. This effect is called fill-in.

Definition 8.7 (Fill-in): The fill-in of a matrix consists of those entries which change from an initial zero to a nonzero value during the execution of an algorithm. ⌟

Theorem 8.8: *Consider a matrix $\underline{\underline{A}}$ with lower bandwidth q_l and upper bandwidth q_u. Let $\underline{\underline{A}} = \underline{\underline{L}}\,\underline{\underline{U}}$ be an LU factorisation. Then, $\underline{\underline{L}}$ has lower bandwidth q_l and $\underline{\underline{U}}$ has upper bandwidth q_u. Moreover, if $n \gg q_l, q_u$, the costs of the LU factorisation are approximately $2nq_lq_u$ flops.*

Proof. See [GL96, Theorem 4.3.1 and p. 153]. ■

8.2 Iterative Methods

An alternative for Gaussian elimination is provided by iterative methods. These methods iteratively improve an initial solution estimation.

Method 8.9 (**Iterative method: general form**)**:** Consider an $n \times n$ linear system $\underline{\underline{A}}\,\underline{\mathbf{x}} = \underline{\mathbf{b}}$. An *iterative method* consists of the following steps:

1. Choose an initial guess of the solution, $\underline{\mathbf{x}}_0 \in \mathbb{R}^n$.
2. Set $k = 1$. Choose an improved approximation of the solution, $\underline{\mathbf{x}}_k$, and set $k = k + 1$, until a certain termination criterion is met.
3. Approximate the solution of $\underline{\underline{A}}\,\underline{\mathbf{x}} = \underline{\mathbf{b}}$ according to $\underline{\mathbf{x}} \approx \underline{\mathbf{x}}_k$. ⌟

Definition 8.10 (**Convergent iterative method**)**:** Method 8.9 is *convergent* if

$$\|\underline{\mathbf{x}}_k - \underline{\mathbf{x}}\| \to 0.$$ ⌟

Remark 8.11 (A good termination criterion)**:** A good termination criterion has the following properties:

1. It is scaling invariant. This means that the number of iterations for $\alpha\,\underline{\underline{A}}\,\underline{\mathbf{x}} = \alpha\underline{\mathbf{b}}$ is independent of $\alpha \in \mathbb{R}$.
2. The number of iterations should not be independent of the initial estimation $\underline{\mathbf{x}}_0$; a better initial guess should lead to a smaller number of iterations.
3. It provides an upper bound for the relative error $\frac{\|\underline{\mathbf{x}} - \underline{\mathbf{x}}_k\|_2}{\|\underline{\mathbf{x}}\|_2}$.

An example of a good termination criterion, satisfying the properties above, is

$$\frac{\|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k\|_2}{\|\underline{\mathbf{b}}\|_2} \quad \leq \quad \epsilon.$$ ⌟

8.2.1 Linear fixed point iteration

Linear fixed point iteration determines the estimates of the solution by means of a matrix splitting of $\underline{\underline{A}}$.

Definition 8.12 (**Matrix splitting**)**:** A *matrix splitting* of a matrix $\underline{\underline{A}}$ consists of matrices $\underline{\underline{M}}$ and $\underline{\underline{N}}$ such that

$$\underline{\underline{A}} \quad = \quad \underline{\underline{M}} - \underline{\underline{N}}.$$ ⌟

Method 8.13 (**Linear fixed point iteration**)**:** *Linear fixed point iteration* follows from Method 8.9 by updating $\underline{\mathbf{x}}_k$ according to a matrix splitting $\underline{\underline{A}} = \underline{\underline{M}} - \underline{\underline{N}}$, with $\underline{\underline{M}}$ nonsingular. The following strategies are equivalent:

$$\underline{\underline{M}}\,\underline{\mathbf{x}}_k = \underline{\underline{N}}\,\underline{\mathbf{x}}_{k-1} + \underline{\mathbf{b}}, \tag{8.1}$$

$$\underline{\mathbf{x}}_k = \underline{\underline{M}}^{-1}\,\underline{\underline{N}}\,\underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}^{-1}\underline{\mathbf{b}}, \tag{8.2}$$

$$\underline{\mathbf{x}}_k = \underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}^{-1}(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}). \tag{8.3}$$

⌟

Example 8.14: Consider a matrix A. Define a diagonal matrix $\underline{\underline{D}}$, a strictly lower triangular matrix $\underline{\underline{L}}$, and a strictly upper triangular matrix $\underline{\underline{U}}$, such that $\underline{\underline{A}} = \underline{\underline{L}} + \underline{\underline{D}} + \underline{\underline{U}}$. Moreover, let $\omega \in \mathbb{R}$ be a constant. The following matrix splittings $\underline{\underline{A}} = \underline{\underline{M}} - \underline{\underline{N}}$ lead to well-known methods:

$\underline{\underline{M}}$	$\underline{\underline{N}}$	Method
$\underline{\underline{D}}$	$-\underline{\underline{L}}-\underline{\underline{U}}$	Gauss-Jacobi (GJ)
$\underline{\underline{D}}+\underline{\underline{L}}$	$-\underline{\underline{U}}$	Gauss-Seidel (GS)
$\underline{\underline{D}}+\underline{\underline{U}}$	$-\underline{\underline{L}}$	Backward Gauss-Seidel
$\underline{\underline{D}}+\omega\underline{\underline{L}}$	$(\omega-1)\underline{\underline{L}}-\underline{\underline{U}}$	Successive Over-Relaxation (SOR)

⌟

Theorem 8.15 (Convergence of linear fixed point iteration)**:** *Method 8.13 converges for any starting vector* $\underline{\mathbf{x}}_0$*, if*

$$\max\{|\lambda| : \lambda \text{ eigenvalue of } \underline{\underline{M}}^{-1}\underline{\underline{N}}\} < 1.$$

Proof. See [GL96, Theorem 10.1.1]. ■

Method 8.16 (**Multiple linear fixed point iteration**)**:** Method 8.13 can be applied twice to obtain a new iterative scheme:

1. Use Method 8.13 with a matrix splitting $\underline{\underline{A}} = \underline{\underline{M}}_1 - \underline{\underline{N}}_1$ to obtain $\underline{\mathbf{x}}_k^*$:
$$\underline{\underline{M}}_1\underline{\mathbf{x}}_k^* = \underline{\underline{N}}_1\underline{\mathbf{x}}_{k-1} + \underline{\mathbf{b}};$$
2. Apply Method 8.13 once more using another matrix splitting $\underline{\underline{A}} = \underline{\underline{M}}_2 - \underline{\underline{N}}_2$ to acquire $\underline{\mathbf{x}}_k$:
$$\underline{\underline{M}}_2\underline{\mathbf{x}}_k = \underline{\underline{N}}_2\underline{\mathbf{x}}_k^* + \underline{\mathbf{b}}.$$

⌟

Proposition 8.17: *Method 8.16 is equivalent to Method 8.13 for the matrix splitting* $\underline{\underline{A}} = \underline{\underline{M}} - (\underline{\underline{M}} - \underline{\underline{A}})$*, with*

$$\underline{\underline{M}} = \underline{\underline{M}}_1\left(\underline{\underline{M}}_1 + \underline{\underline{N}}_2\right)^{-1}\underline{\underline{M}}_2.$$

Proof. Since (8.1) and (8.3) are equivalent, Method 8.18 can be rewritten to obtain:

$$\begin{aligned}\underline{\mathbf{x}}_k^* &:= \underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}_1^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ \underline{\mathbf{x}}_k &= \underline{\mathbf{x}}_k^* + \underline{\underline{M}}_2^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k^*\right).\end{aligned}$$

Thus,

$$\begin{aligned}\underline{\mathbf{x}}_k &= \underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}_1^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ &\quad + \underline{\underline{M}}_2^{-1}\Big(\underline{\mathbf{b}} - \underline{\underline{A}}\left(\underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}_1^{-1}(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1})\right)\Big)\\ &= \underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}_1^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ &\quad + \underline{\underline{M}}_2^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1} - \underline{\underline{A}}\,\underline{\underline{M}}_1^{-1}(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1})\right)\\ &= \underline{\mathbf{x}}_{k-1} + \left(\underline{\underline{M}}_1^{-1} + \underline{\underline{M}}_2^{-1}(\underline{\underline{I}} - \underline{\underline{A}}\,\underline{\underline{M}}_1^{-1})\right)\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ &= \underline{\mathbf{x}}_{k-1} + \left(\underline{\underline{M}}_1^{-1} + \underline{\underline{M}}_2^{-1}\left(\underline{\underline{I}} - (\underline{\underline{M}}_2 - \underline{\underline{N}}_2)\underline{\underline{M}}_1^{-1}\right)\right)\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ &= \underline{\mathbf{x}}_{k-1} + \left(\underline{\underline{I}} + \underline{\underline{M}}_2^{-1}\left(\underline{\underline{M}}_1 - (\underline{\underline{M}}_2 - \underline{\underline{N}}_2)\right)\right)\underline{\underline{M}}_1^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ &= \underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}_2^{-1}(\underline{\underline{M}}_1 + \underline{\underline{N}}_2)\underline{\underline{M}}_1^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right)\\ &= \underline{\mathbf{x}}_{k-1} + \underbrace{\left(\underline{\underline{M}}_1(\underline{\underline{M}}_1 + \underline{\underline{N}}_2)^{-1}\underline{\underline{M}}_2\right)}_{=:\underline{\underline{M}}}{}^{-1}\left(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1}\right).\end{aligned}$$

Applying the equivalence of (8.1) and (8.3) once more completes the proof. ■

An example of Method 8.16 is Symmetric Gauss-Seidel.

