ANALYSIS OF COUPLED GRID METHODS FOR SOLVING CONVECTION-DIFFUSION EQUATIONS

HOW TO PRESERVE SECOND ORDER ACCURACY

by

W.H.C. Janssen

in partial fulfillment of the requirements for the degree of

Bachelor of Science
in Applied Physics & Applied Mathematics

at the Delft University of Technology,
to be defended publicly on Friday July 3, 2015 at 2:30 PM.

Supervisors:	Prof. dr. ir. J.J. Derksen,	TU Delft
	Dr. ir. D.R. van der Heul,	TU Delft
Thesis committee:	Prof. dr. R.F. Mudde,	TU Delft
	Dr. M.C. Veraar,	TU Delft
	Drs. E.M. van Elderen,	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft Delft University of Technology

ABSTRACT

The error behavior of finite difference discretization schemes has been researched. Starting from the one-dimensional case, where two particles of a fluidized bed were taken, the diffusion equation was numerically solved on a single grid as reference. Then the diffusion equation, a model equation for the real problem, was solved on a separated grids setup, where the grid spacing of the grid adjacent to the particles is relatively fine and the other grid relatively coarse. The main problem is the communication/coupling between these two grids. The research question of this report is: Is it possible to couple the two grids in a way that the numerical solution on the coupled grids has the same order of error behavior as the numerical solution on a single grid? If this is possible what are the minimal conditions for the coupling strategy? Different strategies of this coupling are proposed. The order of the error of the numerical solution was first estimated by Richarson error estimation and then computed by the means of the L_2-norm, where the numerical solution is compared with the analytical solution. Coupling by using an interpolation polynomial of at least second order in both directions resulted in an error behavior of second order of the numerical solution, which is the same error behavior as the numerical solution on a single grid. Hereafter the problem is extended to two dimensions where also a convective term is taken into account. The results of the coupling from the one-dimensional problem were used for the two dimensional problem. For the two-dimensional problem the L_2-error could also be determined. The conclusion is the same as for the one dimensional case: At least a second order interpolation polynomial is required to get second order error behavior.

W.H.C. Janssen
Delft, June 2015

CONTENTS

1. INTRODUCTION

A fluidized bed is an arrangement in which lots of the processes, fluid catalytic cracking as an example, of the process industry take place. The concept of a fluidized bed is a lot of tiny solid particles flowing in a stream of some gas or liquid. These beds are very useful in many cases where it is important to create a large surface area between the solid and liquid(/gas).

This report will start from a research to fluidized beds. The main goal of that research is to solve a concentration profile, that depends on time, in the fluidized bed with computer simulations. With the results of those simulations the behavior and the properties of a fluidezed bed should be better understood. In an earlier research[1] the concentration profile around a single particle (of the fluidized bed) was considered for laminar and turbulent flows. In another paper[2] the research is extended to at first two identical particles, treated as perfect spheres, and after that multiple particle systems.

The solution of the problems above involve a technique called COD (Coupled Overlapping Domains). The COD technique means that a fine spherical grid is surrounding the particles and a Cartesian, relatively coarse grid, is more at distance. The relatively fine grid will solve for distances close to the particles and a coarse grid will solve for the rest of the bed. The usage of two separated grids is explained by the fact the problem involves high Schmidt numbers, kinematic viscosity ν over mass diffusivity D. High Schmidt numbers in combination with a Dirichlet boundary condition means strong gradients, which asks for a fine grid to be accurately solved. The use of only the fine grid would cause excessive computation time. A local grid refinement would not work; the flow dynamics can be fully dealt with on a Cartesian grid, but the particles are spherical.

In [1] simulations are done for the passive scalar transfer from spherical particle to the liquid the particle is immersed in. The COD technique, with linear interpolation in both directions, is used and has correct convergence behavior; higher levels of accuracy on finer grids. The performance of the method does not depend on whether the sphere is moving or if it is fixed. Actually there are some imperfections between the two grids. It is suggested that in a future research the accuracy could be improved by using other interpolation schemes.

The primary aim of [2] is to extend the COD technique so it properly works when the spherical grids of two particles overlap. This means the technique is not to cumbersome to code and a problem using COD is still computational solvable. This is done by a simple mixing the concentrations of the overlapping spherical grids and the Cartesian background grid. Furthermore the introduced error by using the COD technique, even in the case when spherical grids do not overlap each other, must still be mathematically validated.

This report will not deal with the overlapping spherical grids, but will investigate the error introduced by the COD technique. The research question in this report is: Is it possible to couple the two grids in a way that the numerical solution on the coupled grids has the same order of error behavior as the numerical solution on a single grid? If it is possible what are the minimal conditions for the coupling strategy?

To answer this question three coupling strategies will be suggested: A separated grids scheme where extrapolation is used to couple both grids, an overlapping grid scheme where simple linear interpolation is used to couple the grids and an overlapping grid scheme where higher order interpolation polynomials are used to couple the grids. To investigate these coupling strategies, first a one-dimensional problem will be discussed. With the results of the coupling strategies more complicated problems in two dimensions are researched.

2. THE UNDERLYING PHYSICS

Mass is a conserved quantity. So inside a specified volume, the control volume, the change of the concentration of a certain substance is equal to the inflow of that substance minus the outflow of that substance plus the production of that substance. The Reynolds transport equation[5, p.21] describes this change of mass inside a control volume, see figure 1.

$$\frac{\partial}{\partial t}\iiint_R c\,\partial V = -\oiint \boldsymbol{\phi} \bullet \hat{n}\,\partial S + \iiint_R Q\,\partial V \tag{1}$$

Here c is the concentration of a certain substance, $\boldsymbol{\phi}$ a vector with the mass flux, $\hat{n}$ the outward pointing normal vector and Q the production term.

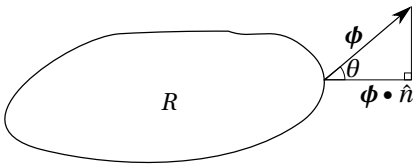


Figure 1: Control volume

Assume that no mass of the substance is created inside the control volume, so the production term $Q = 0$. Equation (1) then reduces to:

$$\frac{\partial}{\partial t}\iiint_R c\,\partial V = -\oiint \boldsymbol{\phi} \bullet \hat{n}\,\partial S \tag{2}$$

Now the divergence theorem[5, p.22] can be used. The divergence theorem states that the volume integral of the divergence of any continuously differentiable vector is the closed surface integral of the outward pointing normal component of that vector. For our case this means:

$$\oiint \boldsymbol{\phi} \bullet \hat{n}\,\partial S = \iiint_R \nabla \bullet \boldsymbol{\phi}\,\partial V \tag{3}$$

And equation (2) can be written as:

$$\frac{\partial}{\partial t}\iiint_R c\,\partial V = -\iiint_R \nabla \bullet \boldsymbol{\phi}\,\partial V \tag{4}$$

Note that the time derivative can be put inside the integral since the R is fixed. If then the right side of equation (4) is taken to the left side:

$$\iiint_R \left[\frac{\partial c}{\partial t} + \nabla \bullet \boldsymbol{\phi}\right]\partial V = 0 \tag{5}$$

It follows that:

$$\frac{\partial c}{\partial t} = -\nabla \bullet \boldsymbol{\phi} \tag{6}$$

Mass transfer can happen by convective transport (convection) and diffusive transport (diffusion). In the case of convective transport the motion of the molecules is driven by the fluid or gas they are in. In the case of diffusive transport the motion of the molecules is driven by concentration differences. According to Fick's law[4, p.72] and the Fourier's law in 3 dimensions[5, p.30] it can be concluded that Fick's law in 3 dimensions is given by:

$$\boldsymbol{\phi}_{\text{diff}} = -D\nabla c \tag{7}$$

Where D is the diffusion coefficient, which depends on the solute and on the solvent. For convective transport it applies that:

$$\boldsymbol{\phi}_{\text{conv}} = \boldsymbol{v}c \tag{8}$$

Where $\boldsymbol{v}$ is the velocity vector. Assume that convective and diffusive transport take place. Then $\boldsymbol{\phi}$=$\boldsymbol{\phi}_{\text{diff}}+\boldsymbol{\phi}_{\text{conv}}$ can be substituted in equation (6) and the following equation is obtained:

$$\frac{\partial c}{\partial t} = -\nabla \bullet (-D\nabla c + \boldsymbol{v}c) = D\nabla^2 c - \boldsymbol{v}\nabla c \tag{9}$$

The second equals sign only holds if D does not depend on location and $\nabla \bullet \boldsymbol{v} = 0$ (which means that the flow is incompressible). Equation (9) is called the convection-diffusion equation for three dimensions. For one dimension it looks like:

$$\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial x^2} - v_x\frac{\partial c}{\partial x} \tag{10}$$

It is possible to make this equation dimensionless. First define the scale for length as L and the scale for time as $\frac{D}{L^2}$. Now the dimensionless variables $\tilde{x}$ and $\tilde{t}$ can be defined as:

$$\tilde{x} = \frac{x}{L}, \quad \tilde{t} = \frac{Dt}{L^2} \tag{11}$$

$$\Rightarrow x = \tilde{x}L, \quad t = \frac{\tilde{t}L^2}{D} \tag{12}$$

Substitution of (12) in equation (10) gives:

$$\frac{D}{L^2}\frac{\partial c}{\partial \tilde{t}} = \frac{D}{L^2}\frac{\partial^2 c}{\partial \tilde{x}^2} - \frac{v_x}{L}\frac{\partial c}{\partial \tilde{x}} \tag{13}$$

If both sides are divided by $\frac{D}{L^2}$ the result is:

$$\frac{\partial c}{\partial \tilde{t}} = \frac{\partial^2 c}{\partial \tilde{x}^2} - \frac{v_x L}{D}\frac{\partial c}{\partial \tilde{x}} = \frac{\partial^2 c}{\partial \tilde{x}^2} - \text{Pe}\frac{\partial c}{\partial \tilde{x}} \tag{14}$$

Here Pe is another dimensionless number, called the Péclet number, defined as:

$$\text{Pe} = \frac{v_x L}{D} \tag{15}$$

The Péclet number represents the relationship between convective and diffusive transport. The nondimensionalization process reduces the amount of variables. Equation (14) is more general than equation (10), if the Péclet number is given the whole system is specified.

3. THE 1D PROBLEM

3.1. SETUP AND DISCRETIZATION

In [2] two particles are taken from a fluidized bed, see figure 2. These particles are considered to be perfect spheres with radius a, on a fixed location, a distance of $2s$ apart and the particles are on a concentration $c = c_0$. This problem can be reduced even further to one dimension if the assumption $a \gg 2s$ is made, so the particles can be considered as flat plates. In this one-dimensional problem it will be easier to investigate the mathematical problems that arise for solving the convection-diffusion equation than directly test them on the two-dimensional setup.

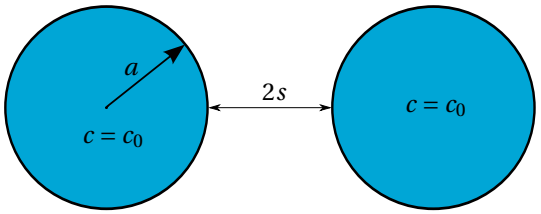


Figure 2: Two particles of the fluidized bed

Now two opposing points are taken one the plates. These points are a distance $2s$ apart both hold a concentration of c_0 while the liquid between the 'plates' is everywhere equal to $c = 0$ at $t_0 = 0$. Because this placing is completely symmetrical it is possible to simplify even further by placing one point at $x = s$ and keep the derivative at $x = 0$ equal to zero: $\frac{\partial c(0,t)}{\partial t} = 0$ (see figure 3). For the moment the fluid does not move, so there is only diffusive transport between the points. Combining the diffusion equation in one dimension, equation (10) with $v_x = 0$, with the boundary conditions from figure 3 the whole problem can be formulated as:

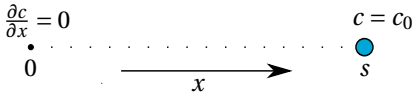


Figure 3: The setup

$$\begin{aligned} &\frac{\partial c(x,t)}{\partial t} = D\frac{\partial^2 c(x,t)}{\partial x^2},\ 0 < x < s,\ \forall\, 0 < t \\ &c(s,t) = c_0 \text{ and } \frac{\partial c(0,t)}{\partial x} = 0,\ \forall\, 0 < t \\ &c(x,0) = 0,\ \forall\, x \neq s \end{aligned} \tag{16}$$

D will be a constant in this report. The analytical solution of (16) is given by:

$$c(x,t) = c_0 + \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi}{2s}(x+s)\right) e^{-D\left(\frac{n\pi}{2s}\right)^2 t} \tag{17}$$

Where the B_n are given by:

$$B_n = \begin{cases} -\frac{4c_0}{n\pi} & \text{if } n \text{ is odd} \\ 0 & \text{else} \end{cases} \tag{18}$$

The derivation of this solution is given in the appendix. Although we now got an exact solution of (16) it still will be very useful to solve (16) numerically. The problems in two dimensions, let alone the whole fluidized bed, will not be analytically solvable. Numerical mathematics has the machinery to approximate (16) and the other problems of this report numerically, which means an approximation of the exact solution will be found. The numerical solutions can be compared with the exact solution in order to find the error between them and thus the accuracy of the numerical solution.

Where the analytical approach results in a continuous solution, the numerical solution will only be defined in a discrete number of points, namely $0, h, 2h, \ldots, s-h$. The discrete points $0, h, 2h, \ldots, s-h$ are called a grid. Here $0 < h$ is called the grid spacing, on every multiple of h the concentration is approximated. The solution will not be calculated in s, because in s the solution is prescribed: $c = c_0$. By using a technique called semi-discretization[3, p.118] (16) can be rewritten as:

$$\frac{\partial \boldsymbol{c}}{\partial t} = \boldsymbol{A}\boldsymbol{c} + \boldsymbol{b},\ \forall\, 0 < t \tag{19}$$

Here $\boldsymbol{c}$ is a vector that holds the concentrations in the points of the grid. In the rest of this report all the vectors

and matrices are represented by bold symbols. $\boldsymbol{A}$ in equation (19) is a matrix and $\boldsymbol{b}$ a vector given by:

$$\boldsymbol{A} = \frac{D}{h^2}\begin{bmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \emptyset \\ & \ddots & \ddots & \ddots & \\ \emptyset & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}, \quad \boldsymbol{b} = \frac{D}{h^2}\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ c_0 \end{pmatrix} \tag{20}$$

The vector $\boldsymbol{b}$ accounts for the boundary value in s. The entries of $\boldsymbol{A}$ were found by so called second order central difference of

$$\left.\frac{\partial^2 c(x,t)}{\partial x^2}\right|_{x=ih} = \frac{\boldsymbol{c}_{i-1} - 2\boldsymbol{c}_i + \boldsymbol{c}_{i+1}}{h^2} + \mathcal{O}(h^2) \tag{21}$$

Where i is an integer and $\boldsymbol{c}_i$ is the concentration in ih. The actual error made in this approximation, $\mathcal{O}(h^2)$, can be determined by expanding $\boldsymbol{c}_{i-1}$ and $\boldsymbol{c}_{i+1}$ into their Taylor series.

$$\boldsymbol{c}_{i-1} = \boldsymbol{c}_i - h\boldsymbol{c}_i' + \frac{h^2}{2}\boldsymbol{c}_i'' - \frac{h^3}{6}\boldsymbol{c}_i''' + \mathcal{O}(h^4) \tag{22}$$

$$\boldsymbol{c}_{i+1} = \boldsymbol{c}_i + h\boldsymbol{c}_i' + \frac{h^2}{2}\boldsymbol{c}_i'' + \frac{h^3}{6}\boldsymbol{c}_i''' + \mathcal{O}(h^4) \tag{23}$$

Substituting these expressions in the difference formula gives:

$$\frac{\boldsymbol{c}_i - h\boldsymbol{c}_i' + \frac{h^2}{2}\boldsymbol{c}_i'' - \frac{h^3}{6}\boldsymbol{c}_i''' + \mathcal{O}(h^4) - 2\boldsymbol{c}_i + \boldsymbol{c}_i + h\boldsymbol{c}_i' + \frac{h^2}{2}\boldsymbol{c}_i'' + \frac{h^3}{6}\boldsymbol{c}_i''' + \mathcal{O}(h^4)}{h^2} = \frac{h^2\boldsymbol{c}_i'' + \mathcal{O}(h^4)}{h^2} = \boldsymbol{c}_i'' + \mathcal{O}(h^2) \tag{24}$$

In a similar way it can be proved that the first derivative can be approximated with an error of $\mathcal{O}(h^2)$ by the following difference formula:

$$\left.\frac{\partial c(x,t)}{\partial x}\right|_{x=ih} = \frac{\boldsymbol{c}_{i+1} - \boldsymbol{c}_{i-1}}{2h} = \frac{\boldsymbol{c}_i + h\boldsymbol{c}_i' + \frac{h^2}{2}\boldsymbol{c}_i'' + \mathcal{O}(h^3) - \boldsymbol{c}_i + h\boldsymbol{c}_i' - \frac{h^2}{2}\boldsymbol{c}_i'' + \mathcal{O}(h^3)}{2h} = \frac{2h\boldsymbol{c}_i' + \mathcal{O}(h^3)}{2h} = \boldsymbol{c}_i' + \mathcal{O}(h^2) \tag{25}$$

With the aid of this difference formula it can be explained how the first row of $\boldsymbol{A}$ deals with the Neumann boundary condition at $x = 0$. A second order approximation for the boundary condition is given by:

$$\begin{cases} \dfrac{\partial c(0,t)}{\partial x} \approx \dfrac{\boldsymbol{c}_{i+1} - \boldsymbol{c}_{i-1}}{2h} \\ \dfrac{\partial c(0,t)}{\partial x} = 0 \end{cases} \tag{26}$$
$$\Leftrightarrow \boldsymbol{c}_{i-1} = \boldsymbol{c}_{i+1}$$

Substitution of this result in (21) for $x = 0$ explains the entries of the first row of $\boldsymbol{A}$.