Method 8.18 (Symmetric Gauss-Seidel): *Symmetric Gauss-Seidel* follows from method 8.16 by applying one step of Gauss-Seidel ($\underline{\underline{M}}_1 = \underline{\underline{D}} + \underline{\underline{L}}$, $\underline{\underline{N}}_1 = -\underline{\underline{U}}$), followed by one step of Backward Gauss-Seidel ($\underline{\underline{M}}_2 = \underline{\underline{D}} + \underline{\underline{U}}$, $\underline{\underline{N}}_2 = -\underline{\underline{L}}$). ⌟

8.2.2 Krylov methods

A large category of iterative schemes is formed by the Krylov methods. These are based on a so-called Krylov space, which is defined below.

Definition 8.19 (Krylov space): Let $\underline{\underline{A}} \in \mathbb{R}^{n\times n}$ and $\underline{\mathbf{r}} \in \mathbb{R}^n$. A *Krylov space* is of the form:

$$\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}) \quad = \quad \text{span}\left\{\underline{\mathbf{r}}, \underline{\underline{A}}\,\underline{\mathbf{r}}, ..., \underline{\underline{A}}^{k-1}\underline{\mathbf{r}}\right\}. \qquad ⌟$$

The link between Krylov methods and linear fixed point iteration becomes clear from the following proposition.

Proposition 8.20: *Let $\underline{\mathbf{x}}_k$ result from linear fixed point iteration (Method 8.13):*

$$\underline{\mathbf{x}}_k \quad = \quad \underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}^{-1}\underbrace{(\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_{k-1})}_{\underline{\mathbf{r}}_{k-1}}.$$

Then,

$$\underline{\mathbf{x}}_k \in \underline{\mathbf{x}}_0 + \underbrace{\mathcal{K}^k(\underline{\underline{M}}^{-1}\underline{\underline{A}}, \underline{\underline{M}}^{-1}\underline{\mathbf{r}}_0)}_{=:\mathcal{K}^k}, \qquad \textit{for all } k \geq 1.$$

Proof. First of all, note that the statement is true for $k = 1$:

$$\underline{\mathbf{x}}_1 \quad = \quad \underline{\mathbf{x}}_0 + \underline{\underline{M}}^{-1}\underline{\mathbf{r}}_0 \in \underline{\mathbf{x}}_0 + \mathcal{K}^1.$$

Now, suppose that $\underline{\mathbf{x}}_k \in \underline{\mathbf{x}}_0 + \mathcal{K}^k$ $(k \geq 1)$. The residual $\underline{\mathbf{r}}_k$ can be expressed in $\underline{\mathbf{r}}_0$, according to:

$$\begin{aligned}
\underline{\mathbf{r}}_k &= \underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k \\
&= \underline{\mathbf{b}} - \underline{\underline{A}}\,(\underline{\mathbf{x}}_{k-1} + \underline{\underline{M}}^{-1}\underline{\mathbf{r}}_{k-1}) \\
&= \underline{\mathbf{r}}_{k-1} - \underline{\underline{A}}\,\underline{\underline{M}}^{-1}\underline{\mathbf{r}}_{k-1} \\
&= (\underline{\underline{I}} - \underline{\underline{A}}\,\underline{\underline{M}}^{-1})\underline{\mathbf{r}}_{k-1} \\
&= (\underline{\underline{I}} - \underline{\underline{A}}\,\underline{\underline{M}}^{-1})^k\underline{\mathbf{r}}_0.
\end{aligned}$$

Hence,

$$\underline{\mathbf{x}}_{k+1} = \underline{\mathbf{x}}_k + \underline{\underline{M}}^{-1}\underline{\mathbf{r}}_k = \underbrace{\underline{\mathbf{x}}_k}_{\in\underline{\mathbf{x}}_0+\mathcal{K}^k} + \underbrace{\underline{\underline{M}}^{-1}(\underline{\underline{I}} - \underline{\underline{A}}\,\underline{\underline{M}}^{-1})^k\underline{\mathbf{r}}_0}_{\in\mathcal{K}^{k+1}} \in \underline{\mathbf{x}}_0 + \mathcal{K}^{k+1}.$$

■

Method 8.21 (Krylov method): A *Krylov method* follows from Method 8.9 by choosing $\underline{\mathbf{x}}_k$ such that:

1. $\underline{\mathbf{x}}_k \in \underline{\mathbf{x}}_0 + \mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$,
2. $\underline{\mathbf{r}}_k = \underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k \perp \mathcal{L}^k$.

Here, $\mathcal{L}^k$ is a k-dimensional subspace of $\mathbb{R}^n$. ⌟

The nature of a Krylov method is mainly determined by two aspects. Of course, the choice of $\mathcal{L}^k$ plays an important role. Additionally, there are several ways to construct a basis for $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$. Three choices are listed below, including the methods resulting from them.

1. Arnoldi provides an orthonormal basis $\mathcal{V} = \{\underline{\mathbf{v}}_1, ..., \underline{\mathbf{v}}_k\}$ for $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{v}}_1)$ (variants: Arnoldi-Modified Gram-Schmidt and Householder Arnoldi)
   - $\mathcal{L}^k = \mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$: Full Orthogonalization Method (FOM) (variants: Restarted FOM (FOM(k), Incomplete Orthogonalisation Method (IOM), Direct IOM (DIOM))
   - $\mathcal{L}^k = \underline{\underline{A}}\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$: General Minimal RESidual (GMRES) (variants: Restarted GMRES (GMRES(k)), Quasi-GMRES (QGMRES), Direct QGMRES (DQGMRES))
2. Lanczos is like Arnoldi, but only applicable to symmetric matrices (variant: Direct Lanczos (D-Lanczos))
   - $\mathcal{L}^k = \mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$: Conjugate Gradient (GG) (for positive definite matrices) (variant: CG-Three-term recurrence variant (for positive definite matrices))
   - $\mathcal{L}^k = \underline{\underline{A}}\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$: Conjugate Residual (CR) (for positive definite hermitian matrices)
3. Lanczos Biothogonalisation (BiLanczos) computes a basis $\mathcal{V} = \{\underline{\mathbf{v}}_1, ..., \underline{\mathbf{v}}_k\}$ for $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{v}}_1)$ and a basis $\mathcal{W} = \{\underline{\mathbf{w}}_1, ..., \underline{\mathbf{w}}_k\}$ for $\mathcal{K}^k(\underline{\underline{A}}^T, \underline{\mathbf{w}}_1)$, such that $\underline{\mathbf{v}}_i \perp \underline{\mathbf{w}}_j$ for all $i, j = 1, ..., k$
   - $\mathcal{L}^k = \mathcal{K}^k(\underline{\underline{A}}^T, \underline{\mathbf{r}}_0)$: BiConjugate Gradient (BCG) (variants: Conjugate Gradient Squared (CGS), Biconjugate Gradient Stabilized (BICGSTAB)) and Quasi Minimal Residual (QMR) (variant: Transpose Free QMR (TFQMR)

In the next sections, Arnoldi and GMRES will be considered in more detail. For more information about other Krylov methods, see [Saa00, Chapter 6 and 7].

Arnoldi

The Arnoldi method constructs an orthonormal basis for a Krylov space with the help of Gram-Schmidt.

Method 8.22 (**Arnoldi**)**:** The *Arnoldi* method constructs an orthonormal basis $\{\underline{\mathbf{v}}_1, ..., \underline{\mathbf{v}}_k\}$ for the Krylov space $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}})$ by means of the following algorithm:

1. $\underline{\mathbf{v}}_1 = \frac{1}{\|r\|_2}\underline{\mathbf{r}}$
2. for $j = 1, .., k$:
3. for $i = 1, ..., j$: $h_{ij} = \left(\underline{\underline{A}}\,\underline{\mathbf{v}}_j\right)^T \underline{\mathbf{v}}_i$
4. $\underline{\mathbf{v}}_{j+1} = \underline{\mathbf{v}}_{j+1} - \sum_{i=1}^{j} h_{ij}\underline{\mathbf{v}}_i$
5. $h_{j+1,j} = \|\underline{\mathbf{v}}_{j+1}\|_2$
6. if $h_{j+1,j} = 0$: stop
7. $\underline{\mathbf{v}}_{j+1} = \frac{1}{h_{j+1,j}}\underline{\mathbf{v}}_{j+1}$
8. end ⌟

The following variant of the algorithm above uses Modified Gram-Schmidt instead of Gram-Schmidt.

Method 8.23 (**Arnoldi-Modified Gram-Schmidt**)**:** The *Arnoldi-Modified Gram-Schmidt* method constructs an orthonormal basis $\{\underline{\mathbf{v}}_1, ..., \underline{\mathbf{v}}_k\}$ for the Krylov space $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}})$ by means of the following algorithm.

1. $\underline{\mathbf{v}}_1 = \frac{1}{\|r\|_2}\underline{\mathbf{r}}$
2. for $j = 1, .., k$:
3. $\underline{\mathbf{v}}_{j+1} = \underline{\underline{A}}\,\underline{\mathbf{v}}_j$
4. for $i = 1, ..., j$:
5. $h_{ij} = \underline{\mathbf{v}}_{j+1}^T\underline{\mathbf{v}}_i$
6. $\underline{\mathbf{v}}_{j+1} = \underline{\mathbf{v}}_{j+1} - h_{ij}\underline{\mathbf{v}}_i$
7. end

8. $h_{j+1,j} = \|\underline{\mathbf{v}}_{j+1}\|_2$
9. if $h_{j+1,j} = 0$: stop
10. $\underline{\mathbf{v}}_{j+1} = \frac{1}{h_{j+1,j}} \underline{\mathbf{v}}_{j+1}$
11. end ⌟

In theory, the results of both algorithms are the same. In practice, Arnoldi-Modified Gram-Schmidt is less sensitive to round-off errors. Another variant is Householder Arnoldi [Saa00, p. 149], which is even more reliable, but also more expensive.

General minimal residual method

The general minimal residual method [SS86] is a Krylov method that uses $\mathcal{L}^k = \underline{\underline{A}}\,\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$. As its name already indicates, it is based on minimizing the residual $\underline{\mathbf{r}}_k = \underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k$. Each step, $\underline{\mathbf{x}}_k$ is chosen such that:

$$\begin{aligned} \underline{\mathbf{x}}_k &= \underline{\mathbf{x}}_0 + \arg \min_{\underline{\mathbf{z}} \in \mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)} \|\underline{\mathbf{b}} - \underline{\underline{A}}\,(\underline{\mathbf{x}}_0 + \underline{\mathbf{z}})\|_2 \\ &= \underline{\mathbf{x}}_0 + \arg \min_{\underline{\mathbf{z}} \in \mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)} \|\underline{\mathbf{r}}_0 - \underline{\underline{A}}\,\underline{\mathbf{z}}\|_2. \end{aligned}$$

Since this involves a minimization problem that is not trivial to solve, it will be rewritten.

Proposition 8.24: *Let $\underline{\mathbf{v}}_j$ (for j=1,...,k+1) and h_{ij} (for $i = 1, ..., k$ and $j = 1, ..., k+1$) result from applying (a variant of) Arnoldi to $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$. Define $\underline{\underline{V}}_k \in \mathbb{R}^{n \times k}$ and $\underline{\underline{H}}_k \in \mathbb{R}^{k+1 \times k}$ (also known as the* Hessenberg matrix*) such that:*

$$\underline{\underline{V}}_k = \begin{bmatrix} \underline{\mathbf{v}}_1 & \dots & \underline{\mathbf{v}}_k \end{bmatrix}, \tag{8.4}$$

$$\underline{\underline{H}}_k = \begin{bmatrix} h_{11} & \dots & h_{1k} \\ h_{21} & \dots & h_{2k} \\ & \ddots & \vdots \\ & & h_{k+1,k} \end{bmatrix}. \tag{8.5}$$

Define $\underline{\mathbf{y}}_k$ as the solution of the following linear least squares problem:

$$\underline{\mathbf{y}}_k = \arg \min_{\underline{\mathbf{y}} \in \mathbb{R}^k} \left\| \|\underline{\mathbf{r}}_0\|_2 \underline{\mathbf{e}}_1 - \underline{\underline{H}}_k \underline{\mathbf{y}} \right\|_2. \tag{8.6}$$

Then

$$\arg \min_{\underline{\mathbf{z}} \in \mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)} \|\underline{\mathbf{r}}_0 - \underline{\underline{A}}\,\underline{\mathbf{z}}\|_2 = \underline{\underline{V}}_k \underline{\mathbf{y}}_k.$$

Proof. See [Saa00, Section 6.5.1]. ■

In order to determine $\underline{\mathbf{y}}_k$ in (8.6) efficiently, $\underline{\underline{H}}_k$ will be transformed by what are called *Givens rotations* in order to achieve the following structure:

$$\underline{\underline{H}}_k^{(i)} = \begin{bmatrix} h_{11}^{(i)} & \dots & \dots & \dots & h_{1k}^{(i)} \\ & \ddots & & & \vdots \\ & & h_{i+1,i+1}^{(i)} & & \vdots \\ & & h_{i+2,i+1}^{(i)} & \ddots & \vdots \\ & & & \ddots & h_{kk}^{(i)} \\ & & & & h_{k+1,k}^{(i)} \end{bmatrix}.$$

Note that in particular $\underline{\underline{H}}_k^{(k)}$ will have a favorable structure.

Proposition 8.25: *Let* $\underline{\underline{H}}_k$ *and* $\underline{\mathbf{r}}_0$ *be as in Proposition 8.24. Define:*

$$\underline{\underline{H}}_k^{(0)} = \underline{\underline{H}}_k. \tag{8.7}$$

Let $\underline{\underline{I}}_i$ *be the* $i \times i$ *identity matrix. Introduce Givens rotations* $\Omega_i \in \mathbb{R}^{k+1 \times k+1}$ $(i = 1, ..., k)$ *according to:*

$$\underline{\underline{\Omega}}_i = \begin{bmatrix} \underline{\underline{I}}_{i-1} & & & \\ & c_i & s_i & \\ & -s_i & c_i & \\ & & & \underline{\underline{I}}_{k-i} \end{bmatrix}, \tag{8.8}$$

$$c_i = \frac{h_{i+1,i}}{\sqrt{\left(h_{i,i}^{(i-1)}\right)^2 + h_{i+1,i}^2}}, \tag{8.9}$$

$$s_i = \frac{h_{i,i}^{(i-1)}}{\sqrt{\left(h_{i,i}^{(i-1)}\right)^2 + h_{i+1,i}^2}}. \tag{8.10}$$

Apply the Givens rotations to $\underline{\underline{H}}_k$ *and* $\|\underline{\mathbf{r}}_0\|_2 \underline{\mathbf{e}}_1$ *to obtain:*

$$\underline{\underline{H}}_k^{(i)} = \underline{\underline{\Omega}}_i ... \underline{\underline{\Omega}}_1 \underline{\underline{H}}_k, \tag{8.11}$$

$$\underline{\mathbf{g}}^{(i)} = \underline{\underline{\Omega}}_i ... \underline{\underline{\Omega}}_1 \|\underline{\mathbf{r}}_0\|_2 \underline{\mathbf{e}}_1. \tag{8.12}$$

Then, the following two statements hold:

1. *The solution of the linear least square problem (8.6) satisfies:*

$$\hat{\underline{\underline{H}}}_k^{(k)} \underline{\mathbf{y}}_k = \hat{\underline{\mathbf{g}}}^{(k)},$$

   in which $\hat{\underline{\underline{H}}}_k^{(k)}$ *and* $\hat{\underline{\mathbf{g}}}^{(k)}$ *result from* $\underline{\underline{H}}_k^{(k)}$ *and* $\underline{\mathbf{g}}^{(k)}$ *by deleting their last rows. Note that* $\hat{\underline{\underline{H}}}_k^{(k)}$ *is an upper triangular matrix, so the system is easily solved by means of backward substitution (Method 8.3).*

2. *The 2-norm of the residual,* $\|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k\|_2$*, is given by the last element of* $\underline{\mathbf{g}}^{(k)}$*. This result is convenient for the implementation of a stopping criterion.*

Proof. See [Saa00, Proposition 6.9] ■

Putting it all together:

Method 8.26 (General Minimal RESidual method (GMRES)): The *general minimal residual method* follows from Method 8.9 by using the following strategy to obtain $\underline{\mathbf{x}}_k$:

1. Compute $\underline{\underline{H}}_k$ and $\underline{\underline{V}}_k$ by applying (a variant of) Arnoldi to $\mathcal{K}^k(\underline{\underline{A}}, \underline{\mathbf{r}}_0)$ and using equations (8.4) and (8.5).
2. Determine $\underline{\underline{H}}_k^{(k)}$ and $\underline{\mathbf{g}}^{(k)}$ using equations (8.7)-(8.12).
3. Calculate $\underline{\mathbf{y}}_k$ by means of the first statement of Proposition 8.25.
4. $\underline{\mathbf{x}}_k = \underline{\mathbf{x}}_0 + \underline{\underline{V}}_k \underline{\mathbf{y}}_k$.

Note that the vectors $\underline{\mathbf{y}}_k$ and $\underline{\mathbf{x}}_k$ only need to be computed after the termination criterion has been reached. ⌟

Theorem 8.27 (Convergence of GMRES)**:** *Consider a diagonalizable matrix* $\underline{\underline{A}}$ *with eigenvalues* $\lambda_1, ..., \lambda_n$ *and corresponding eigenvectors* $\underline{\mathbf{v}}_1, ..., \underline{\mathbf{v}}_n$*. So,* $\underline{\underline{V}} = [\underline{\mathbf{v}}_1 \dots \underline{\mathbf{v}}_n]$ *is an invertible matrix. Then, the relative error in step* k *of GMRES (Method 8.26) satisfies:*

$$\frac{\|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k\|_2}{\|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_0\|_2} \leq \|\underline{\underline{V}}\|_2 \|\underline{\underline{V}}^{-1}\|_2 \min_{p\in\mathbb{P}_k, p(0)=1} \max_{i\{1,\dots,n\}} |p(\lambda_i)|.$$

If, moreover, all eigenvalues are contained in an ellipse $\mathcal{E} \subset \mathbb{C}$*, excluding the origin and having center* $c \in \mathbb{C}$*, focal distance* $d \in \mathbb{C}$*, and semi major axis* $a \in \mathbb{C}$ *(see Figure 8.1 for an example), then, the relative error satisfies:*

$$\begin{aligned}\frac{\|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k\|_2}{\|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_0\|_2} &\leq \|\underline{\underline{V}}\|_2 \|\underline{\underline{V}}^{-1}\|_2 \left|\frac{p_k\left(\frac{a}{d}\right)}{p_k\left(\frac{c}{d}\right)}\right| \\ &\approx \|\underline{\underline{V}}\|_2 \|\underline{\underline{V}}^{-1}\|_2 \left|\frac{a+\sqrt{a^2-d^2}}{c+\sqrt{c^2-d^2}}\right|^k.\end{aligned}$$