3.2. Methods of numerical integration

Reconsider equation (19). The left side of this equation could also be discretized, but now in time:

$$\frac{\boldsymbol{c}^{m+1} - \boldsymbol{c}^m}{\delta t} = \boldsymbol{A}\boldsymbol{c}^m + \boldsymbol{b} \tag{27}$$

$$\boldsymbol{c}^{m+1} = \boldsymbol{c}^m + \delta t\left(\boldsymbol{A}\boldsymbol{c}^m + \boldsymbol{b}\right) \tag{28}$$

So with the concentration vector at time m, $\boldsymbol{c}^m$, the concentration vector at time $m+1$, $\boldsymbol{c}^{m+1}$, can be calculated. δt is called the integration time step. Equation (28) is also called the method of Euler Forward, which is a numerical integration method. Two other methods of integration are Modified Euler and the classical 4^{th} order Runge-Kutta. The method of Modified Euler for matrices is given by[3, p.68]

$$\overline{\boldsymbol{c}^{m+1}} = \boldsymbol{c}^m + \delta t\left(\boldsymbol{A}\cdot\boldsymbol{c}^m + \boldsymbol{b}\right) \tag{29}$$

$$\boldsymbol{c}^{m+1} = \boldsymbol{c}^m + \frac{\delta t}{2}\left[\left(\boldsymbol{A}\cdot\boldsymbol{c}^m + \boldsymbol{b}\right) + \left(\boldsymbol{A}\cdot\overline{\boldsymbol{c}^{m+1}} + \boldsymbol{b}\right)\right] \tag{30}$$

The classical 4$^{\text{th}}$ order Runge-Kutta method for matrices is given by[3, p.76]:

$$\boldsymbol{k}_1 = \delta t\left(\boldsymbol{A}\cdot\boldsymbol{c}^m + \boldsymbol{b}\right) \tag{31}$$
$$\boldsymbol{k}_2 = \delta t\left(\boldsymbol{A}\cdot\left(\boldsymbol{c}^m + 0.5\cdot\boldsymbol{k}_1\right) + \boldsymbol{b}\right) \tag{32}$$
$$\boldsymbol{k}_3 = \delta t\left(\boldsymbol{A}\cdot\left(\boldsymbol{c}^m + 0.5\cdot\boldsymbol{k}_2\right) + \boldsymbol{b}\right) \tag{33}$$
$$\boldsymbol{k}_4 = \delta t\left(\boldsymbol{A}\cdot\left(\boldsymbol{c}^m + \boldsymbol{k}_3\right) + \boldsymbol{b}\right) \tag{34}$$
$$\boldsymbol{c}^{m+1} = \boldsymbol{c}^m + \frac{1}{6}\left(\boldsymbol{k}_1 + 2\cdot\boldsymbol{k}_2 + 2\cdot\boldsymbol{k}_3 + \boldsymbol{k}_4\right) \tag{35}$$

Like earlier stated the numerical solutions are approximations of analytical solutions, so there is a certain error between the exact analytical solution and the numerical solution . For the method of Euler Forward this error is $\mathcal{O}(\delta t)$, the method of Modified Euler has an error of $\mathcal{O}(\delta t^2)$ and like it suggests 4$^{\text{th}}$ order Runge-Kutta has an error of $\mathcal{O}(\delta t^4)$[3, p.76].

When only real eigenvalues, λ_i, of $\boldsymbol{A}$ are considered, which is the case in this article, the stability condition for Euler Forward and Modified Euler is the same: $\delta t < \frac{2}{\lambda}$. Where λ is defined by $\lambda = \max_i\{|\lambda_i|\}$. For 4$^{\text{th}}$ order Runge-Kutta the stability condition is given by: $\delta t < \frac{2.8}{\lambda}$. See [3, p.70] for the derivation. If the stability condition is obeyed a round-off error is not magnified. Such methods are called numerically stable. Table 1 gives an overview of the properties of the three integration methods[3, p.78].

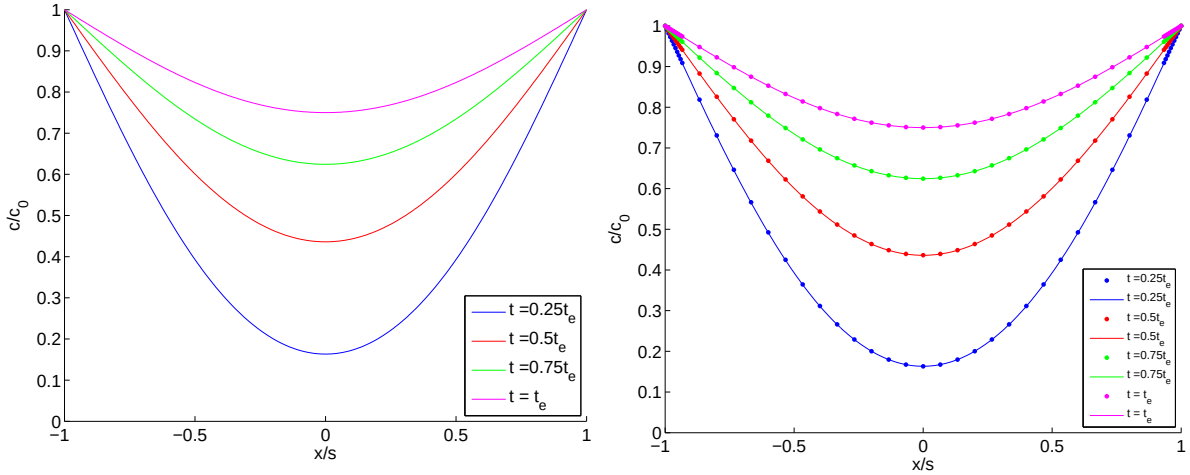
In this report Modified Euler is used, because of its error of $\mathcal{O}(\delta t^2)$ and the only 2 function evaluations per cycle.

Method	Stability condition	Function evaluations per cycle	Error
Euler Forward	$\delta t < \frac{2}{\lambda}$	1	$\mathcal{O}(\delta t)$
Modified Euler	$\delta t < \frac{2}{\lambda}$	2	$\mathcal{O}(\delta t^2)$
4th order Runge-Kutta	$\delta t < \frac{2.8}{\lambda}$	4	$\mathcal{O}(\delta t^4)$

Table 1: The stability condition, number of function evaluations per cycle and the order of the error of the three integration methods.

3.3. Integration with a single grid

The initial condition is $c(x,0) = 0,\ \forall\ x \neq s$, so $\boldsymbol{c}^0$ is a vector with all its entries equal to zero. s is set to a value of 1.5 (from [2]) and D to a value of 0.01 (arbitrary, but small so the stability condition is easier satisfied). These values of s and D apply for the rest of the report. Then the method of Modified Euler was used to integrate with $h = 0.01$ and $\delta t = 0.005$. These values satisfy the stability condition of Modified Euler: $\lambda < 400$ for $h = 0.01$ so $\delta t = 0.005 = \frac{2}{400} < \frac{2}{\lambda}$. Figure 4a shows the numerical solution at various intermediate times.



(a) Concentration plot at various times with $h = 0.01$, $\delta t = 0.005$, $s = 1.5$, $D = 0.01$ and $t_e \approx 0.6598\frac{s^2}{D}$

(b) Concentration plots with separated grids (dots): $h_\chi = 0.1$, $h_r = 0.01$. $\delta t = 0.005$, and $t_e \approx 0.6598\frac{s^2}{D}$

Figure 4: 4a shows the numerical on a single grid, 4b shows the numerical solution for the two separated grids.

3.4. INTEGRATION WITH SEPARATED GRIDS

In section 3.3 the integration is done on a single grid. As mentioned in the introduction, the problem involves high Schmidt numbers, which is the motivation to apply a fine grid close to the particles. Like in section 3.1 this problem can be simplified to one dimension. In this section the first suggestion is to solve the problem with completely separated grids.

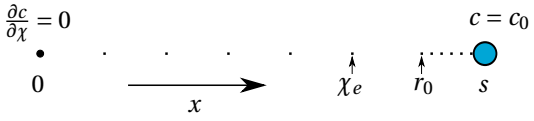


Figure 5: The setup

So there will be two grids both with two boundary conditions, see figure (5). Notice that $\chi_e < r_0$. The coarse grid is defined as the χ-grid and the fine grid as the r-grid. The continuous problem description on the χ-grid is then given by:

$$\begin{aligned} \frac{\partial c_\chi(\chi,t)}{\partial t} &= D\frac{\partial^2 c_\chi(\chi,t)}{\partial \chi^2},\ 0<\chi<\chi_e,\ \forall\, 0<t \\ c(\chi_e,t) &= f(r) \text{ and } \frac{\partial c(0,t)}{\partial \chi} = 0,\ \forall\, 0<t \\ c(\chi,0) &= 0,\ 0<\chi<\chi_e \end{aligned} \tag{36}$$

Here $f(r)$ is a function of the r-grid and the concentrations in the r-grid. The continuous problem description of the r-grid is given by:

$$\begin{aligned} \frac{\partial c_r(r,t)}{\partial t} &= D\frac{\partial^2 c_r(r,t)}{\partial r^2},\ r_0<r<s,\ \forall\, 0<t \\ c(r_0,t) &= g(\chi) \text{ and } c(s,t) = c_0,\ \forall\, 0<t \\ c(r,0) &= 0,\ \forall\, r \neq s \end{aligned} \tag{37}$$

Here $g(\chi)$ is a function of the χ-grid and the concentrations in the χ-grid. The functions $f(r)$ and $g(\chi)$ couple the concentrations from one grid to the other.

To make (36) and (37) suitable to be solved by numerical integration the problem needs to be discretized again. First the two grid at which the problem is solved on are further specified. The χ grid is defined by $[0, h_\chi, \ldots, \chi_e - h_\chi, \chi_e]$ and the r-grid by $[r_0, r_0 + h_r, \ldots, s - 2h_r, s - h_r]$ where h_χ and h_r are the grid spacings. The discretization won't be any different than in section 3.1 and the discretized versions of (36) and (37) are similar to (19):

$$\frac{\partial \boldsymbol{c}_\chi}{\partial t} = \boldsymbol{A}_\chi \boldsymbol{c}_\chi + \boldsymbol{b}_\chi,\ \forall\, 0<t \tag{38}$$

$$\frac{\partial \boldsymbol{c}_r}{\partial t} = \boldsymbol{A}_r \boldsymbol{c}_r + \boldsymbol{b}_r,\ \forall\, 0<t \tag{39}$$

Here $\boldsymbol{c}_\chi$ is the vector that holds the concentrations on the χ-grid and $\boldsymbol{c}_r$ the vector that hold the concentrations on the r-grid. Furthermore extra notation is required for convenience: $\boldsymbol{c}_{\chi_j}^m$ represent the j^{th} element of the vector $\boldsymbol{c}_\chi^m$ where the m stands for the number of time steps that has been done creating the vector. The vectors $\boldsymbol{c}_\chi$ and $\boldsymbol{c}_r$ can only be solved if the boundary values of the χ-grid and the r-grid are known and thus they need to be solved together. For now assume that the $h_\chi = r_0 - \chi_e$ and thus the point r_0 holds the boundary value for the χ-grid. The boundary value for the r-grid is in $x = r_0 - h_r$. The problem is that this point is not a point in the χ-grid. Extrapolation is needed to find an approximation of the concentration in $x = r_0 - h_r$. A Lagrange polynomial[3, p.14] over some of the entries of the vector $\boldsymbol{c}_\chi$ is used to extrapolate the concentration value at the $x = r_0 - h_r$. The n^{th} order Lagrange polynomial for extrapolation to $x = r_0 - h_r$ is given by:

$$\mathscr{L}_n(r_0 - h_r) = \sum_{j=1}^{n} \boldsymbol{c}_{\chi_j} \left(\prod_{\substack{j \le l \le n \\ l \neq j}} \frac{(r_0 - h_r) - \chi_l}{\chi_j - \chi_l} \right) \tag{40}$$

The start value of j in the summation of equation (40) can be chosen $1 \le j \le n$; when the start value is chosen to be $j = 1$ the summation is over all entries of $\boldsymbol{c}_\chi$ and when $1 < j \le n$ some entries, the entries belonging to points closest to $\chi = 0$, are left out. In this case the start value is chosen to be $j = n-3$, so only 4 points are taken

into account. This choice is explained by the fact that a n^{th} order polynomial can possibly have $n-2$ inflection points, points where the second derivative is equal to zero. An high order Lagrange polynomial could results in a large number of inflection points, which means a bad approximation away from the points used for the Lagrange polynomial.

At $t = 0$ the entries of $\boldsymbol{c}_\chi^0$ and $\boldsymbol{c}_r^0$ are all zero due to the initial conditions. The first integration cycle goes as follows. At first $\boldsymbol{c}_r^0$ is integrated to get $\boldsymbol{c}_r^1$. The entry of the vector that represents the concentration at r_0 is added to the boundary vector $\boldsymbol{b}_\chi$. Then $\boldsymbol{c}_\chi^0$ can be integrated to get $\boldsymbol{c}_\chi^1$. Then $\mathscr{L}_4(r_0 - h_r)$ is used to extrapolated a concentration in $x = r_0 - h_r$. This concentration is added to the boundary vector $\boldsymbol{b}_r$. In the next integration cycle this new boundary vector is used to integrate $\boldsymbol{c}_r^1$.

Now the problem is solved on the coupled separated grid setup as described above with $r_0 = 1.4$ (this value applies for the rest of this report unless stated otherwise), $h_\chi = 0.1$ and $h_r = 0.01$. Now it is possible to compare the solution of the coupled separated grids with the solution of the single grid with $h = 0.01$. Figure 4b shows the solution for the separated grids method compared to the single grid integration.

Although it looks like the dots exactly match the continuous line curves (the same curves as in figure 4a), there is some offset which is probably caused by an error in the Lagrange extrapolation and the fact that boundary conditions are coupled at the end of the integration steps. That actually means the boundary values used for the coupling are a certain fraction of an integration step ahead in time. In the next sections some mathematics will quantify this error exactly.

3.5. Richardson error estimation

One technique that is often used for error estimation for numerical problems is based on the Richardson extrapolation formulas[3, p.34,35]. Suppose $N(h,\delta t)$ is used to estimate an unknown value M. The difference between M and $N(h,\delta t)$ can be written as:

$$M - N(h,\delta t) = K_1 h^{\alpha_1} + K_2 \delta t^{\beta_1} + K_3 h^{\alpha_2} + K_4 \delta t^{\beta_2} + \ldots \tag{41}$$

Where $K_i \in \mathbb{R} \setminus \{0\}$ and for $\alpha_i, \beta_i \in \mathbb{N}$ where $0 < \alpha_1 < \alpha_2 < \ldots$ and $0 < \beta_1 < \beta_2 < \ldots$

Under the assumption that h and δt are small the higher order terms in equation (41) can be ignored and thus equation (41) reduces to:

$$M - N(h,\delta t) \approx K_1 h^{\alpha_1} + K_2 \delta t^{\beta_1} \tag{42}$$

The basic idea is to determine $N(h,\delta t)$,$N(2h,\delta t)$ and $N(4h,\delta t)$ for a certain value of h.

$$M - N(h,\delta t) \approx K_1 h^{\alpha_1} + K_2 \delta t^{\beta_1} \tag{43}$$

$$M - N(2h,\delta t) \approx K_1 (2h)^{\alpha_1} + K_2 \delta t^{\beta_1} \tag{44}$$

$$M - N(4h,\delta t) \approx K_1 (4h)^{\alpha_1} + K_2 \delta t^{\beta_1} \tag{45}$$

Subtracting (44) from (45) gives (46) and subtraction of (43) from (44) gives (47):

$$N(2h,\delta t) - N(4h,\delta t) \approx K_1 (4h)^{\alpha_1} \left(1 - \left(\frac{1}{2}\right)^{\alpha_1}\right) \tag{46}$$

$$N(h,\delta t) - N(2h,\delta t) \approx K_1 (2h)^{\alpha_1} \left(1 - \left(\frac{1}{2}\right)^{\alpha_1}\right) \tag{47}$$

Division of equation (46) by equation (47) gives:

$$\frac{N(2h,\delta t) - N(4h,\delta t)}{N(h,\delta t) - N(2h,\delta t)} \approx 2^{\alpha_1} \tag{48}$$

Following a same reasoning for δt results in:

$$\frac{N(h,2\delta t) - N(h,4\delta t)}{N(h,\delta t) - N(h,2\delta t)} \approx 2^{\beta_1} \tag{49}$$

Equations (48) and (49) can now be used to determine α_1 and β_1, the order of the error in h and δt respectively.

With this technique it is possible to check that the orders stated for Modified Euler in Section 3.2 are correct. $N(h,\delta t)$, $N(2h,\delta t)$ and $N(4h,\delta t)$ in equation (48) are chosen to be the numerical solutions of the concentrations at $x = 0$ for $t = 100$. For $h = 0.025$ and $\delta t = 0.005$ equation (48) gives 3.95, so it can be concluded

that $\alpha_1 \approx 2$. This is something you could expect, because a second order central difference formula was used. $N(h,\delta t)$, $N(h,2\delta t)$ and $N(h,4\delta t)$ in equation (49) are chosen to be the concentrations at $x = 0$ for $t = 100$. For $h = 0.01$ and $\delta t = 0.001$ equation (49) gives 4.00, so it can be concluded that $\beta_1 \approx 2$. This is also something that was to be expected based on the theory, while it was stated that the method of Modified Euler is $\mathcal{O}(\delta t^2)$

Now it should be possible to do the same thing for the separated grid integration done in section 3.4. Let $h_\chi = 0.05$, $h_r = 0.005$ and $\delta t = 1{\cdot}10^{-4}$. $N(h_\chi, h_r, \delta t)$, $N(2h_\chi, 2h_r, \delta t)$ and $N(4h_\chi, 4h_r, \delta t)$ are again the numerical solutions of the concentrations at $x = 0$ for $t = 100$. Equation (48) now results 3.87, from which can be concluded that $\alpha_1 \approx 2$. For $h_\chi = 0.1$, $h_r = 0.01$ and $\delta t = 1 \cdot 10^{-4}$ equation(49) with $N(h_\chi, h_r, \delta t)$, $N(h_\chi, h_r, 2\delta t)$ and $N(h_\chi, h_r, 4\delta t)$ results in 1.01, which means $\beta_1 \approx 0$. This last result means that the value of δt does not matter when the stability condition is satisfied. Actually this is not true, $N(h_\chi, h_r, \delta t)$, $N(h_\chi, h_r, 2\delta t)$ and $N(h_\chi, h_r, 4\delta t)$ contain a more or less constant h_χ, h_r-error that dominates the δt-error. Later in this report this constant term will be 'filtered' out.

An other way of verifying the numerical solution is by comparing it with the exact analytical solution, equation (17). In the next section this is done to find the error between them.

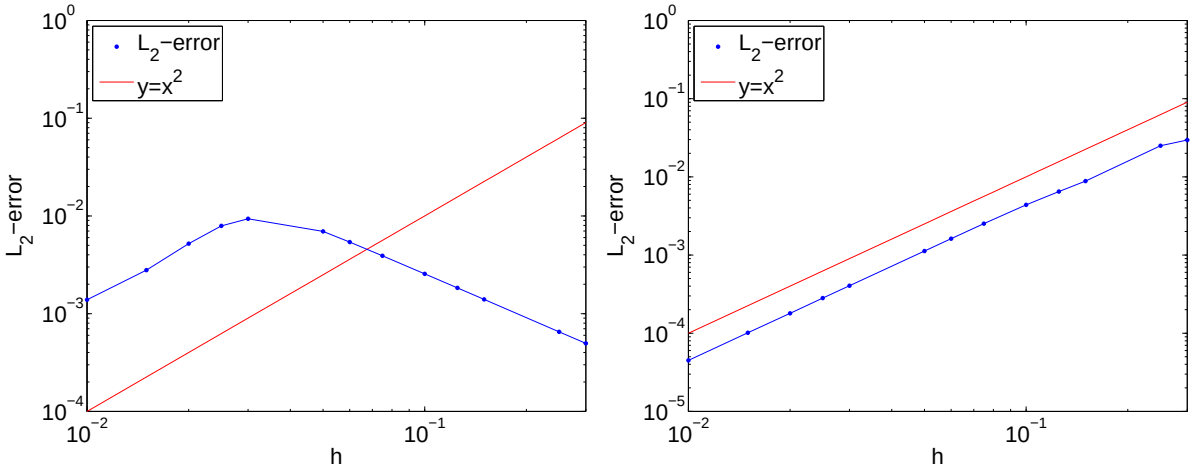
3.6. THE L_2-ERROR

With the exact solution found, a good expression for the error between the numerical solutions and the exact solution is given by the L_2-error defined by:

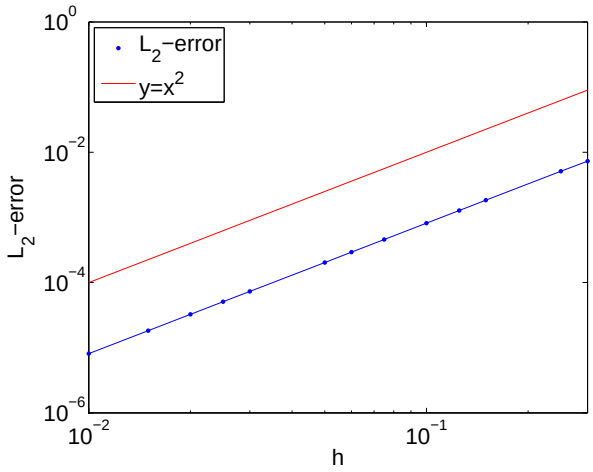
$$L_2 = \sqrt{\frac{\sum_{i=1}^{N} (c(x_i, t) - \boldsymbol{c}_i)^2}{N}} \tag{50}$$

Here $c(x_i, t)$ is the exact solution in x_i at time t and $\boldsymbol{c}_i$ is the numerical solution in x_i at time t. The L_2-error actually represents the average squared difference between the points where the numerical solution is known and the exact solution in those points.

At first the single grid numerical solution is compared with the exact solution. The L_2-errors determined in the rest of this report only depend on the points on the positive x-axis, because, like earlier stated, the problem is symmetrical. With $\delta t = 5 \cdot 10^{-5}$ and for various values of h the L_2-errors at $t = 0.01$, $t = 1$, $t = 10$ are compared, see figure 6. For t large enough, so the problem is in the asymptotic region, the L_2-error shows a quadratic, or second order, behavior; the dots lay on a straight line approximately parallel to $y = x^2$. This proves once again that Modified Euler is $\mathcal{O}(h^2)$. The strange behavior of the L_2-error at $t = 0.01$ in figure 6a can be clarified. For t small and h relatively large the effect of the discontinuity at $t = 0$ plays a role in the error behavior. The discontinuity at $t = 0$ results in large values of the derivatives for t close to zero. Small values of h are needed to be in the asymptotic region.



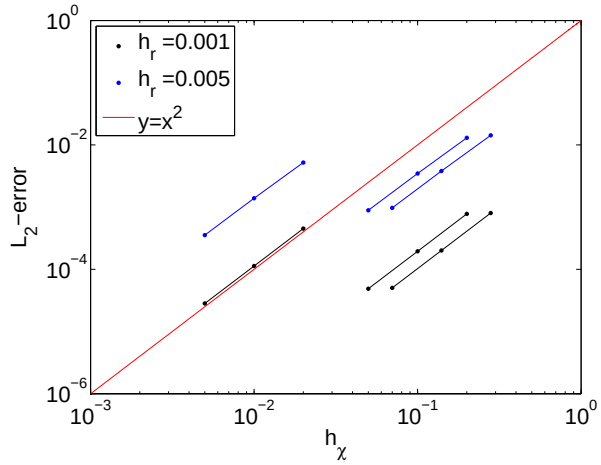
(a) $t = 0.01$ (b) $t = 1$



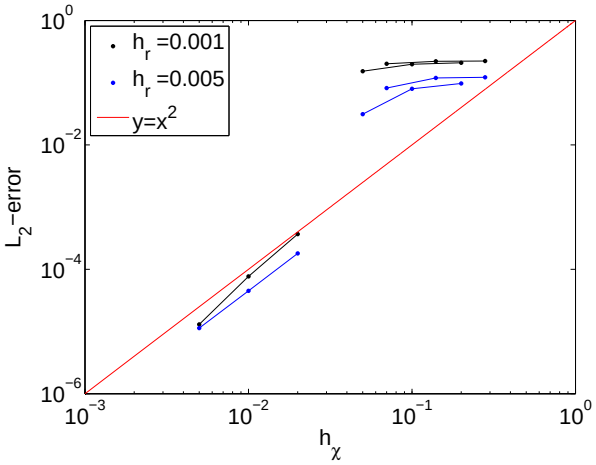
(c) $t = 10$

Figure 6: The L_2-error behavior for the single grid case at various times.

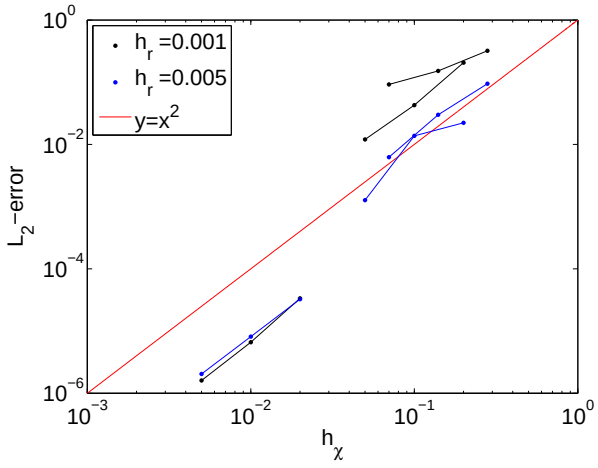
This is also done for the separated grid integration. With $\delta t = 5 \cdot 10^{-6}$ and again the t values of 0.01, 1, 10 the grid size in both grids is repeatedly doubled. This result can be seen in figure 7. The entries of the legend are every time the values of h_r for the smallest values of h_χ on every curve. So for example $(h_r, h_\chi) = (0.001, 0.005), (0.002, 0.01), (0.004, 0.02)$ are the three dots that lie nearly on the red line in figure 7a.



(a) $t = 0.01$



(b) $t = 1$



(c) $t = 10$

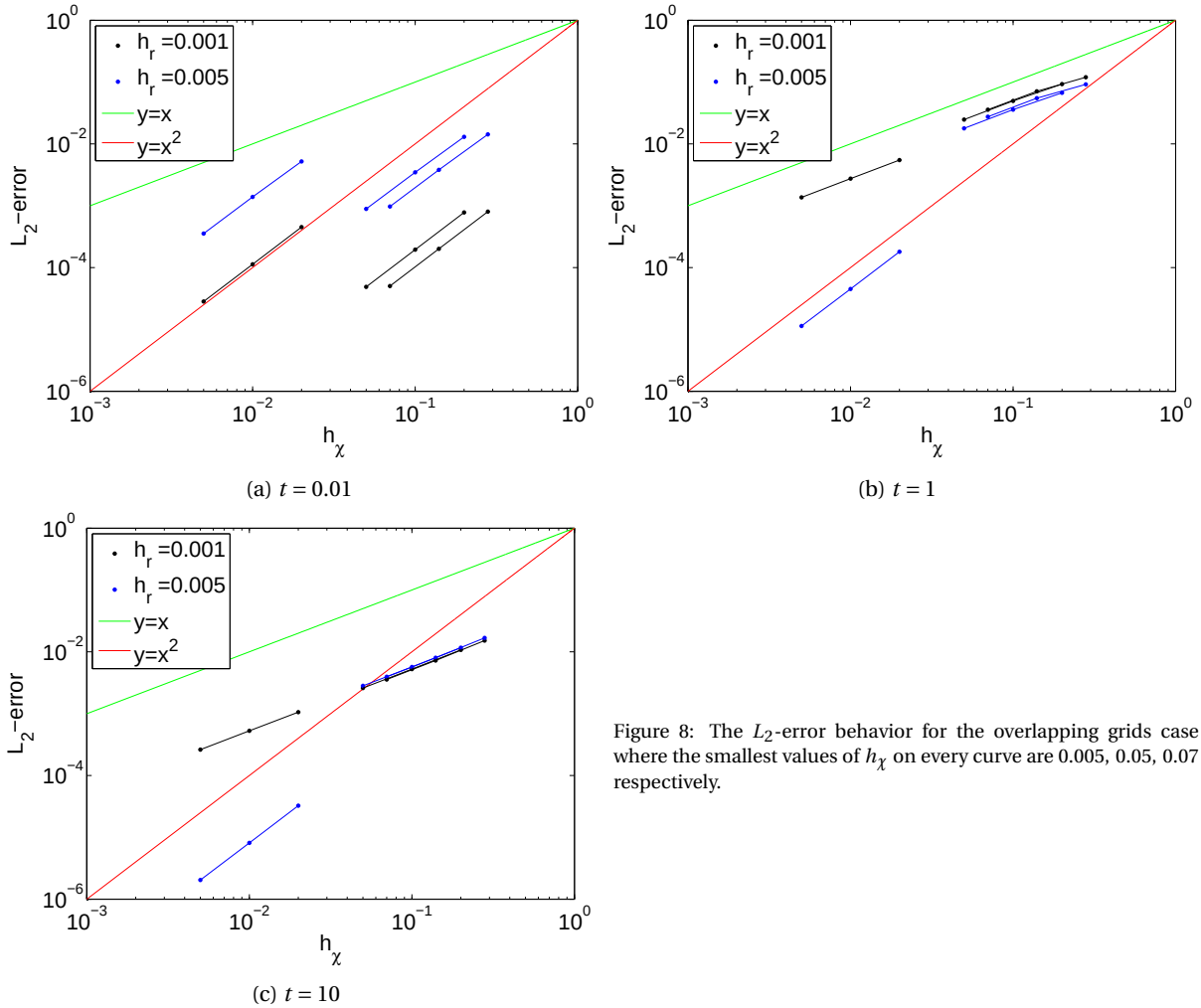
Figure 7: The L_2-error behavior for the separated grid case where the smallest values of h_χ on every curve are 0.005, 0.05, 0.07 respectively.