Here, $p_k : \mathbb{C} \to \mathbb{C}$ *is the complex* Chebychev polynomial*, which can be defined recursively according to:*

$$\begin{aligned} p_1(z) &= 1, \\ p_2(z) &= z, \\ p_k(z) &= 2zp_k(z) - p_{k-1}(z) \qquad (k \geq 2). \end{aligned}$$

Proof. See [Saa00, Proposition 6.15, Corollary 6.1, and (6.100)] ■

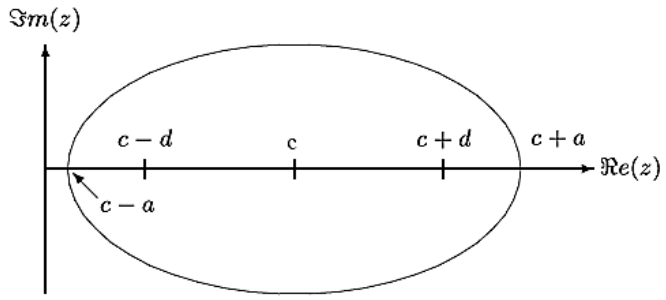


Figure 8.1: Example of an ellipse for $c, d, a \in \mathbb{R}$

8.3 Summary

If an implicit FVM is used to solve the water quality model, many large sparse linear systems need to be solved. For such systems, iterative methods are more suitable than direct methods. At present, GMRES is implemented in WAQ. This Krylov method converges faster as the eigenvalues are more clustered.

Chapter 9

Preconditioning

In practice, iterative methods often have an unsatisfactory convergence speed. A popular way to deal with this problem is enhancing the spectrum by means of preconditioning, which is discussed in this chapter.

9.1 Basic preconditioning

Preconditioning transforms a linear system into an equivalent system that can be handled better by an iterative method.

Method 9.1 (**Preconditioning: general form**)**:** Consider an $n \times n$ linear system $\underline{\underline{A}}\,\underline{\mathbf{x}} = \underline{\mathbf{b}}$. *Preconditioning* consists of the following steps:

1. Choose invertible matrices $\underline{\underline{P}}_l$, $\underline{\underline{P}}_r \in \mathbb{R}^{n\times n}$, the preconditioners.
2. Solve $\underline{\mathbf{y}}$ from:
$$\underline{\underline{P}}_l^{-1}\,\underline{\underline{A}}\,\underline{\underline{P}}_r^{-1}\underline{\mathbf{y}} \quad = \quad \underline{\underline{P}}_l^{-1}\underline{\mathbf{b}}.$$
3. Determine $\underline{\mathbf{x}}$ according to:
$$\underline{\mathbf{x}} \quad = \quad \underline{\underline{P}}_r^{-1}\underline{\mathbf{y}}.$$

⌟

Remark 9.2: If $\underline{\underline{P}}_r = \underline{\underline{I}}$, one speaks of *left preconditioning*. Similarly, choosing $\underline{\underline{P}}_l = \underline{\underline{I}}$ results in *right preconditioning*. Usually, the termination criterion of an iterative method is based on the residual norm $\|\underline{\mathbf{r}}_k\| = \|\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k\|$. Preconditioning translates the residual norm to $\|\underline{\underline{P}}_l^{-1}\underline{\mathbf{r}}_k\|$. For this reason, right preconditioning is often preferred. ⌟

Remark 9.3 (Recycling)**:** In general, it is more costly to compute a preconditioner than to solve the preconditioned system. Therefore, it might be efficient to recycle the preconditioner, i.e. using the same preconditioner in multiple time steps. If the matrix does not vary much in time, the preconditioner hopefully remains effective. ⌟

Good preconditioners at least guarantee that:

1. $\underline{\underline{P}}_l$ and $\underline{\underline{P}}_r$ can be determined inexpensively;
2. the spectrum[1] of $\underline{\underline{P}}_l^{-1}\,\underline{\underline{A}}\,\underline{\underline{P}}_r^{-1}$ is favorable with respect to convergence[2];
3. $\underline{\underline{P}}_l^{-1}\underline{\mathbf{v}}$ and $\underline{\underline{P}}_r^{-1}\underline{\mathbf{v}}$ can be computed at low cost ($\underline{\mathbf{v}} \in \mathbb{R}^n$).

What are suitable preconditioners? This will be treated in the following sections.

[1] set of eigenvalues
[2] Remember Theorem 8.27 and 8.15

9.2 Preconditioners based on matrix splitting

Matrix splitting gives rise to a class of preconditioners that are relatively simple to construct.

Method 9.4 (**Preconditioners based on matrix splitting**)**:** There are two ways to derive preconditioners from a matrix splitting $\underline{\underline{A}} = \underline{\underline{M}} - \underline{\underline{N}}$:

1. $\underline{\underline{P}}_l = \underline{\underline{M}}$,
   $\underline{\underline{P}}_r = \underline{\underline{I}}$.

2. $\underline{\underline{P}}_l = \underline{\underline{I}}$,
   $\underline{\underline{P}}_r = \underline{\underline{M}}$.

Note that the construction costs are 0 flops. ⌟

Example 9.5 (Symmetric GS preconditioner)**:** An example of the preconditioner above is the symmetric GS preconditioner, which is currently used in WAQ to speed up GMRES (schemes 15 and 16). Remembering Method 8.18 and Proposition 8.17 results in:

$$\underline{\underline{M}} = \underline{\underline{M}}_1 \left(\underline{\underline{M}}_1 + \underline{\underline{N}}_2 \right)^{-1} \underline{\underline{M}}_2 = \left(\underline{\underline{D}} + \underline{\underline{L}} \right) \underline{\underline{D}}^{-1} \left(\underline{\underline{D}} + \underline{\underline{U}} \right). \qquad ⌟$$

The symmetric GS preconditioner seems to be inadequate for diffusion dominated problems. This is illustrated in Figure 9.1, in which the relative residual $\frac{\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k}{\|\underline{\mathbf{b}}\|_2}$ is plotted for each iteration k for a two-dimensional problem. The blue line corresponds to the actual diffusion dominated case in which the diffusion coefficient has been increased by $10\,\mathrm{m}^2\,\mathrm{s}$ (see Remark 3.6). The red line is the result of neglecting this amount of extra diffusion. Indeed, the current preconditioning seems unsuitable for diffusion dominated problems.

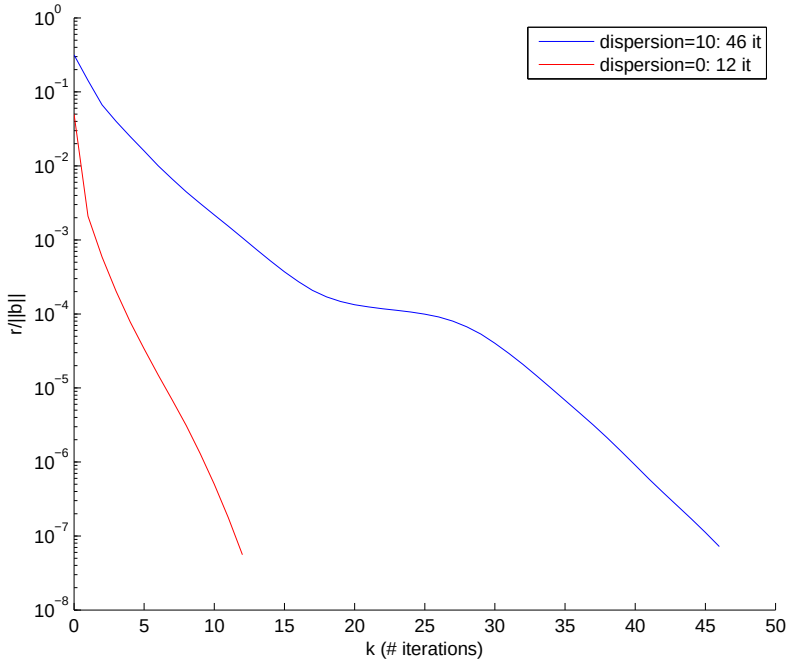


Figure 9.1: The effect of diffusion dominance on the convergence of GMRES

9.3 Preconditioners based on an incomplete LU factorisation

As mentioned before, LU factorisations are inefficient for sparse matrices (due to fill-in), which is why iterative methods are essential in the first place. However, approximate factorisations often result in powerful preconditioners.

Definition 9.6 (**Incomplete LU factorisation**)**:** An *incomplete LU factorisation* of a matrix $\underline{\underline{A}}$ consists of a unit lower triangular matrix $\underline{\underline{\tilde{L}}}$, an upper triangular matrix $\underline{\underline{\tilde{U}}}$, and a matrix $\underline{\underline{R}}$ such that:

$$\underline{\underline{A}} = \underline{\underline{\tilde{L}}}\,\underline{\underline{\tilde{U}}} - \underline{\underline{R}}.$$

⌟

Method 9.7 (**Preconditioners based on an incomplete LU factorisation**)**:** There are three ways to derive preconditioners from an incomplete LU factorisation:

1. $\underline{\underline{P}}_l = \underline{\underline{\tilde{L}}}\,\underline{\underline{\tilde{U}}}$
   $\underline{\underline{P}}_r = \underline{\underline{I}}$
2. $\underline{\underline{P}}_l = \underline{\underline{I}}$
   $\underline{\underline{P}}_r = \underline{\underline{\tilde{L}}}\,\underline{\underline{\tilde{U}}}$
3. $\underline{\underline{P}}_l = \underline{\underline{\tilde{L}}}$
   $\underline{\underline{P}}_r = \underline{\underline{\tilde{U}}}$

⌟

In the sections hereafter, several algorithms to compute an incomplete LU factorisation will be discussed.

9.3.1 Incomplete LU threshold

The incomplete LU threshold method provides a basic strategy to compute an incomplete LU factorisation. The algorithm results from adding two dropping rules to the algorithm that computes an ordinary LU factorisation (see Method 8.5). A dropping rule sets an element equal to zero if it satisfies certain criteria. To put it more bluntly: If you don't want to compute it, discard it.

Method 9.8 (**Incomplete LU Threshold (ILUT)**)**:** The *incomplete LU threshold* algorithm computes an incomplete LU factorisation $\underline{\underline{A}} = \underline{\underline{\tilde{L}}}\,\underline{\underline{\tilde{U}}} - \underline{\underline{R}}$ by means of the following algorithm, provided that the pivots, $\tilde{u}_{kk}$, are nonzero:

1. for $i = 1, ..., n$:
2. $\quad w := a_{i*}$
3. $\quad$ for $k = 1, ..., i-1$:
4. $\quad\quad w_k = \frac{w_k}{\tilde{u}_{kk}}$
5. $\quad\quad$ Apply a dropping rule to w_k
6. $\quad\quad w := w - w_k \tilde{u}_{k*}$
7. $\quad$ end
8. $\quad$ Apply a dropping rule to w
9. $\quad \tilde{l}_{ij} = w_j$ for $j = 1, ..., i-1$
10. $\quad\ \tilde{u}_{ij} = w_j$ for $j = i, ..., n$
11. end

⌟

An application of Method 9.8 is ILUT(p,τ), which drops elements that are small in some sense. Moreover, it limits the number of elements per row.

Method 9.9 (**ILUT(p,τ)**)**:** *ILUT(p,τ)* follows from Method 9.8 by using the following dropping rules:

Line 5 is replaced by:

if $w_k < \tau \|a_{i*}\|_2$: $w_k = 0$

Line 8 is replaced by:

for $k = 1, ..., n$:
  if $w_k < \tau \|a_{i*}\|_2$: $w_k = 0$
end
Drop all elements in $\underline{\mathbf{w}}$, except w_i, the p largest elements in $\{w_1, ..., w_{i-1}\}$, and the p largest elements in $\{w_{i+1}, ..., w_n\}$.

If A contains at most q nonzero elements per row, the costs of this factorisation amount to approximately

$$n\Big(p\underbrace{\big(2(p+1)+1\big)}_{\text{row update}} + \underbrace{2q}_{\|a_{i*}\|_2}\Big)$$

flops. ⌟

9.3.2 Incomplete LU

Incomplete LU preconditioners form a subcategory of ILUT preconditioners. Their dropping rules are based on a zero pattern.

Method 9.10 (Incomplete LU (ILU): general form): Let

$$\mathcal{Z} \subset \{(i,j) \in [1, ..., n] \times [1, ..., n] : i \neq j\}$$

be a zero pattern. The general *Incomplete LU* algorithm follows from Method 9.8 by letting both dropping rules set w_k equal to zero if $(i,k) \notin \mathcal{Z}$. In other words:

Line 5 is replaced by:

if $(i,k) \in \mathcal{Z}$: $w_k = 0$

Line 8 is replaced by:

for $k = 1, ..., n$:
  if $(i,k) \in \mathcal{Z}$: $w_k = 0$
end

Note that $\underline{\underline{R}}$ follows from:

$$r_{ij} = \begin{cases} a_{ij}, & (i,j) \in \mathcal{Z}, \\ 0, & (i,j) \notin \mathcal{Z}. \end{cases}$$ ⌟

An application of Method 9.10 is $ILU(0)$.

Method 9.11 (ILU(0)): *ILU(0)* follows from Method 9.10 by taking $\mathcal{Z}$ equal to the zero pattern of $\underline{\underline{A}}$. ⌟

In practice, ILU(0) can have insufficient accuracy, resulting in inefficiency and unreliability. ILU(0) has no fill-in. ILU(p), the generalisation of ILU(0), attempts to improve ILU(0) by allowing some fill-in. The method drops elements that have a level of fill (see Method 9.12 below for a definition) larger than p.

Method 9.12 (ILU(p)): Let f_{ij} denote the level of fill. Initially, this quantity is defined as follows:

$$f_{ij} = \begin{cases} 0, & a_{ij} \neq 0 \text{ or } i = j, \\ \infty, & a_{ij} = 0. \end{cases}$$

ILU(p) follows from Method 9.10 by using the zero pattern:

$$\mathcal{Z} = \{(i,j) \in [1,...,n] \times [1,...,n] : f_{ij} > p\}\,.$$

Right before the end of the k-loop (so before line 7 in Method 9.8), f_{ij} is updated according to:

$$f_{ij} \quad = \quad \begin{cases} \min\{f_{ij}, f_{ik} + f_{kj} + 1\}, & w_j \neq 0, \\ f_{ij}, & w_j = 0, \end{cases} \qquad j = 1,...,n.$$

If $\underline{\underline{\tilde{L}}}$ has at most $\tilde{q}_l$ nonzero off-diagonal elements per row and $\underline{\underline{\tilde{U}}}$ has at most $\tilde{q}_u$ nonzero off-diagonal elements per row, then the costs of the factorization amount to approximately

$$n\tilde{q}_l\big(\underbrace{(2\tilde{q}_u + 1)}_{\text{row update}} + \underbrace{(\tilde{q}_l + \tilde{q}_u + 1)2}_{\text{level of fill update}}\big)$$

flops. ⌟

ILU(p) might be a good alternative for the current preconditioner. This idea is based on Figure 9.2, in which the relative residual $\frac{\underline{\mathbf{b}} - \underline{\underline{A}}\,\underline{\mathbf{x}}_k}{\|\underline{\mathbf{b}}\|_2}$ has been plotted for each iteration k for a two-dimensional diffusion dominated problem. The black line, which corresponds to the case without preconditioning, shows that preconditioning is indispensable. The blue line, which is the result of the symmetric Gauss-Seidel preconditioner, demonstrates that the current preconditioner is insufficient for this type of problems. The red line, which coincides with ILU(3) preconditioning, illustrates that ILU(p) performs rather well for the diffusion dominates problems.

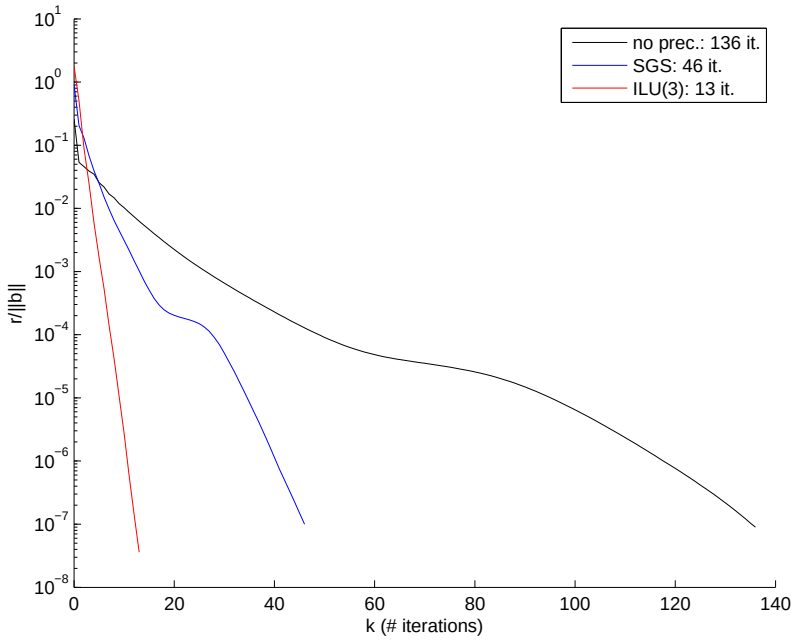


Figure 9.2: The effect of preconditioning on the convergence of GMRES

Nonetheless, according to Saad [Saa00, p. 280], there "are a number of drawbacks to the above algorithm. First, the amount of fill-in and computational work for obtaining the ILU(p) factorization is not predictable for $p > 0$. Second, the cost of updating the levels can be quite high. Most

importantly, the level of fill for indefinite matrices may not be a good indicator of the size of the elements that are being dropped. Thus, the algorithm may drop large elements and result in an inaccurate incomplete factorisation".

So far, the elements that were dropped were simply discarded. Modified ILU uses a different approach. It adds dropped elements to the diagonal of $\underline{\underline{\tilde{U}}}$.

Method 9.13 (Modified ILU (MILU)): *Modified ILU* follows from Method 9.10, by inserting the following diagonal update of $\underline{\underline{\tilde{U}}}$ right after line 10 in Method 9.8:

$$\tilde{u}_{ii} \quad := \quad \tilde{u}_{ii} + \sum_{m=1}^{n} r_{im}. \qquad \lrcorner$$

MILU guarantees that $\underline{\underline{A}}$ and $\underline{\underline{\tilde{L}}}\,\underline{\underline{\tilde{U}}}$ have the same row sums. Its results are especially good for matrices resulting from the discretisation of a PDE that has a more or less constant solution.