Figure 7a shows that for $t = 0.01$ the L_2-error behavior is still quadratic, as hardly any coupling of concentration, other than zero, between the grids took place. For t larger there is definitely something else going on. The error behavior is not quadratic anymore and in figure 7b and figure 7c the curves for $h_r = 0.001$ lay above the curves for $h_r = 0.005$. The latter is caused by the fact that for $h_r = 0.005$ and $h_\chi = 0.005$ every time the boundary points of the r-grid and the last point of the χ-grid coincide, so the extrapolation is exact.

The L_2-error behavior in Figure 7b and Figure 7c is not the behavior sought-after. Besides it is not possible to extend this method to higher dimensions, because the end point of the fine grid, r_0, will not always be a boundary point for the coarse grid. Another possibility for coupling two grids is to use overlapping grids with interpolation for the coupling instead of separated grids with extrapolation.

3.7. Overlapping grids

Where in the separated grids case the two grids where completely separated, in the overlapping grids case there is at least one point of the χ-grids that lies inside r-grid and at least one point of the r-grid that lies inside the χ-grid. For now the only thing that changes is that $\chi_e = r_0$ and instead of extrapolation, with a Lagrange polynomial to find a boundary value for the r-grid in $x = r_0 - h_r$, linear interpolation between $\chi_e - h_\chi$ and χ_e will be used to determine the concentration in $x = r_0 - h_r$. Figure 8 shows the results for the same grid sizes as in the separated grids case.



(a) $t = 0.01$

(b) $t = 1$

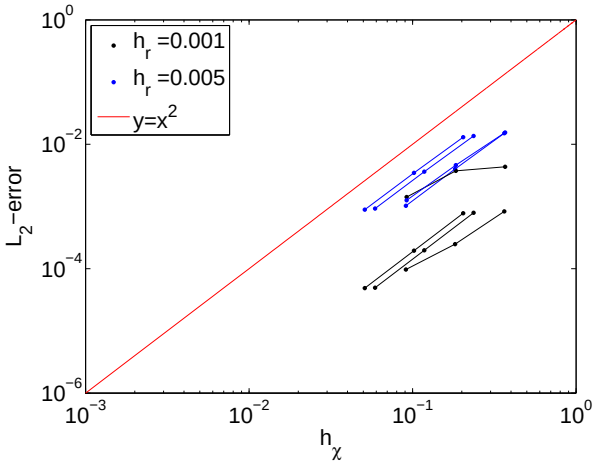
(c) $t = 10$

Figure 8: The L_2-error behavior for the overlapping grids case where the smallest values of h_χ on every curve are 0.005, 0.05, 0.07 respectively.

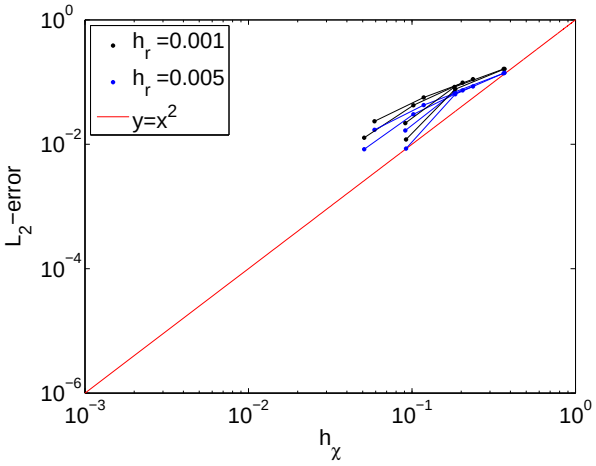
The curves in figure 8b and figure 8c are still straight lines, but the order is in most cases less than 2, which is what is sought-after. From comparison with figure 7b and figure 7c it can be concluded that in most cases the

error is less with linear interpolation than with extrapolation and the error behavior nicer with linear interpolation than with extrapolation. The only curves that do not meet this conclusion are those with $h_r = 0.001$ and $h_\chi = 0.005$ as start values, but this is of minor importance since h_χ will be relatively large in the actual problem.

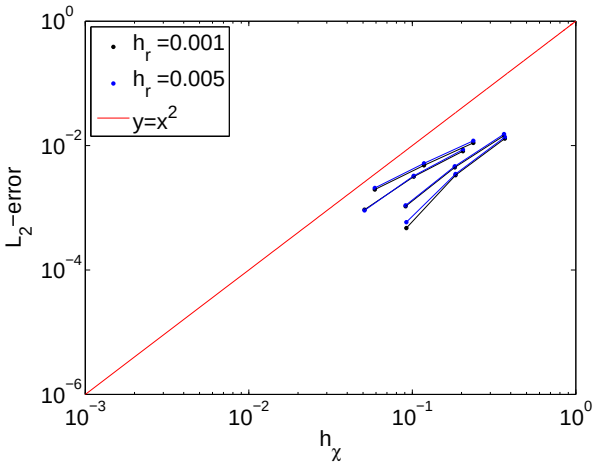
To check if this method if still usable in higher dimensions the values 0.051, 0.059, 0.091. 0.092 for h_χ were picked. These values result in linear interpolation from the r-grid to the χ-grid and linear interpolation from the coarse χ-grid to the r-grid. Where on the curves of $h_r = 0.001$ the linear interpolation from the r-grid to the χ-grid is in all cases exact, for $h_r = 0.005$ the linear interpolation is not exact anymore. Figure 9 shows the results.



(a) $t = 0.01$



(b) $t = 1$

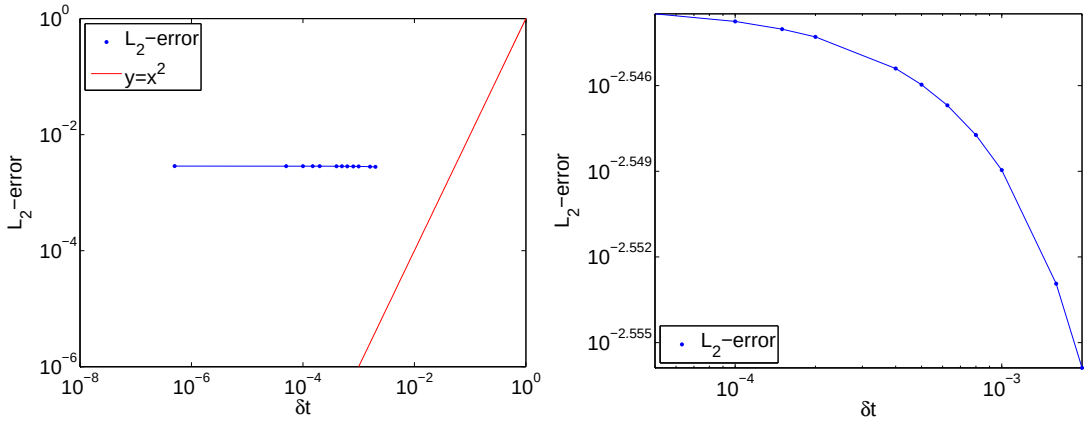


(c) $t = 10$

Figure 9: The L_2-error behavior for the overlapping grids case where the smallest values of h_χ on every curve are 0.051, 0.059, 0.091, 0.092 respectively.

Only for $t = 0.01$ and start values $h_\chi = 0.051$ and $h_\chi = 0.059$ in figure 9a there is error behavior of a certain order. All other curves are no straight lines in figure 9. Although it is hard to say something about the graphs, it can be concluded that in comparison with figure 7 the errors are small. So also in this case linear interpolation seems better than extrapolation.

For further investigation of the coupling, it might also be useful to have a look at the L_2-error relative to the time step size. From section 3.5 the order of the error for the single grid case is 2. Now for the overlapping grids case with $h_\chi = 0.07$ and $h_r = 0.01$ the L_2-error is determined for various values of δt, of course all obeying the stability conditions. For $t_e = 15$ the following result is obtained:



(a) L_2-error vs. δt

(b) Zoomed version of figure 10a

Figure 10: L_2-error as a function of δt on a double logaritmic scale;
$h_\chi = 0.07$, $h_r = 0.01$ and $t_e = 15$

In figure 10a it seems that the L_2-error is independent of δt, but a the zoomed window in figure 10b, at what seemed to be a straight line, shows that there is clearly some dependence of δt. However the striking result is that for larger values of δt the L_2-error gets smaller. This behavior is opposite the behavior sought-after. There must be a more or less constant term in the L_2-error that is dominating the δt-dependent error. This is the h_χ, h_r-dependent error that is encapsulated in every numerical solution. In the previous figures where the L_2-error depended on h, h_χ and h_r there was also a constant δt-error encapsulated in the numerical solution, but this error was that small that it was not noticeable in the figures. The opposite error behavior in figure 10b is possibly caused by the coupling. When δt is large the coupling in total takes place less often, so less times an extra error is introduced.

In order to find the error dependence of δt, the constant h_χ and h_r dependent error must be filtered out. Therefore the numerical solution is compared, by means of the L_2-error, with a numerical solution with δt very small instead of the exact solution. These numerical solutions both have the same constant h_χ, h_r-dependent error, so it drops out of the L_2-error. In figure 11 the numerical solution for various values of δt is compared with the numerical solution with $\delta t = 5 \cdot 10^{-7}$.

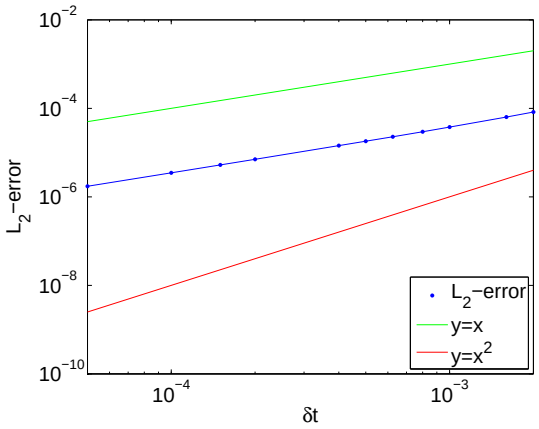
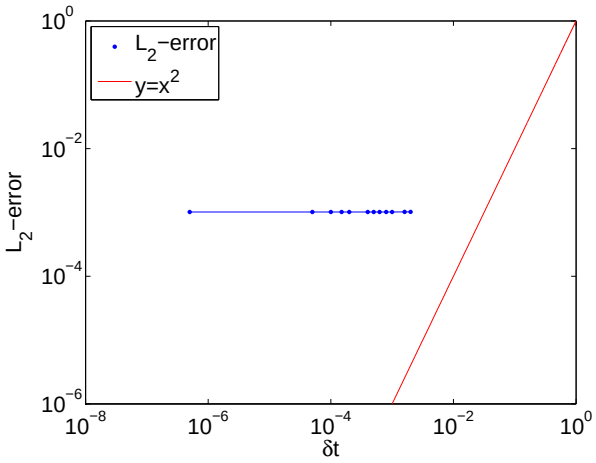
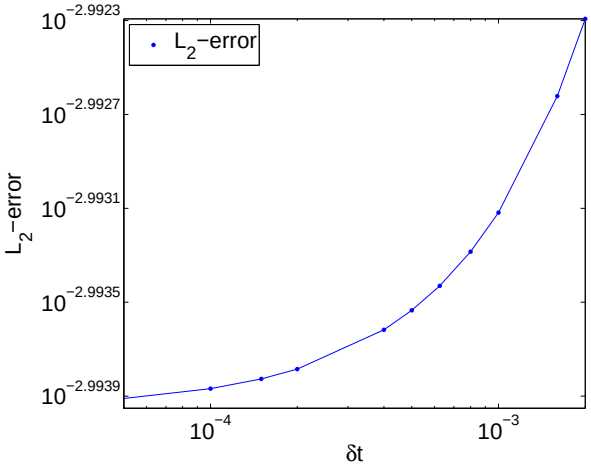


Figure 11: L_2-error, compared to a numerical solution with $\delta t = 5 \cdot 10^{-7}$,
as a function of δt on a double logaritmic scale;
$h_\chi = 0.07$, $h_r = 0.01$ and $t_e = 15$

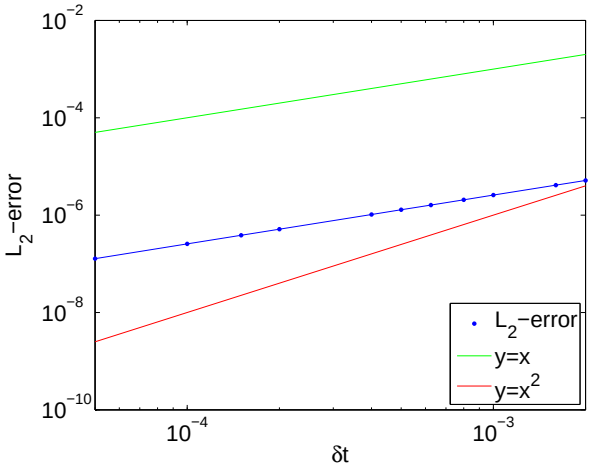
The Richardson error estimation formula gives approximately 2.02 or an order of 1 which is in accordance with figure 11. Maybe a change of location of the last point of the χ-grid in the r-grid will improve the results; so now $h_\chi = 0.08$ and $h_r = 0.01$ in which case $\chi_e = 1.44$ instead of 1.4 which was the case for $h_\chi = 0.07$.



(a)



(b)



(c)

Figure 12: $h_\chi = 0.08$, $h_r = 0.01$ and $t_e = 15$; 12a represents the L_2-error as an function of δt on a logaritmic scale; 12b is a zoomed version of 12a; 12c shows the L_2-error, compared to a numerical solution with $\delta t = 5 \cdot 10^{-7}$, as a function of δt on a double logaritmic scale.

Again there is first order error behavior, but this time the L_2-error gets smaller if δt is smaller. The Richardson error estimation formula now gives approximately 2.00 which is again 1st order. It seems that there are two cases: for some grid sizes the bigger δt, the smaller the L_2-error and for some grid sizes the smaller δt the smaller the L_2-error. The first case is probably caused by the fact that every time step the coupling introduces an error. For δt bigger there are less time steps and so less errors introduced. In the latter case the coupling is still that good that it does not dominate the normal error behavior. It could be concluded that with this coupling mechanism one order for the δt-dependent L_2-error behavior is lost, because Modified Euler normally gives second order behavior.

3.8. Direct coupling and higher order interpolation

Although the results from section 3.7 are better than those from 3.6 it should be possible to improve the coupling. At first we might look for higher order Lagrange interpolation polynomials instead of using linear interpolation for the coupling mechanism. Furthermore the coupling only takes place after a full cycle of the Modified Euler method, i.e. there is only coupling after the corrector step, equation (30). If the coupling would also take place after the predictor step, equation (29), the coupling should improve.

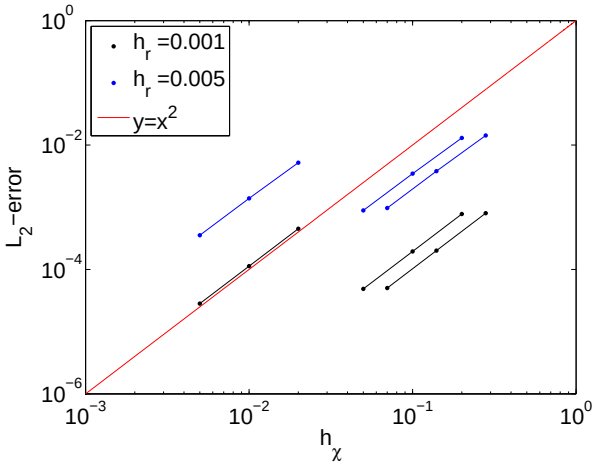
To implement the coupling after every stage (not cycle) of the Modified Euler method it would be easier to implement and quicker to compute if one matrix $\boldsymbol{A}$ would contain the entries for the r-grid, the entries for the χ-grid and the coupling mechanism between the grids. In that case there again will be one vector $\boldsymbol{c}$, instead of the vectors $\boldsymbol{c}_\chi$ and $\boldsymbol{c}_r$, that contains the concentrations of both grids. There will again also be one boundary

vector $\boldsymbol{b}$. The matrix $\boldsymbol{A}$ will have the following structure:

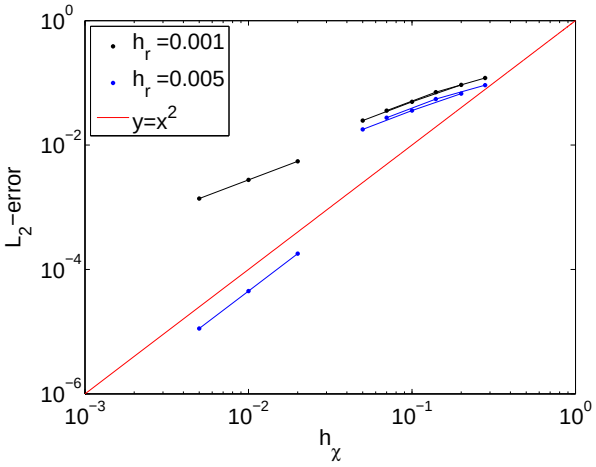
$$\boldsymbol{A} = \begin{bmatrix} & \boldsymbol{A}_\chi & & \emptyset & \\ 0 \quad \dots \quad 0 & \mathscr{L}(\chi) & \dots & \mathscr{L}(\chi) & \mathscr{L}(r) \quad \dots \quad \mathscr{L}(r) \quad 0 \quad \dots \quad 0 \\ & \emptyset & & \boldsymbol{A}_r & \end{bmatrix} \tag{51}$$

Here $\boldsymbol{A}_\chi$ and $\boldsymbol{A}_r$ are the matrices that solve the problem on the χ-grid respectively the r-grid. The entries $\mathscr{L}(r)$ are the weights of the Lagrange interpolation polynomial in the boundary point of χ-grid multiplied by $\frac{D}{h_\chi^2}$. These entries couple the concentration values of the r-grid to the boundary of the χ-grid. The entries $\mathscr{L}(\chi)$ represent the weights of the Lagrange interpolation polynomial in the boundary point of the r-grid multiplied by $\frac{D}{h_r^2}$. The total problem, including the coupling, can now be solved by just using equation (29) and (30).

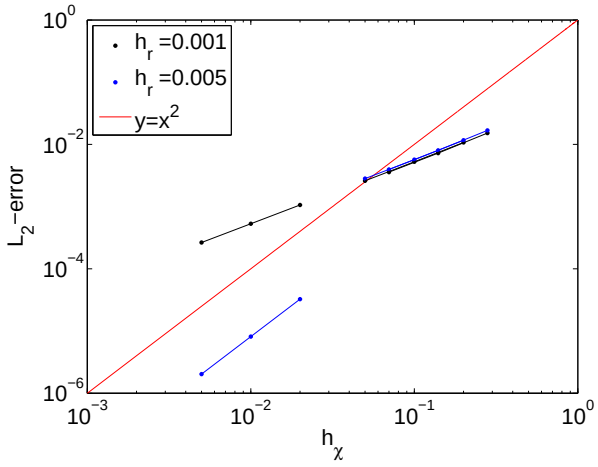
It might be possible that the new coupling method will work that good that linear interpolation, in both directions, will do the job. For that reason figures that compare to the figures in Figure 8, again with $\delta t = 5{\cdot}10^{-6}$, are generated. Note that the coupling from the r-grid to the χ-grid is exact in all case below, which means as much as that the order of the interpolation polynomial used in that direction does not effect the accuracy.



(a) $t = 0.01$



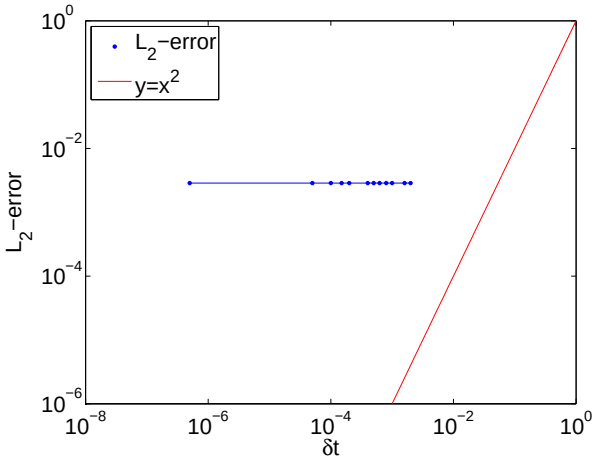
(b) $t = 1$



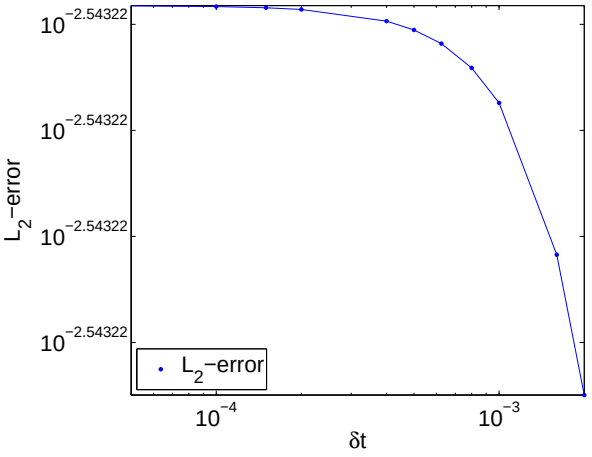
(c) $t = 10$

Figure 13: The L_2-error behavior for direct coupling with linear interpolation. The smallest values of h_χ on every curve are 0.005, 0.05, 0.07 respectively.

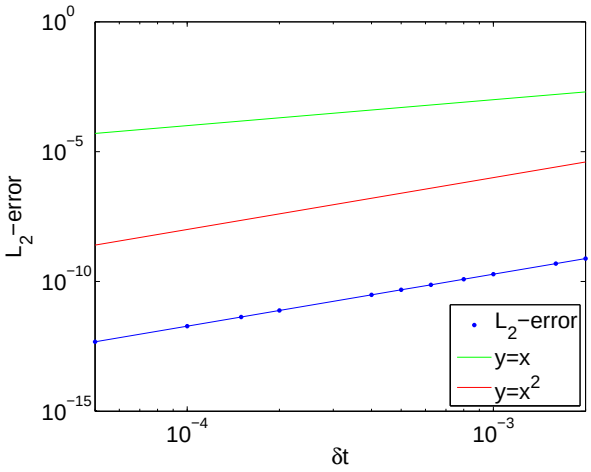
These figures resemble the figures in figure 8, so there can be concluded that the coupling after every step instead of every cycle does not make much difference for the h_χ, h_r-dependent L_2-error. A look at the δt-dependent behavior instead gives improved results:



(a)



(b)

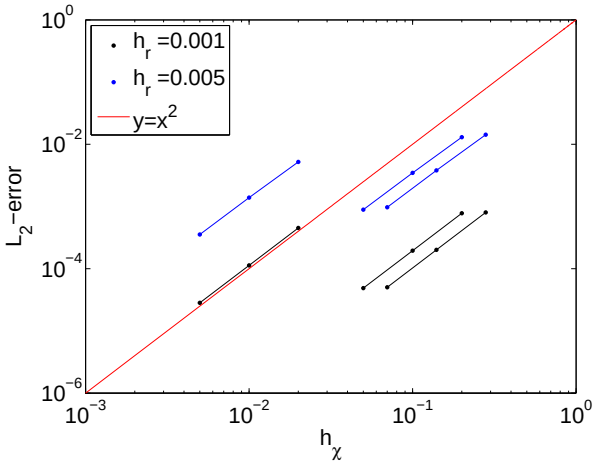


(c)

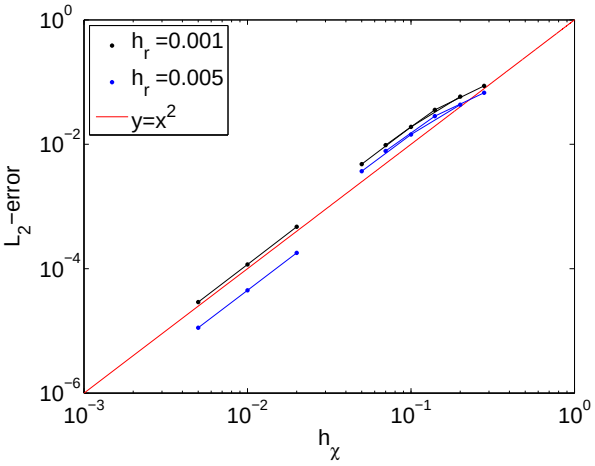
Figure 14: $h_\chi = 0.07$, $h_r = 0.01$ and $t_e = 15$; 14a represents the L_2-error as an function of δt on a logaritmic scale; 14b is a zoomed version of 14a; 14c shows the L_2-error, compared to a numerical solution with $\delta t = 5 \cdot 10^{-7}$, as a function of δt.

The Richardson extrapolation formula now gives approximately 4.00 and thus 2$^{\text{nd}}$ order behavior, which is as good as it can possibly be with the method of Modified Euler.

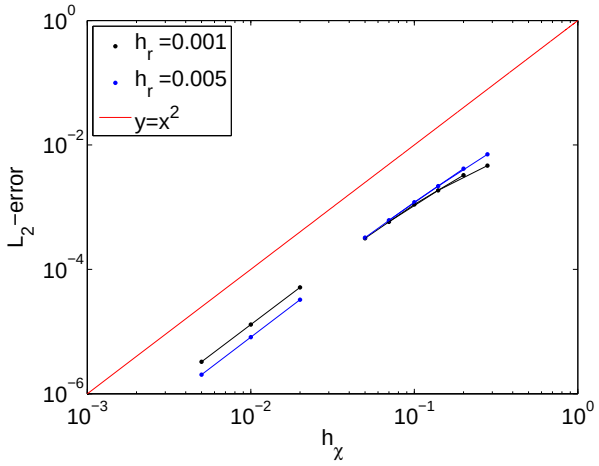
The second improvement suggested is to use higher order Lagrange interpolation polynomials. With a second order interpolation polynomial, in both directions, these are the results:



(a) $t = 0.01$



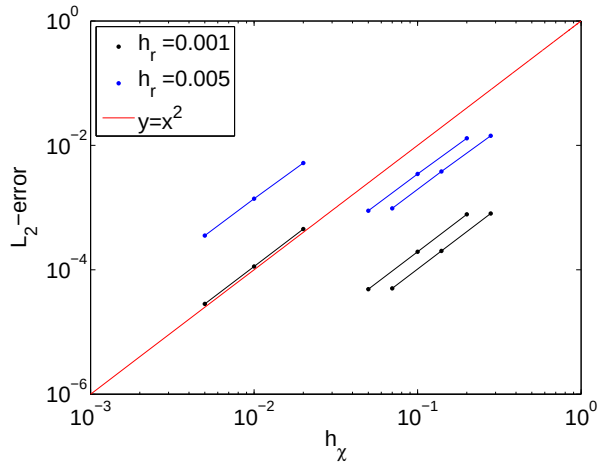
(b) $t = 1$



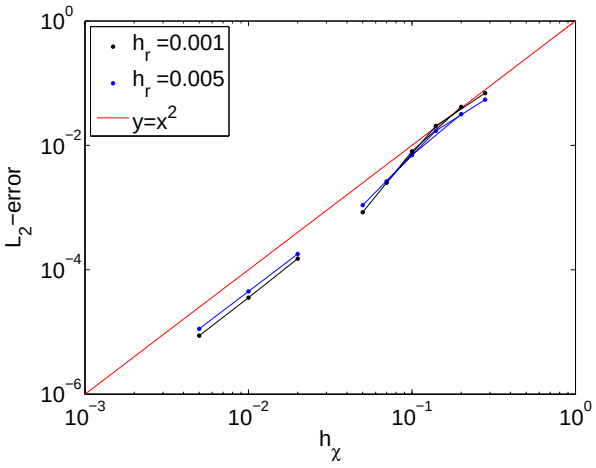
(c) $t = 10$

Figure 15: The L_2-error behavior for direct coupling with a second order Lagrange interpolation polynomial. The smallest values of h_χ on every curve are 0.005, 0.05, 0.07 respectively.