9.4 Summary

The convergence speed of an iterative method depends on the spectrum of the matrix. Preconditioning transforms a linear system into an equivalent system that has better spectral properties. At present, WAQ uses symmetric Gauss-Seidel preconditioning, which is costless to construct. Unfortunately, for diffusion dominated problems, this preconditioner does not lead to a satisfactory reduction of the number of iterations. An alternative preconditioning, such as ILU(p), could lead to a much smaller number of iterations. However, construction costs and the costs per iteration should also be taken into account.

Chapter 10

Reordering

In Chapter 9, preconditioning was introduced to speed up the convergence of an iterative method. The construction of a good preconditioner is generally expensive. An important tool in facing this problem, which is especially useful for matrices arising from discretisation on an unstructured grid, is reordering of the matrix elements in advance. This is treated in this chapter.

In this chapter, level-set ordering, independent set ordering, and multi-color ordering are discussed. Other orderings can be found in [DER86, Chapter 8] and [GL81, Chapter 5].

10.1 Symmetric permutation

Reordering is actually is special case of preconditioning, in which the preconditioners are permutation matrices. A permutation matrix is the identity matrix with its rows or columns permuted.

Definition 10.1 (**Interchange matrix**)**:** An *interchange matrix* is the identity matrix with two of its rows interchanged. ⌟

Definition 10.2 (**Permutation matrix**)**:** A *permutation matrix* $\underline{\underline{P}} \in \mathbb{R}^{n\times n}$ is a product of (at most n) interchange matrices. ⌟

Proposition 10.3: *If* $\underline{\underline{P}}$ *is a permutation matrix, then*

$$\underline{\underline{P}}^{-1} = \underline{\underline{P}}^{T}.$$

Proof. See [Saa00, p. 73] ■

Method 10.4 (**Symmetric permutation**)**:** A *symmetric permutation* follows from Method 9.1 by using

$$\begin{aligned} \underline{\underline{P}}_l &= \underline{\underline{P}}^{T}, \\ \underline{\underline{P}}_r &= \underline{\underline{P}}. \end{aligned}$$

⌟

Example 10.5: Consider a system

$$\underbrace{\begin{bmatrix} a_{11} & 0 & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & a_{42} & 0 & a_{44} \end{bmatrix}}_{\underline{\underline{A}}} \underline{\mathbf{x}} = \underline{\mathbf{b}}.$$

Choose the following permutation matrix:

$$\underline{\underline{P}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Then

$$\underline{\underline{P}}^T \underline{\underline{A}}\, \underline{\underline{P}} = \begin{bmatrix} a_{11} & a_{13} & 0 & 0 \\ a_{31} & a_{33} & a_{32} & 0 \\ 0 & a_{23} & a_{22} & a_{24} \\ 0 & 0 & a_{42} & a_{44} \end{bmatrix}.$$

Note that an (I)LU-factorisation of this matrix would have zero fill-in. ⌟

10.2 Renumbering the adjacency graph

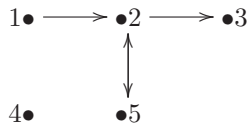
A symmetric permutation of a matrix is equivalent to renumbering the vertices of its adjacency graph [Saa00, p.75].

Definition 10.6 (**Graph**)**:** A *graph* G consists of a set of vertices $\mathcal{V} = \{v_1, ..., v_n\}$ and a set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. Notation: $G = (\mathcal{V}, \mathcal{E})$. ⌟

Definition 10.7 (**Adjacency graph**)**:** The *adjacency graph* of a matrix $\underline{\underline{A}} \in \mathbb{R}^{n \times n}$ is a graph $G = (\mathcal{V}, \mathcal{E})$ such that:

- The vertices represent the unknowns, i.e. $v_1, ..., v_n \in \mathcal{V}$.
- The edges represent the nonzero elements of the matrix, in other words, $(v_i, v_j) \in \mathcal{E}$, if $a_{ij} \neq 0$ and $i \neq j$. ⌟

Example 10.8 (Adjacency graph)**:** The adjacency graph



corresponds to the matrix structure

$$\begin{bmatrix} * & * & & & \\ & * & * & & * \\ & & * & & \\ & & & * & \\ & * & & & * \end{bmatrix}.$$

⌟

10.2.1 Level-set orderings

Level-set orderings are based on traversing the graph by level sets.

Definition 10.9 (**Adjacent**)**:** Two vertices in a graph are *adjacent* if they have a common edge. To put it more precisely: If $G = (\mathcal{V}, \mathcal{E})$ is a graph, then $v, w \in \mathcal{V}$ are adjacent, if $(v, w) \in \mathcal{E}$ or $(w, v) \in \mathcal{E}$. ⌟

Definition 10.10 (**Level set**)**:** A *level set* of a graph $G = (\mathcal{V}, \mathcal{E})$ is a recursively defined subset of $\mathcal{V}$. The initial level set L_1 can be any subset of $\mathcal{V}$. Each next level-set L_k $(k \geq 2)$ contains the unmarked neighbors of the vertices of the previous level set:

$$L_k \quad = \quad \{v \in \mathcal{V} \setminus (L_1 \cup ... \cup L_{k-1}) : \exists w \in L_1 \cup ... \cup L_{k-1} \text{ adjacent to } v\}.$$ ⌟

Method 10.11 (**Level set ordering**)**:** Consider a graph $G = (\mathcal{V}, \mathcal{E})$. A basic *level set ordering* can be constructed by applying the following steps:

1. Choose an initial level set $L_1 = \{v_{m_1}, ..., v_{M_1}\} \subset \mathcal{V}$. Mark all vertices $v \in L_1$.
2. While unmarked vertices are available: Determine the next level set $L_k = \{v_{m_k}, ..., v_{M_k}\}$ by traversing L_{k-1} in a certain way. Mark all vertices $v \in L_k$.
3. Order the vertices in the following manner:

$$v_{m_1}, ..., v_{M_1}, v_{m_2}, ..., v_{M_2}, ...$$ ⌟

Level set orderings differ from one another in initial level set, way of traversing, and way of numbering. The Cuthill-McKee ordering, for example, is based on the degrees of the vertices.

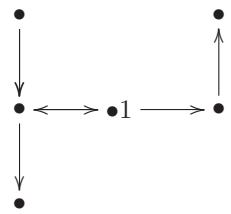
Definition 10.12 (**Degree**)**:** The *degree* of a vertex of a graph is the number of edges incident to it. Loops[1] are counted twice. ⌟

Method 10.13 (**Cuthill-McKee (CMK) ordering**)**:** The *Cuthill-McKee ordering* follows from Method 10.11 by using the following strategies:

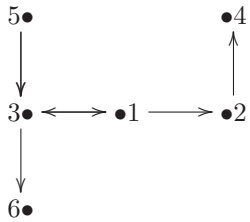
- The initial level set consists of a single node: $L_1 = \{v_{m_1}\}$.
- The elements of a level set are traversed from the nodes of lowest degree to those of highest degree.
- Nodes are numbered in the same order as they are traversed. ⌟

CMK ordering normally leads to a smaller bandwidth.

Example 10.14 (CMK ordering)**:** Consider the following adjacency graph with initial level set $\{1\}$:



Applying CMK ordering yields:



⌟

[1] Note that loops do not occur in adjacency graphs

10.2.2 Independent set ordering

An independent set ordering isolates unknowns that are independent of one another. This results in a matrix of the form:

$$\begin{bmatrix} \underline{\underline{D}} & \underline{\underline{E}} \\ \underline{\underline{F}} & \underline{\underline{C}} \end{bmatrix},$$

in which $\underline{\underline{D}}$ is a diagonal matrix. This structure is especially useful for parallel computing.

Definition 10.15 (**Independent set**)**:** An *independent set* of a graph $G = (\mathcal{V}, \mathcal{E})$ is a set $\mathcal{S} \subset \mathcal{V}$ such that no two vertices are adjacent. More precisely, $\forall v \in \mathcal{S}$:

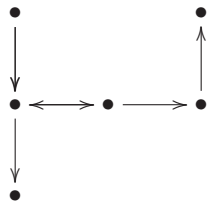
$$(v, w) \in \mathcal{E} \text{ or } (w, v) \in \mathcal{E} \Rightarrow w \notin \mathcal{S}.$$

⌟

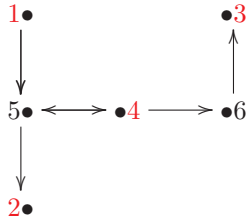
Method 10.16 (**Independent Set Ordering (ISO)**)**:** An *independent set ordering* for a graph $G = (\mathcal{V}, \mathcal{E})$ can be obtained by means of the following algorithm:

1. Initially, put $\mathcal{S} = \varnothing$.
2. While unmarked vertices are available: Choose[2] an unmarked vertex v and add it to $\mathcal{S}$. Mark v and all vertices adjacent to v.
3. Number the vertices that belong to the independent set $\mathcal{S}$ first. Then, number the other vertices. ⌟

Example 10.17 (ISO)**:** Consider the following adjacency graph:



An example of an ISO ordering is:



⌟

Remark 10.18: Observe that, for a linear system that results from the local-theta scheme (Method 6.1), it is possible to obtain an independent set ordering of the form

$$\begin{bmatrix} \underline{\underline{D}} & \\ & \underline{\underline{C}} \end{bmatrix},$$

where the elements of the diagonal matrix $\underline{\underline{D}}$ correspond to the explicit cells ($\theta_{ij}^n = 0$ for each face). As a result, the dimension of the linear system that needs to be solved has been reduced to the dimension of $\underline{\underline{C}}$. For this reason, this type of reordering may lead to a reduction of computational costs, especially when the number of explicit grid cells is high. ⌟

[2] Choose for instance the vertex of lowest degree. Heuristically, this yields a large independent set.

10.2.3 Multicolor orderings

Graph coloring is the process of coloring (labeling) vertices such that adjacent vertices do not have the same color. Moreover, this should be done with the least possible amount of colors.

A multicolor ordering is a color by color ordering after graph coloring has been executed. If k colors are used, k independent sets are obtained. This yields a $k \times k$ block matrix with diagonal matrices on the diagonal: Lovely for parallel computing.

In practice, the smallest possible number of colors to color the graph can rarely be easily determined. Therefore, this criterion is relaxed to obtain the following greedy algorithm.

Method 10.19 (**Multicolor Ordering**)**:** A *multicolor ordering* for a graph $G = (\mathcal{V}, \mathcal{E})$ can be obtained by means of the following greedy algorithm:

1. Let $\mathcal{S}_i$ $(i = 1, ..., n)$ contain the vertices with color i. Initially, none of the vertices is colored, so put $\mathcal{S}_i = \varnothing$ $(i = 1, ..., n)$.
2. While unmarked nodes are available:
   i. Choose an unmarked vertex $v \in \mathcal{V}$.
   ii. Determine the 'smallest' color that none of its neighbors has:
   $$i = \min\{i \in \{1, ..., n\} | \forall w \in \mathcal{V} \text{ adjacent to } v : w \notin \mathcal{S}_i \}.$$
   iii. Color v with color i: $\mathcal{S}_i = \mathcal{S}_i \cup \{v\}$.
   iv. Mark v.
3. First number the vertices of $\mathcal{S}_1$, then the nodes in $\mathcal{S}_2$, and so on. ⌟

10.3 Summary

The costs and the quality of a preconditioner partly depend on the structure of the original matrix. Therefore, it can be a good strategy to reorder the matrix elements in advance. This can be executed in several manners by renumbering the corresponding adjacency graph. Examples of orderings are level set ordering, independent set ordering, and multi-color ordering.

Chapter 11

Storage of sparse matrices

Since sparse matrices contain a large number of zero elements, an efficient way of storing them can save memory as well as computing time. Two popular storing formats are the coordinate format and the compressed sparse row format, which are both discussed in this chapter. Other formats can be found in [DER86, Chapter 2].

11.1 Coordinate format

The most basic format is the coordinate format[1], which only stores the nonzero elements and their row and column index.

Definition 11.1 (**Coordinate format**)**:** The *coordinate format* of a matrix $\underline{\underline{A}} \in \mathbb{R}^{m\times n}$ with k nonzero elements consists of three vectors. $\underline{\mathbf{e}} \in \mathbb{R}^k$ contains the nonzero elements of A, $\underline{\mathbf{r}} \in \mathbb{N}^k$ contains their row indices, and $\underline{\mathbf{c}} \in \mathbb{N}^k$ contains their column indices. The vectors can be filled in any order. If they are filled by row, the vectors are constructed according to:

1. $k = 0$
2. for $i = 1, ..., m$:
3. for $j = 1, ..., n$:
4. if $a_{ij} \neq 0$:
5. $k = k + 1$
6. $e_k = a_{ij}$
7. $r_k = i$
8. $c_k = j$
9. end
10. end
11. end ⌟

Example 11.2 (Coordinate format)**:** Consider the matrix

$$\underline{\underline{A}} = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 3 & 4 & 0 & 5 & 0 \\ 6 & 0 & 7 & 8 & 9 \\ 0 & 0 & 10 & 11 & 0 \\ 0 & 0 & 0 & 0 & 12 \end{bmatrix}. \tag{11.1}$$

The corresponding coordinate format reads:

$$\begin{aligned} \underline{\mathbf{e}} &= [\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ ]^T, \\ \underline{\mathbf{r}} &= [\ 1\ 1\ 2\ 2\ 2\ 3\ 3\ 3\ 3\ 4\ 4\ 5\ ]^T, \\ \underline{\mathbf{c}} &= [\ 1\ 4\ 1\ 2\ 4\ 1\ 3\ 4\ 5\ 3\ 4\ 5\ ]^T. \end{aligned}$$

[1] This format is used by MATLAB

11.2 Compressed sparse row format

One of the most popular formats is the compressed sparse row format, which is comparable to the coordinate format. The difference is that the row indices are stored more efficiently.

Definition 11.3 (**Compressed Sparse Row format (CSR)**)**:** The *compressed sparse row format* of a matrix $\underline{\underline{A}} \in \mathbb{R}^{m \times n}$ with k nonzero elements consists of three vectors. $\underline{\mathbf{e}} \in \mathbb{R}^k$ contains the nonzero elements of A, $\underline{\mathbf{c}} \in \mathbb{N}^k$ contains their column indices, and $\underline{\mathbf{r}} \in \mathbb{N}^{m+1}$ contains the pointers to the beginning of each row in the vectors $\underline{\mathbf{e}}$ and $\underline{\mathbf{c}}$. The vectors are filled by row, so according to:

1. $k = 0$
2. for $i = 1, ..., m$:
3. for $j = 1, ..., n$:
4. if $a_{ij} \neq 0$:
5. $k = k + 1$
6. $e_k = a_{ij}$
7. $c_k = j$
8. if the row pointer is not already set for row i: $r_i = k$
9. end
10. end
11. if the row pointer is not already set for row i: $r_i = k$
12. end
13. $r_{m+1} = k + 1$

Example 11.4 (CSR format)**:** The CSR format for (11.1) reads:

$$\begin{array}{rcl} \underline{\mathbf{e}} & = & [\ 1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10\ \ 11\ \ 12\ \ \ ]^T, \\ \underline{\mathbf{r}} & = & [\ 1\ \ \ \ 3\ \ \ \ 6\ \ \ \ \ \ 10\ \ \ 12\ \ 13\ ]^T, \\ \underline{\mathbf{c}} & = & [\ 1\ \ 4\ \ 1\ \ 2\ \ 4\ \ 1\ \ 3\ \ 4\ \ 5\ \ 3\ \ 4\ \ 5\ \ \ ]^T. \end{array}$$

Method 11.5 (**CSR-vector product**)**:** A matrix stored in CSR format can be multiplied by a vector $\underline{\mathbf{x}}$ as follows:

1. for $i = 1 : n$
2. $k_1 = r_i$
3. $k_2 = r_{i+1} - 1$
4. $(\underline{\mathbf{e}}(k_1 : k_2))^T \underline{\mathbf{x}}(c(k_1 : k_2))$
5. end

Remark 11.6 (Current format in WAQ)**:** The current storage format in WAQ is comparable to the CSR format. The difference is that the diagonal elements are stored separately in a vector $\underline{\mathbf{d}}$. Furthermore, the row pointers point to the end of each row, instead of the beginning.

Some numerical schemes of WAQ treat the horizontal and the vertical direction separately. In that case, the format is slightly different. In the current implementation, for each grid cell $\mathcal{V}_i$, there is exactly one coefficient t_i that represents the relation to grid cell right above it, and exactly one coefficient b_i that represents the relation to the grid cell below it. These two coefficients, that may be zero, are stored in $\underline{\mathbf{e}}$ in front of the (other) nonzero elements of the row.

For example, consider a matrix that corresponds to a three-dimensional $2 \times 2 \times 2$ rectangular grid:

$$\begin{bmatrix} d_1 & h_{12} & h_{13} & & \mathbf{t_1} & & & \\ h_{21} & d_2 & h_{23} & h_{24} & & \mathbf{t_2} & & \\ h_{31} & h_{32} & d_3 & h_{34} & h_{35} & & \mathbf{t_3} & \\ & h_{42} & h_{43} & d_4 & h_{45} & h_{46} & & \mathbf{t_4} \\ \mathbf{b_5} & & h_{53} & h_{54} & d_5 & h_{56} & h_{57} & \\ & \mathbf{b_6} & & h_{64} & h_{65} & d_6 & h_{67} & h_{68} \\ & & \mathbf{b_7} & & h_{75} & h_{76} & d_7 & h_{78} \\ & & & \mathbf{b_8} & & h_{86} & h_{87} & d_8 \end{bmatrix}.$$

This matrix would be stored as follows:

$$\begin{array}{lcccccccccccccccc} \underline{\mathbf{d}} = [& d_1 & & & & d_2 & & & & & & d_8 & & & &]^T, \\ \underline{\mathbf{e}} = [& \mathbf{0} & \mathbf{t_1} & h_{12} & h_{13} & \mathbf{0} & \mathbf{t_2} & h_{21} & h_{23} & h_{24} & \ldots & \mathbf{b_8} & \mathbf{0} & h_{86} & h_{87} &]^T, \\ \underline{\mathbf{c}} = [& 0 & 5 & 2 & 3 & 0 & 6 & 1 & 3 & 4 & \ldots & 4 & 0 & 6 & 7 &]^T, \\ \underline{\mathbf{r}} = [& & & & 5 & & & & & 9 & \ldots & & & & 42 &]^T. \end{array}$$

⌟

11.3 Summary

Since sparse matrices contain a large number of zero elements, an efficient way of storing them can save both memory and computing time. The main idea is to store the nonzero elements only. Additionally, information about their location in the original matrix is stored in some efficient manner. WAQ's current storage format is comparable to the compressed sparse row format.

Chapter 12

Summary

If an implicit FVM is used to solve the water quality model, many large sparse linear systems need to be solved. For such systems, iterative methods are more suitable than direct methods. At present, GMRES is implemented in WAQ. This Krylov method converges faster as the eigenvalues are more clustered.