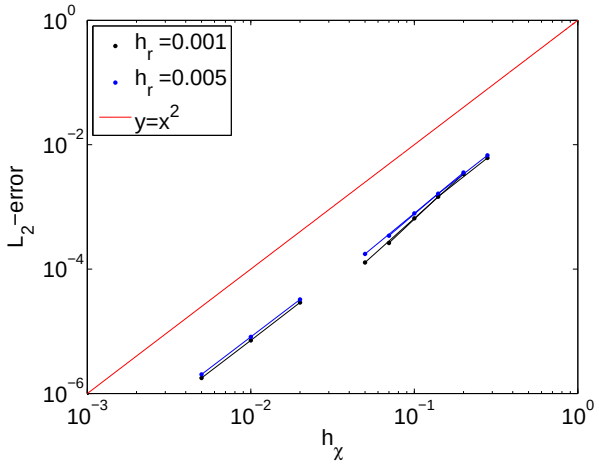
The results in figure 15 are already much better than those in figure 13 and the error behavior tends to second order. Maybe it can get even be better when using a third order Lagrange interpolation polynomial, in both directions, these are the results:



(a) $t = 0.01$



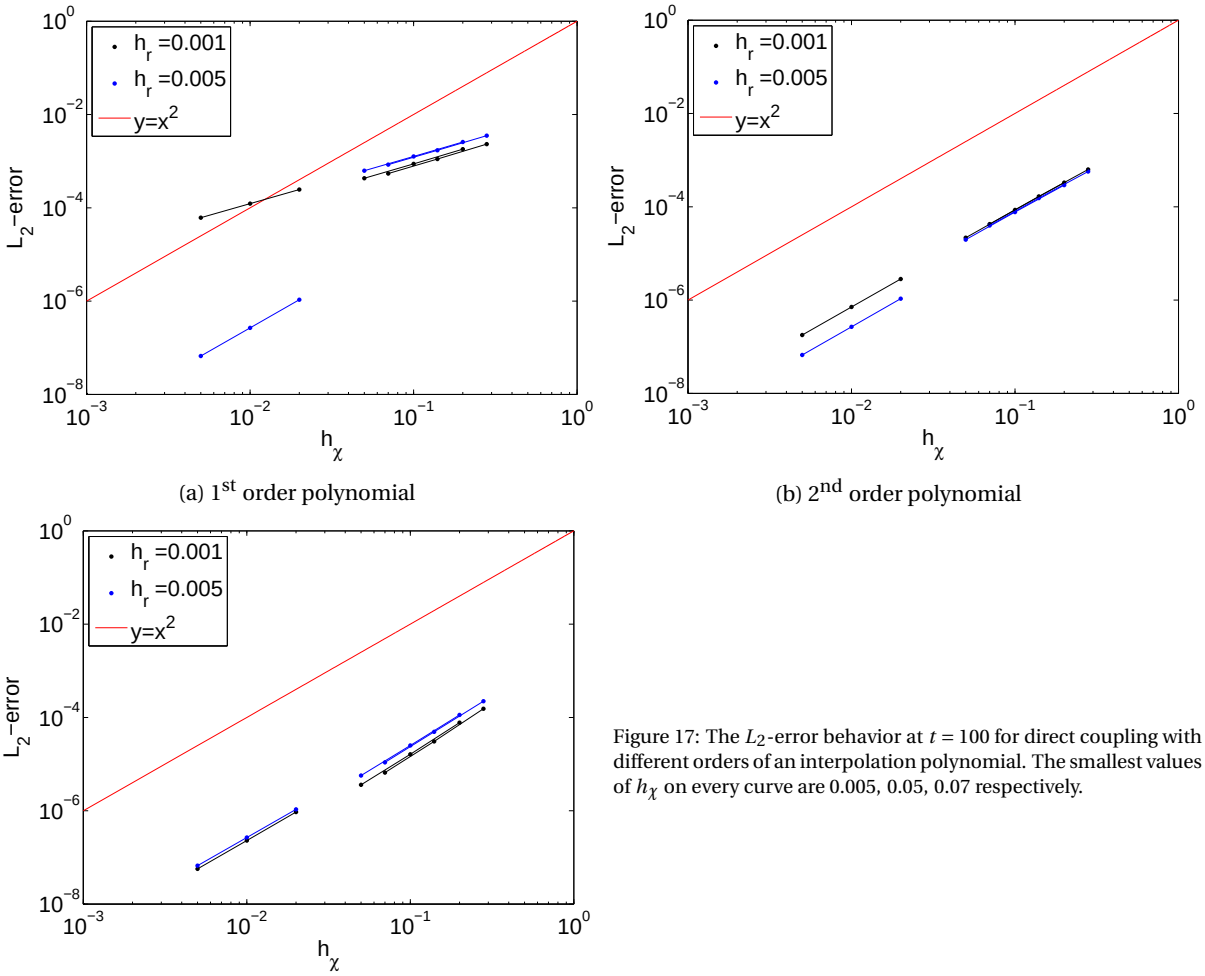
(b) $t = 1$



(c) $t = 10$

Figure 16: The L_2-error behavior for direct coupling with a third order Lagrange interpolation polynomial. The smallest values of h_χ on every curve are 0.005, 0.05, 0.07 respectively.

In figure 16b and 16c still not all the points lie on a straight lines of second order. Just like in the single grid case, figure 6, the solution is asymptotically stable. So maybe the straight lines will indeed appear for larger values of t. For $t = 100$ the cases of linear interpolation, interpolation with a second order Lagrange interpolation polynomial an interpolation with a third order Lagrange interpolation polynomial are displayed in figure 17.



(a) 1st order polynomial

(b) 2nd order polynomial

(c) 3rd order polynomial

Figure 17: The L_2-error behavior at $t = 100$ for direct coupling with different orders of an interpolation polynomial. The smallest values of h_χ on every curve are 0.005, 0.05, 0.07 respectively.

Most lines in Figure 17a still show an error behavior that is not of second order, but the lines in figure 17b and 17c both seem to be of second order. The method of Richardson error estimation could now possibly determine the true order of these lines.

It nice to see that the coupling with higher order interpolation polynomials results in a better error behavior, but what is the theory behind it. Consider the first point, the point $r = 1.4$ in the example of this report, in the r-grid. The difference formula for the second derivative is according to (21) given by:

$$\frac{\boldsymbol{c}_{i-1} - 2\boldsymbol{c}_i + \boldsymbol{c}_{i+1}}{h_r^2} \tag{52}$$

Normally the error in the difference formula relative to the continuous case would be $\mathcal{O}(h_r^2)$, but the Taylor expansion from (22) does not hold anymore for this case of coupling. Where $\boldsymbol{c}_i$ and $\boldsymbol{c}_{i+1}$ are just known concentrations in the r-grid, $\boldsymbol{c}_{i-1}$ is determined by interpolation from the χ-grid. To start with linear interpolation, we investigate the influence of the interpolation on the error. The expression for the linear interpolation polynomial is given by:

$$p(\chi_{i-1}) = \boldsymbol{c}_j + \frac{\chi_{i-1} - \chi_j}{\chi_{j-1} - \chi_j}\left(\boldsymbol{c}_{j-1} - \boldsymbol{c}_j\right) \tag{53}$$

Where $\boldsymbol{c}_j$ is the concentration in χ_j, $\boldsymbol{c}_{j-1}$ the concentration in χ_{j-1} and $p(\chi_{i-1})$ the interpolated concentration $\boldsymbol{c}_{i-1}$. A theorem on [3, p.13] states:

$$\boldsymbol{c}_{i-1} - p(\chi_{i-1}) = \frac{1}{2}(\chi_{i-1} - \chi_j)(\chi_{i-1} - \chi_{j-1})c''(\mu) \tag{54}$$

Here $\chi_{j-1} < \mu < \chi_j$, so $c(\mu)$ is the concentration somewhere between χ_{j-1} and χ_j. If $\chi_{i-1} = \chi_j$ or $\chi_{i-1} = \chi_{j-1}$ it follows that $\boldsymbol{c}_{i-1} - p(\chi_{i-1}) = 0$ and there is no additional error made in the difference formula due to the linear interpolation. If $\chi_{i-1} \neq \chi_j$ and $\chi_{i-1} \neq \chi_{j-1}$ it can be stated without the loss of generality that

$$|\mu - \chi_j| \leq \frac{|\chi_j - \chi_{j-1}|}{2} \text{ and } |(\mu - \chi_j)(\mu - \chi_{j-1})| \leq \frac{(\chi_j - \chi_{j-1})^2}{4} \tag{55}$$

And from this it can be concluded that in the worst case scenario the error is bounded by

$$|\boldsymbol{c}_{i-1} - p(\chi_{i-1})| \leq \left|\frac{1}{2}(\mu - \chi_j)(\mu - \chi_{j-1})c''(\mu)\right| \leq \left|\frac{1}{8}(\chi_j - \chi_{j-1})^2 c''(\mu)\right| = Kh_\chi^2 \tag{56}$$

Where $K = \frac{c''(\mu)}{8}$. The error introduced by linear interpolation is thus $\mathcal{O}(h_\chi^2)$. In a similar way it can be shown that the error introduced by a second order interpolation polynomial is $\mathcal{O}(h_\chi^3)$ and for a third order interpolation polynomial $\mathcal{O}(h_\chi^4)$.

From the preceding discussion the overall error in the difference formula can be derived in the case of linear interpolation as follows. It was already proved that the numerator of (52) leads to an error of $\mathcal{O}(h_r^4)$. Now the extra error of $\mathcal{O}(h_\chi^2)$ caused by $\boldsymbol{c}_{i-1}$ leads upon a total error of

$$\frac{h_r^2 \boldsymbol{c}_i'' + \mathcal{O}(h_\chi^2) + \mathcal{O}(h_r^4)}{h_r^2} = \boldsymbol{c}_i'' + \mathcal{O}\left(\frac{h_\chi^2}{h_r^2}\right) + \mathcal{O}(h_r^2) \tag{57}$$

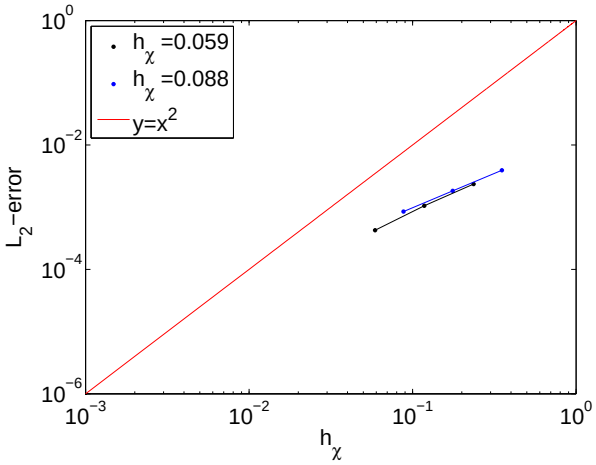
While $\mathcal{O}\left(\frac{h_\chi^2}{h_r^2}\right) \gg \mathcal{O}(h_r^2)$ the total result will be that second order error behavior will theoretically not be reached. Furthermore $\mathcal{O}\left(\frac{h_\chi^2}{h_r^2}\right) \nrightarrow 0$ as $h_\chi \to 0$ and $h_r \to 0$. There will always be a constant term in the error. So for h_r relatively large the error behavior could look like second order, but for h_r small the constant term will dominate the second order behavior. It follows that at least interpolation with $\mathcal{O}(h_\chi^3)$ is required to get $\mathcal{O}\left(\frac{h_\chi^3}{h_r^2}\right) \to 0$ as $h_\chi \to 0$ and $h_r \to 0$.

It would be nice if from the above results something could be said about the error of the total problem, the global error. The discussion above was about the local error. The discretizaton of the diffusion equation is given by:

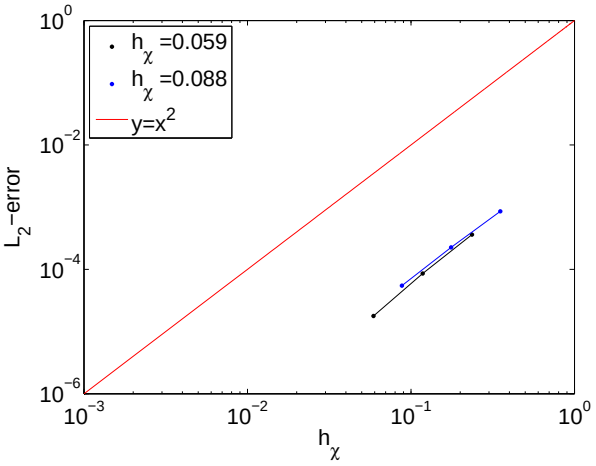
$$\frac{\boldsymbol{c}_i^{m+1} - \boldsymbol{c}_i^m}{\delta t} + \mathcal{O}(\delta t) = \frac{\boldsymbol{c}_{i-1}^m - 2\boldsymbol{c}_i^m + \boldsymbol{c}_{i+1}^m}{h^2} + \mathcal{O}(h^2) \tag{58}$$

Note that Euler Forward is used, since it is much more complicated to explain this concept for Modified Euler. The Lax equivalence theorem[10] says something about the relation between the local and the global error: If a numerical scheme is consistent and zero-stable, then the numerical solution will converge to the exact solution. Consistent means that the local error goes to zero when $h \to 0$ and $\delta t \to 0$. This is for 58 certainly the case. Zero-stable means that the if a perturbation in the starting value of ϵ causes the numerical solution over that time interval to change by no more than $K\epsilon$ for some constant $K \in \mathbb{R}$. This is even the case for linear interpolation, see (56). But for linear interpolation the local error is not consistent and thus the conditions of Lax equivalence theorem are not fulfilled. For a second order interpolation polynomial these conditions are fulfilled and the theorem says that the numerical solution will converge to the exact solution when $h \to 0$ and $\delta t \to 0$.

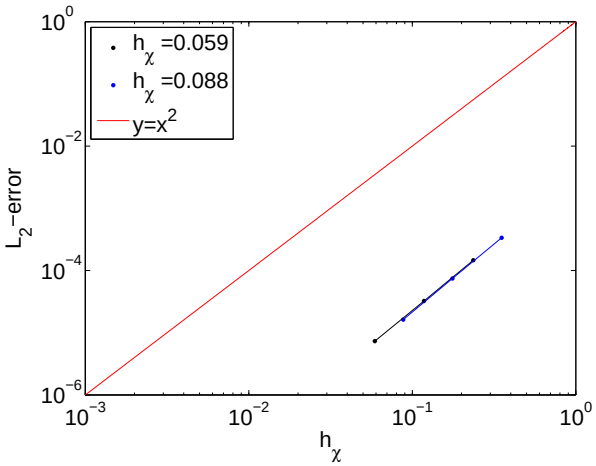
As a conclusion it might be interesting to look at some of those cases where the coupling in both directions is not exact, so in both directions there really is interpolation. For smallest values of $h_\chi = 0.059, 0.088$ and smallest value of $h_r = 0.005$ the following figures are generated:



(a) 1st order polynomial



(b) 2nd order polynomial



(c) 3rd order polynomial

Figure 18: The L_2-error behavior at $t = 100$ for direct coupling with different orders of interpolation polynomials. The smallest values of h_χ on every curve are 0.059 and 0.088, the smallest value for $h_r = 0.005$.

The conclusion is that interpolation with a second order or a third order polynomial in both directions results in a second order error behavior. This result will be used in the two-dimensional problem of the next section.

4. THE 2D PROBLEM WITH FLAT PLATES

With the knowledge that at least a second order interpolation polynomial is needed to get second order error behavior for the 1D problem from section 3, the problem is extended again to two dimensions. From the two points on the 'plates', the problem is now extended to two infinite parallel 'plates' that hold a concentration $c = c_0$, so a 2D problem is created. Between the plates the fluid will move in the direction parallel to the plates. So besides diffusive transport also convective transport takes place in this problem. As before the problem is symmetric and thus half the problem can be left out, see figure 22 (for now ignore the fine grid). The 2D problem is formally defined:

$$\begin{aligned} &\frac{\partial c(x,y,t)}{\partial t} = D\left(\frac{\partial^2 c(x,y,t)}{\partial x^2} + \frac{\partial^2 c(x,y,t)}{\partial y^2}\right) - v_y \frac{\partial c(x,y,t)}{\partial y},\ 0 < x < s,\ y_e < y < 0,\ \forall\, 0 < t \\ &c(s,y,t) = c_0,\ c(x,0,t) = 0,\ \frac{\partial c(0,y,t)}{\partial x} = 0 \text{ and } \frac{\partial c(x,y_e,t)}{\partial y} = 0,\ \forall\, 0 < t \\ &c(x,y,0) = 0,\ \forall\, x \neq s \end{aligned} \tag{59}$$

Where y_e is large enough not to cause any influence for the plates not being infinite, the boundary condition that the first derivative of the concentration at y_e is zero must hold. v_y is the fluid velocity along the y-axis. To get familiar with the concept of numerically solving a 2D problem, the convective term is ignored for now. So we are left with:

$$\frac{\partial c(x,y,t)}{\partial t} = D\left(\frac{\partial^2 c(x,y,t)}{\partial x^2} + \frac{\partial^2 c(x,y,t)}{\partial y^2}\right) \tag{60}$$

4.1. SPATIAL DISCRETISATION OF THE DIFFUSION EQUATION

Like in the 1D problem the diffusion equation in 2D, equation (60), could be rewritten when semi-discretization is used to approximate the second order derivatives. So we strive for a matrix $\boldsymbol{A}_{2D}$ and a vector $\boldsymbol{b}_{2D}$ such that equation (19), with $\boldsymbol{c}$ still a vector, holds for the 2D problem. Figure 19 shows a 2D grid by which the principle of the $2D$ numerical scheme will be explained; it is not the actual problem. The black dots are points that are included in the problem and the blue dots are boundary points. Assume the blue dots hold a concentration c_0. For a typical black point (i, j) the 2D version of equation (21) looks like:

$$\begin{aligned} \left.\frac{\partial^2 c}{\partial x^2}\right|_{x=ih_x} + \left.\frac{\partial^2 c}{\partial y^2}\right|_{y=jh_y} &= \frac{\boldsymbol{c}_{i-1,j} - 2\boldsymbol{c}_{i,j} + \boldsymbol{c}_{i+1,j}}{h_x^2} + \mathcal{O}(h_x^2) \\ &+ \frac{\boldsymbol{c}_{i,j-1} - 2\boldsymbol{c}_{i,j} + \boldsymbol{c}_{i,j+1}}{h_y^2} + \mathcal{O}(h_y^2) \end{aligned} \tag{61}$$

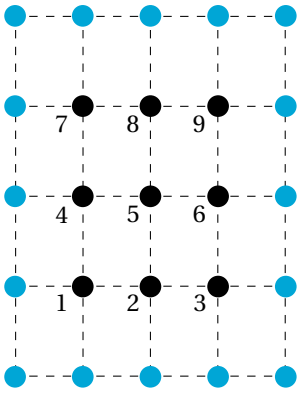


Figure 19: Simple 2D grid

Where h_x is the grid size in the horizontal direction and h_y the grid size in the vertical direction. A closer look at one specific point, point 5 for example, will make things more clear:

$$\left.\frac{\partial^2 c}{\partial x^2}\right|_{x=2h_x} + \left.\frac{\partial^2 c}{\partial y^2}\right|_{x=2h_y} \approx \frac{\boldsymbol{c}_4 - 2\boldsymbol{c}_5 + \boldsymbol{c}_6}{h_x^2} + \frac{\boldsymbol{c}_2 - 2\boldsymbol{c}_5 + \boldsymbol{c}_8}{h_y^2} \tag{62}$$

The fifth row, the row that belongs to point 5, of matrix $\boldsymbol{A}_{2D}$ will look like:

$$D\begin{bmatrix} 0 & \frac{1}{h_y^2} & 0 & \frac{1}{h_x^2} & \frac{-2}{h_x^2} + \frac{-2}{h_y^2} & \frac{1}{h_y^2} & 0 & \frac{1}{h_y^2} & 0 \end{bmatrix} \tag{63}$$

This can be done for every black dot in Figure 19 to get the matrix $\boldsymbol{A}_{2D}$:

$$\boldsymbol{A}_{2D} = D\begin{bmatrix} \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & \frac{1}{h_x^2} & 0 & \frac{1}{h_y^2} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{h_x^2} & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & \frac{1}{h_x^2} & 0 & \frac{1}{h_y^2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{h_x^2} & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & 0 & 0 & \frac{1}{h_y^2} & 0 & 0 & 0 \\ \frac{1}{h_y^2} & 0 & 0 & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & \frac{1}{h_x^2} & 0 & \frac{1}{h_y^2} & 0 & 0 \\ 0 & \frac{1}{h_y^2} & 0 & \frac{1}{h_x^2} & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & \frac{1}{h_x^2} & 0 & \frac{1}{h_y^2} & 0 \\ 0 & 0 & \frac{1}{h_y^2} & 0 & \frac{1}{h_x^2} & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & 0 & 0 & \frac{1}{h_y^2} \\ 0 & 0 & 0 & \frac{1}{h_y^2} & 0 & 0 & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & \frac{1}{h_x^2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{h_y^2} & 0 & \frac{1}{h_x^2} & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} & \frac{1}{h_x^2} \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{h_y^2} & 0 & \frac{1}{h_x^2} & \frac{-2}{h_x^2}+\frac{-2}{h_y^2} \end{bmatrix} \tag{64}$$

The points in which the discretization of the second derivative needs one or two boundary values, in fact all points except point 5, are not completely dealt with in $\boldsymbol{A}_{2D}$. These boundary values are entries of the boundary vector $\boldsymbol{b}_{2D}$, witch will be discussed later.

Taking a closer look at the representation of $\boldsymbol{A}_{2D}$ will show that $\boldsymbol{A}_{2D}$ can also be written as:

$$\boldsymbol{A}_{2D} = \begin{bmatrix} \boldsymbol{A}_{1D_x} & 0 & 0 \\ 0 & \boldsymbol{A}_{1D_x} & 0 \\ 0 & 0 & \boldsymbol{A}_{1D_x} \end{bmatrix} + \frac{D}{h_y^2}\begin{bmatrix} -2\boldsymbol{I}_x & \boldsymbol{I}_x & 0 \\ \boldsymbol{I}_x & -2\boldsymbol{I}_x & \boldsymbol{I}_x \\ 0 & \boldsymbol{I}_x & -2\boldsymbol{I}_x \end{bmatrix} \tag{65}$$

With

$$\boldsymbol{A}_{1D_x} = \frac{D}{h_x^2}\begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{bmatrix} \text{ and implicit } \boldsymbol{A}_{1D_y} = \frac{D}{h_y^2}\begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{bmatrix} \tag{66}$$

and $\boldsymbol{I}_x$ the identity matrix of size $N_x \times N_x$ where N_x the number of points along the x-direction, 3 in this case. The notation of $\boldsymbol{A}_{2D}$ as in (65) is particularly useful for MATLAB®. MATLAB® has a build-in function called 'kron$(\boldsymbol{X},\boldsymbol{Y})$', which calculates the Kronecker product of $\boldsymbol{X}$ and $\boldsymbol{Y}$. This means as much as that $\boldsymbol{A}_{2D}$ can be written as 'kron$\left(\boldsymbol{I}_y,\boldsymbol{A}_{1D_x}\right)$ + kron$\left(\boldsymbol{A}_{1D_y},\boldsymbol{I}_x\right)$', which makes it particularly easy to construct $\boldsymbol{A}_{2D}$. In general the matrix $\boldsymbol{A}_{2D}$ with size $N_xN_y \times N_xN_y$, where N_y is the number of points along the y-direction, will look like:

$$\boldsymbol{A}_{2D} = \begin{bmatrix} \boldsymbol{A}_{1D_x} & & & \\ & \boldsymbol{A}_{1D_x} & \varnothing & \\ & \ddots & & \\ & \varnothing & \boldsymbol{A}_{1D_x} & \\ & & & \boldsymbol{A}_{1D_x} \end{bmatrix} + \frac{D}{h_y^2}\begin{bmatrix} -2\boldsymbol{I}_x & \boldsymbol{I}_x & & & \varnothing \\ \boldsymbol{I}_x & -2\boldsymbol{I}_x & \boldsymbol{I}_x & & \\ & \ddots & \ddots & \ddots & \\ & & \boldsymbol{I}_x & -2\boldsymbol{I}_x & \boldsymbol{I}_x \\ \varnothing & & & \boldsymbol{I}_x & -2\boldsymbol{I}_x \end{bmatrix} \tag{67}$$

At this point we take a closer look at the boundary vector $\boldsymbol{b}_{2D}$. Point 1 in Figure 19 has one boundary point in the x-direction, which results in $\frac{Dc_0}{h_x^2}$, and one boundary point in the y-direction, which results in $\frac{Dc_0}{h_y^2}$. So the first entry of $\boldsymbol{b}_{2D}$ will be $Dc_0\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)$. The whole vector will, by the same reasoning, look like:

$$\boldsymbol{b}_{2D} = Dc_0\begin{pmatrix} \frac{1}{h_x^2}+\frac{1}{h_y^2} \\ \frac{1}{h_y^2} \\ \frac{1}{h_x^2}+\frac{1}{h_y^2} \\ \frac{1}{h_x^2} \\ 0 \\ \frac{1}{h_x^2} \\ \frac{1}{h_x^2}+\frac{1}{h_y^2} \\ \frac{1}{h_y^2} \\ \frac{1}{h_x^2}+\frac{1}{h_y^2} \end{pmatrix} \tag{68}$$

The length of $\boldsymbol{b}_{2D}$ is $N_x N_y$. Now define $\boldsymbol{b}_x$ as the vector of length N_x that holds the boundary values in the x-direction and define $\boldsymbol{b}_y$ of length N_y that holds the boundary values in the y-direction. In general the vector $\boldsymbol{b}_{2D}$ can be written as:

$$\boldsymbol{b}_{2D} = \begin{pmatrix} \boldsymbol{b}_x \\ \boldsymbol{b}_x \\ \vdots \\ \boldsymbol{b}_x \\ \boldsymbol{b}_x \end{pmatrix} + \begin{pmatrix} \boldsymbol{b}_y \\ 0 \\ \vdots \\ 0 \\ \boldsymbol{b}_y \end{pmatrix} \tag{69}$$

4.2. Discretisation of the convection operator

From now on the convective term in (59) will also be considered. This means v_y needs to be known and therefore the system needs to be specified further. The 1D problem is extended just a little bit: a downward flow, due to gravity, with initial condition $c = 0$ will flow between the two infinite plates that will be on a concentration c_0. So for this setup the flow profile has to be calculated. Consider a small control volume like in figure 20a. At x y-impulse, p_y, is moving into the control volume at a rate $\phi_{p_{yx}}|_{x=x}$ and at $x+\delta x$ p_y is moving out at a rate $\phi_{p_{yx}}|_{x=x+\delta x}$. In the stationary state the y-impulse balance equation looks like:

$$0 = \phi_{p_{yx}}|_{x=x} - \phi_{p_{yx}}|_{x=x+\delta x} + \sum F_y \tag{70}$$

These $\phi_{p_{yx}}$ values can be thought of as a force acting on the control volume, namely the shear stress, τ_{xy}, multiplied by the surface area τ_{xy} is acting on. Using this concept and the fact $\sum F_y = -\rho g \delta x \delta y$, equation (70) can be rewritten as:

$$0 = \tau_{xy}|_{x=x}\delta y - \tau_{xy}|_{x=x+\delta x}\delta y - \rho g \delta x \delta y \tag{71}$$

Equation (71) leads to the following differential equation:

$$\frac{d\tau_{xy}}{dx} = -\rho g \tag{72}$$

Assumed dealing with an Newtonian fluid, the law of Newton[4, p.72] is used:

$$\left.\begin{array}{l} \tau_{xy} = -\mu \frac{dv_y}{dx} \\ \frac{d\tau_{xy}}{dx} = -\rho g \end{array}\right\} \mu \frac{d^2 v_y}{dx^2} = \rho g \tag{73}$$

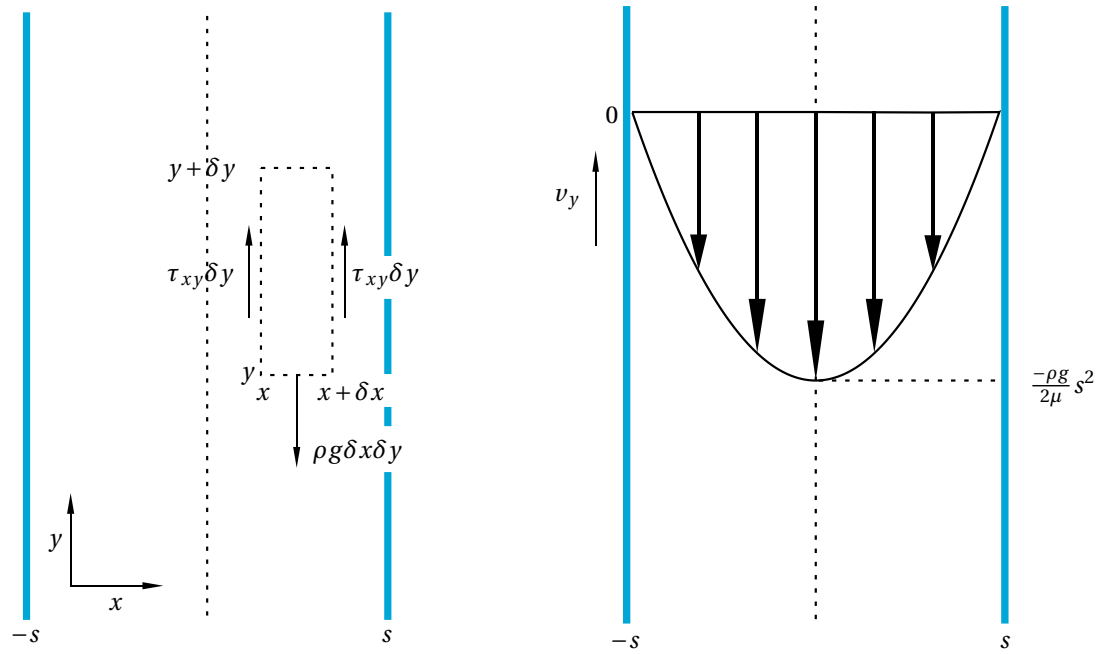
$$\Rightarrow \frac{dv_y}{dx} = \frac{\rho g}{\mu} x + C_1 \tag{74}$$

$$\Rightarrow v_y = \frac{\rho g}{2\mu} x^2 + C_1 x + C_2 \tag{75}$$

C_1 and C_2 can be determined with the aid of the boundary conditions that the flow speed at the plates is zero: $x = \pm s \rightarrow v_y = 0$. That gives $C_1 = 0$ and $C_2 = -\rho g s^2/2\mu$, which results in:

$$v_y = -\frac{\rho g}{2\mu}(s^2 - x^2) \tag{76}$$

This flow profile can be seen in Figure 20b. The whole derivation above is also mentioned on[4, p.241-242].



(a) Control volume

(b) Flow profile

Figure 20: At $y = 0$ a fluid at concentration $c = 0$ enters the system.