The convergence speed of an iterative method depends on the spectrum of the matrix. Preconditioning transforms a linear system into an equivalent system that has better spectral properties. At present, WAQ uses symmetric Gauss-Seidel preconditioning, which is costless to construct. Unfortunately, for diffusion dominated problems, this preconditioner does not lead to a satisfactory reduction of the number of iterations. An alternative preconditioning, such as ILU(p), could lead to a much smaller number of iterations. However, construction costs and the costs per iteration should also be taken into account.

The costs and the quality of a preconditioner partly depend on the structure of the original matrix. Therefore, it can be a good strategy to reorder the matrix elements in advance. This can be executed in several manners by renumbering the corresponding adjacency graph. Examples of orderings are level set ordering, independent set ordering, and multi-color ordering.

Since sparse matrices contain a large number of zero elements, an efficient way of storing them can save both memory and computing time. The main idea is to store the nonzero elements only. Additionally, information about their location in the original matrix is stored in some efficient manner. WAQ's current storage format is comparable to the compressed sparse row format.

To answer the question that was asked in the introduction of this thesis (Chapter 1):

> *The convergence speed of the current solver for linear systems be enhanced for diffusion dominated problems by using an alternative combination of iterative solver, preconditioner, reordering strategy, and storage format.*

The optimal combination that leads to the desired computational speed will have to be the result of further investigation...

Appendix A

Current schemes of WAQ

At present, fifteen different numerical schemes can be used in WAQ. These are described briefly below.

Scheme 1 is the explicit first order upwind scheme (Method 4.4).

Scheme 2 is like scheme 1, except that it uses the predictor corrector method for time integration.

Scheme 3 is the explicit Lax-Wendroff scheme (Method 4.11).

Scheme 4 is an Alternation Direction Implicit (ADI) method. It can only be applied in two dimensions on a structured grid. This method calculates two successive timesteps in two different ways, using a semi-implicit scheme. In one time step the derivatives in the y-direction are evaluated explicitly instead of implicitly. In the other time step the derivatives in the x-direction are evaluated explicitly instead of implicitly. This scheme uses the theta scheme (Method 5.1) for $\theta = \frac{1}{2}$. Explicit fluxes are second and third order upwind fluxes. Implicit fluxes are central fluxes (4.3). Since only one direction at the time is implicit, this results in a tridiagonal matrix, which is relatively easy to solve with a direct method.

Scheme 5 is an explicit FCT scheme a la Boris & Book (almost similar to Method 4.8) with Lax-Wendroff flux correction (Method 4.11).

Scheme 10 is the theta upwind scheme (Method 5.1) with $\theta = 1$.

Scheme 11 treats the horizontal and vertical direction seperately. In the horizontal direction, the explicit upwind scheme (scheme 1) is applied. In the vertical direction, the theta scheme (Method 5.1) with $\theta = \frac{1}{2}$ and central fluxes (4.3) is used, possibly with a smoothing Forrester filter. The resulting tridiagonal linear systems are solved by means of a direct method.

Scheme 12 is like scheme 11, except that it uses an explicit FCT scheme (scheme 5) in the horizontal direction.

Scheme 13 is like Scheme 11, except that it uses the theta upwind scheme (Method 5.1) with $\theta = 1$ in the vertical direction.

Scheme 14 is like scheme 12, except that it uses the theta upwind scheme (Method 5.1) with $\theta = 1$ in the vertical direction.

Scheme 15 is like scheme 10, except that, in the horizontal direction, the linear systems are solved by means of GMRES (see Method 8.26) with a symmetric GS preconditioner (see Example 9.5). In the vertical direction, a direct method is used.

Scheme 16 is like Scheme 15, except that it uses the theta scheme (Method 5.1) with $\theta = \frac{1}{2}$ and central discretisation (4.3) in the vertical direction, possibly with a smoothing Forrester filter.

Scheme 19 treats the horizontal and vertical direction seperately. In the horizontal direction, an ADI method (scheme 4) is used. In the vertical direction, central fluxes (4.3) are used, possibly with a smoothing Forrester filter.

Scheme 20 is like scheme 19, except that it uses first order upwind discretisation (4.1) in the vertical direction.

Bibliography

[BB73] J. P. Boris and D. L. Book. Flux corrected transport. 1. shasta, a fluid transport algorithm that works. *Journal of Computational Physics*, 11:38–69, 1973.

[BF01] R.L. Burden and J.D. Faires. *Numerical Analysis*. Brooks/Cole, Pacific Grove, 2001.

[BO04] T. Barth and M. Ohlberger. Finite volume methods: Foundation and analysis. In *Encyclopedia of Computational Mechanics*. John Wiley & Sons, 2004.

[DER86] I.S. Duff, A.M. Erisman, and J.K. Reid. *Direct Methods for Sparse Matrices*. Clarendon Press, Oxford, 1986.

[GL81] A. George and J.W. Liu. *Computer Solution of Large Sparse Positive Definite Systems*. Prentice Hall, Inc., New Jersey, 1981.

[GL96] G.H. Golub and C.F. Van Loan. *Matrix computations*. The Johns Hopkins University Press, Baltimore and London, third edition, 1996.

[God96] E. Godlewski. *Numerical Approximation of Hyperbolic Systems of Conservation Laws*. Springer, New York, 1996.

[Jam93] A. Jameson. Computational algorithms for aerodynamic analysis and design. *Applied Numerical Mathematics*, 13:383–422, 1993.

[KMT] D. Kuzmin, M. Möller, and S. Turek. High-resolution fem-fct schemes for multidimensional conservation laws. Technical Report 231, Institute of Applied Mathematics (LS III), University of Dortmund.

[Krö97] D. Kröner. *Numerical Schemes for Conservation Laws*. John Wiley & Sons Ltd, West Sussex and B.G. Teubner, Stuttgart, 1997.

[KT] D. Kuzmin and S. Turek. High-resolution fem-tvd schemes based on a fully multidimensional flux limiter. Technical Report 229, Institute of Applied Mathematics (LS III), University of Dortmund.

[KT02] D. Kuzmin and S. Turek. Flux correction tools for finite elements. *Journal of Computational Physics*, 175:525–558, 2002.

[LeV02] R.J. LeVeque. *Finitie Volume Methods for Hyperbolic Problems*. Cambridge University Press, New York, 2002.

[man05] *Delft3D-WAQ; Versatile water quality modeling in 1D, 2D or 3D systems including physical, (bio)chemical and biological processes*, November 2005.

[Pos05] L. Postma. Water quality of surface waters. Technical report, WL Delft Hydraulics, January 2005.

[Saa00] Y. Saad. Iterative methods for sparse linear systems. This is a revised version of the book published in 1996 by PWS Publishing, Boston. It can be downloaded from http://www-users.cs.umn.edu/~saad/books.html, 2000.

[SS86] Y. Saad and M.H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7:856–869, 1986.

[Wes01] P. Wesseling. *Principles of computational fluid dynamics*. Springer, Berlin, 2001.

[Zal79] S.T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31:335–362, 1979.

Index

adjacency graph, 76
adjacent, 76
advection, 5
Arnoldi, 64
Arnoldi-Modified Gram-Schmidt, 64

backward Gauss-Seidel, 62
backward substitution, 59

cell centered grid, 9
Chebychev polynomial, 67
compressed sparse row (CSR) format, 82
consistency, 15
convergence of
    FVM, 14
    GMRES, 67
    iterative method, 61
    linear fixed point iteration, 62
coordinate format, 81
Courant-Friedrichs-Lewy (CFL) condition, 21
Cuthill-McKee (CMK) ordering, 77

degree, 77
Dirichlet boundary condition, 6

explicit FCT scheme, 23
explicit scheme, 21
explicit upwind FCT scheme, 23
explicit upwind scheme, 21

finite volume method (FVM), 13
flops, 59
forward substitution, 59

Gauss-Jacobi, 62
Gauss-Seidel, 62
Gaussian elimination, 60
general minimal residual (GMRES) method, 66
Givens rotations, 65
global discrete maximum principle, 16
global truncation error, 14
graph, 76

Hessenberg matrix, 65

ILU, 72
ILU(0), 72
ILU(p), 73
ILUT, 71
ILUT(p,τ), 71
incomplete LU factorisation, 71
independent set, 78
independent set ordering (ISO), 78
interchange matrix, 75
iterative method, 61

Krylov method, 63
Krylov space, 63

Lax-Wendroff, 25
left preconditioning, 69
level set, 77
level set ordering, 77
linear fixed point iteration, 61
local discrete maximum principle, 16
local extremum diminishing (LED), 16
local truncation error, 15
local-theta FCT scheme, 44
local-theta scheme, 39
local-theta upwind FCT scheme, 44
local-theta upwind scheme, 39
LU factorisation, 60

mass conservative, 14
matrix splitting, 61
modified equation, 30
modified ILU (MILU), 74
molecular diffusion, 5
monotonicity preserving, 16
multicolor ordering, 79

Neumann boundary condition, 6
non-oscillatory, 17
numerical flux function, 13

order of accuracy, 15

permutation matrix, 75
positivity preserving, 15
preconditioning, 69

right preconditioning, 69

robust, 15

sigma grid, 10
stability (absolute), 15
successive over-relaxation (SOR), 62
symmetric Gauss-Seidel, 63
symmetric permutation, 75

theta FCT scheme, 32
theta Lax-Wendroff scheme, 36
theta scheme, 29
theta upwind FCT scheme, 32
theta upwind scheme, 29
turbulent diffusion, 5
turbulent mixing, 5

z-grid, 10