The numerical scheme from section 4.1 could now be adjusted for the convective term of equation (59). A discretization of this first order derivative with $\mathcal{O}(h_y^2)$ is given by:

$$\left.\frac{\partial c}{\partial y}\right|_{y=jh_y} \approx \frac{\boldsymbol{c}_{j+1} - \boldsymbol{c}_{j-1}}{2h_y} \tag{77}$$

For the case of point 5 in Figure 19 this gives:

$$\left.\frac{\partial c}{\partial y}\right|_{y=2h_y} \approx \frac{\boldsymbol{c}_8 - \boldsymbol{c}_2}{2h_y} \tag{78}$$

These differences equations multiplied by $-v_y$ can now be added to $\boldsymbol{A}_{2D}$ in order to complete the matrix with the convective entries:

$$\boldsymbol{A}_{2D} = \begin{bmatrix} \boldsymbol{A}_{1D_x} & 0 & 0 \\ 0 & \boldsymbol{A}_{1D_x} & 0 \\ 0 & 0 & \boldsymbol{A}_{1D_x} \end{bmatrix} + \frac{D}{h_y^2}\begin{bmatrix} -2\boldsymbol{I}_x & \boldsymbol{I}_x & 0 \\ \boldsymbol{I}_x & -2\boldsymbol{I}_x & \boldsymbol{I}_x \\ 0 & \boldsymbol{I}_x & -2\boldsymbol{I}_x \end{bmatrix} + \frac{1}{2h_y}\begin{bmatrix} 0 & -\boldsymbol{V}_y & 0 \\ \boldsymbol{V}_y & 0 & -\boldsymbol{V}_y \\ 0 & \boldsymbol{V}_y & 0 \end{bmatrix} \tag{79}$$

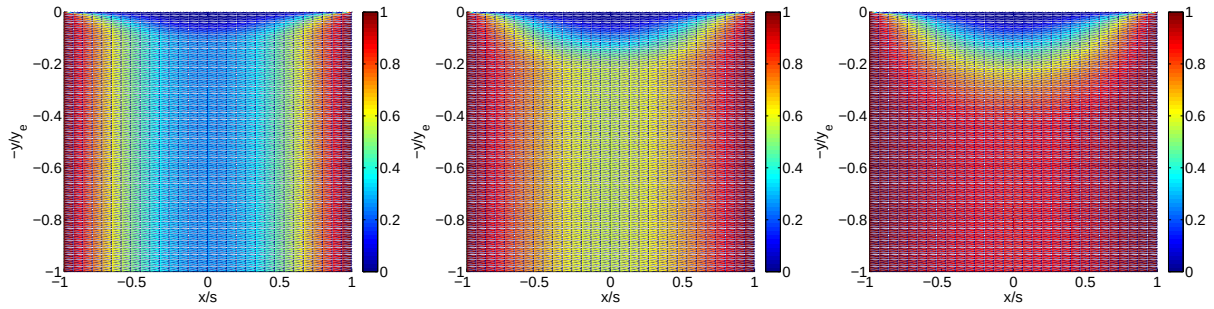
$\boldsymbol{V}_y$ is a $N_x \times N_x$ matrix that contains the discretized values of parabolic flow profile, as calculated in section 4.2, on its diagonal. According to the boundary conditions of the problem the matrices $\boldsymbol{A}_{1D_x}$ and $\boldsymbol{A}_{1D_x}$ can be adjusted:

$$\boldsymbol{A}_{1D_x} = \frac{D}{h_x^2}\begin{bmatrix} -2 & 2 & & & \emptyset \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ \emptyset & & & 1 & -2 \end{bmatrix}, \quad \boldsymbol{A}_{1D_y} = \frac{D}{h_y^2}\begin{bmatrix} -2 & 1 & & & \emptyset \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ \emptyset & & & 2 & -2 \end{bmatrix} \tag{80}$$

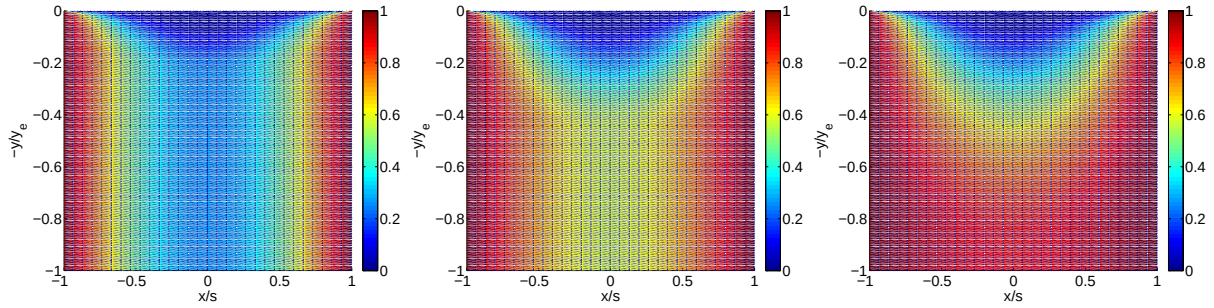
Resulting:

$$\boldsymbol{A}_{2D} = \begin{bmatrix} \boldsymbol{A}_{1D_x} & & & & \\ & \boldsymbol{A}_{1D_x} & & \emptyset & \\ & & \ddots & & \\ & \emptyset & & \boldsymbol{A}_{1D_x} & \\ & & & & \boldsymbol{A}_{1D_x} \end{bmatrix} + \frac{D}{h_y^2} \begin{bmatrix} -2\boldsymbol{I}_x & \boldsymbol{I}_x & & & \emptyset \\ \boldsymbol{I}_x & -2\boldsymbol{I}_x & \boldsymbol{I}_x & & \\ & \ddots & \ddots & \ddots & \\ & & \boldsymbol{I}_x & -2\boldsymbol{I}_x & \boldsymbol{I}_x \\ \emptyset & & & 2\boldsymbol{I}_x & -2\boldsymbol{I}_x \end{bmatrix} + \frac{1}{2h_y} \begin{bmatrix} 0 & -\boldsymbol{V}_y & & & \emptyset \\ \boldsymbol{V}_y & 0 & -\boldsymbol{V}_y & & \\ & \ddots & \ddots & \ddots & \\ & & \boldsymbol{V}_y & 0 & -\boldsymbol{V}_y \\ \emptyset & & & 0 & 0 \end{bmatrix} \tag{81}$$

In the matrix with the convective terms, the 0 entry in the last row and the penultimate column is due to the Neumann boundary condition at y_e. The representation of the boundary vector $\boldsymbol{b}_{2D}$ is like equation (69). By making matrix $\boldsymbol{A}_{2D}$ sparse, see appendix, the problem can quickly be solved. Problem (59) is solved with $\delta t = 0.01,\ t_e = 200,\ s = 1.5,\ h_x = 0.1,\ h_y = -2,\ y_e = -300s$. The results of the simulations are visible in figure 21 for various times and two different flow profiles.



(a) $\frac{-\rho g}{2\mu} = -0.5$ and $t = 50$ (b) $\frac{-\rho g}{2\mu} = -0.5$ and $t = 100$ (c) $\frac{-\rho g}{2\mu} = -0.5$ and $t = 200$



(d) $\frac{-\rho g}{2\mu} = -1$ and $t = 50$ (e) $\frac{-\rho g}{2\mu} = -1$ and $t = 100$ (f) $\frac{-\rho g}{2\mu} = -1$ and $t = 200$

Figure 21: Concentration plots with $\delta t = 0.01,\ s = 1.5,\ h_x = 0.1,\ h_y = -3,\ y_e = -300s$

These results seem quite probable. Something that immediately strikes is the sharp concentration gradient near the plates at $t = 50$. This asks for a grid refinement near the plates. In figure 21d this gradient is even a bit larger than in figure 21a, concluding that higher velocity results in larger gradients as was theoretically predicted.

4.3. Richardson error estimation

As in the one-dimensional case Richardson error estimation could be used to test the accuracy of the integration process. Equation (42) from section 3.5 can be rewritten for the two-dimensional case into:

$$M - N(h_x, h_y, \delta t) \approx K_1 h_x^{\alpha_1} + K_2 h_y^{\beta_1} + K_3 \delta t^{\gamma_1} \tag{82}$$

And analog to equations (48) and (49):

$$\frac{N(2h_x,h_y,\delta t)-N(4h_x,h_y,\delta t)}{N(h_x,h_y,\delta t)-N(2h_x,h_y,\delta t)} \approx 2^{\alpha_1} \tag{83}$$

$$\frac{N(h_x,2h_y,\delta t)-N(h_x,4h_y,\delta t)}{N(h_x,h_y,\delta t)-N(h_x,2h_y,\delta t)} \approx 2^{\beta_1} \tag{84}$$

$$\frac{N(h_x,h_y,2\delta t)-N(h_x,h_y,4\delta t)}{N(h_x,h_y,\delta t)-N(h_x,h_y,2\delta t)} \approx 2^{\gamma_1} \tag{85}$$

Remember that for equation (82) to hold the assumption that h_x, h_y and δt are small has been made. For that reason $s = 1.5$, $y_e = -4$, $h_x = 0.0125$, $h_y = -0.05$ and $\delta t = 0.001$ are chosen. Further $\frac{-\rho g}{2\mu} = -0.01$. At $t = 10$ the values of N are determined at a random, but every time the same, point in the grid. With those values equation (83) gives approximately 4.01 or $\alpha_1 \approx 2$, equation (84) gives approximately 4.03 or $\beta_1 \approx 2$ and equation (85) gives approximately 4.00 or $\gamma_1 \approx 2$. From these values it can be concluded that the integration process works correctly.

4.4. Overlapping grids

Now the correct way of handling the two-dimensional problem for a single grid has been found, we can move on to the case of overlapping grids. The formal definition of the problemis given by:

$$\begin{aligned}
&\frac{\partial c_\chi(\chi,y,t)}{\partial t} = D\left(\frac{\partial^2 c_\chi(\chi,y,t)}{\partial \chi^2}+\frac{\partial^2 c_\chi(\chi,y,t)}{\partial y^2}\right) - v_y\frac{\partial c_\chi(\chi,y,t)}{\partial y},\ 0<\chi<\chi_e,\ y_e<y<0,\ \forall\, 0<t\\
&c_\chi(\chi,y,t) = f(r),\ c_\chi(\chi,0,t)=0,\ \frac{\partial c_\chi(0,y,t)}{\partial \chi}=0 \text{ and } \frac{\partial c_\chi(\chi,y_e,t)}{\partial y}=0,\ \forall\, 0<t\\
&c_\chi(\chi,y,0)=0,\ \forall\, x
\end{aligned} \tag{86}$$

$$\begin{aligned}
&\frac{\partial c_r(r,y,t)}{\partial t} = D\left(\frac{\partial^2 c_r(r,y,t)}{\partial r^2}+\frac{\partial^2 c_r(r,y,t)}{\partial y^2}\right) - v_y\frac{\partial c_r(r,y,t)}{\partial y},\ r_0<r<s,\ y_e<y<0,\ \forall\, 0<t\\
&c_r(r,y,t) = g(\chi),\ c_r(r,0,t)=0,\ \frac{\partial c_r(0,y,t)}{\partial r}=0 \text{ and } \frac{\partial c_r(r,y_e,t)}{\partial y}=0,\ \forall\, 0<t\\
&c_r(r,y,0)=0,\ \forall\, r\neq s
\end{aligned} \tag{87}$$

Where $r_0 \leq \chi_e$. Next to the plates the fine r-grid is laid. This grid will only be fine in the x-direction for the reason that the diffusive process will especially take place in that direction and the convective process dominates the diffusive process in the y-direction. Figure 22 shows the grid for the overlapping grid case. Remember the symmetry of the problem in $x = 0$, the red line. Every intersection of horizontal and vertical lines is a grid point. Define h_χ and h_r as the distance between two grid points in the x-direction of the χ-grid respectively the r-grid and h_y the distance between two grid points in the y-direction. Just as in the one-dimensional case for the overlapping grids there is one vector $\boldsymbol{c}$ which can be integrated with the matrix $\boldsymbol{A}_{2D}$ from (81).

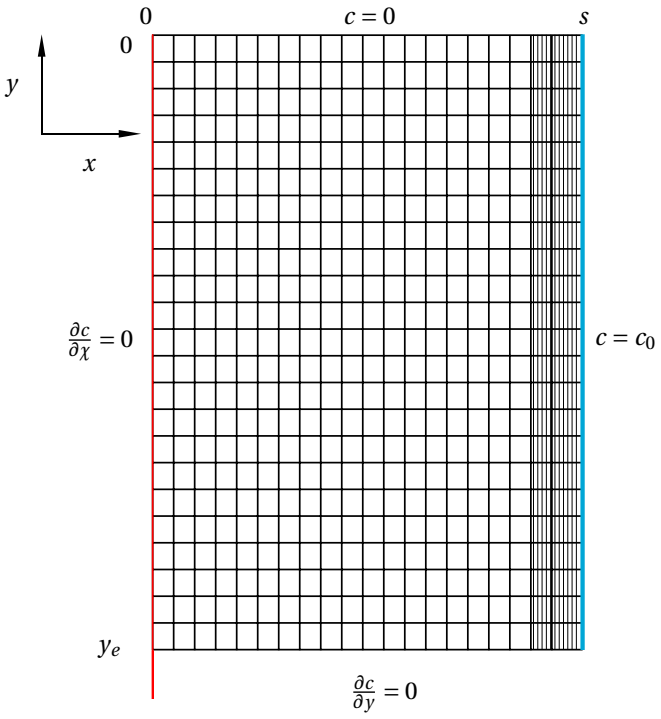


Figure 22: The overlapping grids in 2D

4.5. The L_2-error

The problem could now be solved, but that would be useless, because the problem is not analytically solvable and thus there would not be an exact solution to compare the numerical solution with. Although it is not possible to test the accuracy of the

numerical solution of 86, there actually is a technique to test the accuracy of the code according to [6, p.41-58]. This technique is based on a verified guess of a solution of the convection-diffusion equation with a source term, $Q(x,y,t)$:

$$\frac{\partial c(x,y,t)}{\partial t} = D\left(\frac{\partial^2 c(x,y,t)}{\partial x^2} + \frac{\partial^2 c(x,y,t)}{\partial y^2}\right) - v_y \frac{\partial c(x,y,t)}{\partial y} + Q(x,y,t) \tag{88}$$

If the guessed solution is of certain shape, obeying to the rules in [6], the source term is used to hold the equality of the equation. The following prescribed solution obeys these rules:

$$c(x,y,t) = c_0 - c_0 \sin^2\left(\frac{\pi x}{s}\right)\sin^2\left(\frac{\pi y}{y_e}\right)e^{-t} \tag{89}$$

If we substitute (89) in equation (88) and assume, for ease, v_y is independent of x, so a constant, we get:

$$\begin{aligned} c_0 \sin^2\left(\frac{\pi x}{s}\right)\sin^2\left(\frac{\pi y}{y_e}\right)e^{-t} = &- D2c_0\pi^2\left(\frac{\cos\left(\frac{2\pi x}{s}\right)\sin^2\left(\frac{\pi y}{y_e}\right)}{s^2} + \frac{\sin^2\left(\frac{\pi x}{s}\right)\cos\left(\frac{2\pi y}{y_e}\right)}{y_e^2}\right)e^{-t} \\ &+ v_y c_0 \pi\left(\frac{\sin^2\left(\frac{\pi x}{s}\right)\sin\left(\frac{2\pi y}{y_e}\right)}{y_e}\right)e^{-t} + Q(x,y,t) \end{aligned} \tag{90}$$

Rearranging:

$$\begin{aligned} Q(x,y,t) = \Bigg[c_0 \sin^2\left(\frac{\pi x}{s}\right)\sin^2\left(\frac{\pi y}{y_e}\right) + D2c_0\pi^2\left(\frac{\cos\left(\frac{2\pi x}{s}\right)\sin^2\left(\frac{\pi y}{y_e}\right)}{s^2} + \frac{\sin^2\left(\frac{\pi x}{s}\right)\cos\left(\frac{2\pi y}{y_e}\right)}{y_e^2}\right) \\ - v_y c_0 \pi\left(\frac{\sin^2\left(\frac{\pi x}{s}\right)\sin\left(\frac{2\pi y}{y_e}\right)}{y_e}\right)\Bigg] e^{-t} \end{aligned} \tag{91}$$

A discrete version of the source term $Q(x,y,t)$ in (91) could now be used in the numerical problem. Notice that $c(x,y,t)$ from the guessed solution (89) is equal to c_0 when $x = 0$, $x = s$, $y = 0$ and $y = y_e$ and furthermore $\frac{\partial c(x,y,t)}{\partial x} = 0$ in $x = 0$, $x = s$ and $\frac{\partial c(x,y,t)}{\partial y} = 0$ in $y = 0$ and $y = y_e$. To stick as close as possible to the original problem the boundary conditions $c(s,y,t) = c_0$, $c(x,0,t) = c_0$, $\frac{\partial c(0,y,t)}{\partial x} = 0$ and $\frac{\partial c(x,y_e,t)}{\partial y} = 0$ are implemented. The solution of the numerical problem with the given source term and boundary conditions should be an approximation of the guessed solution from (89). Now it is possible for this solution to compare the exact solution and the numerical solution and determine the orders of the error of this numerical scheme.

The definition of the L_2-error in (50) can be extended to two dimensions.

$$L_2 = \sqrt{\frac{\sum_{i=1}^{N_x}\sum_{i=1}^{N_y}\left(c(x_i,y_j,t) - \boldsymbol{c}_{ij}\right)^2}{N_x N_y}} \tag{92}$$

For the completeness of the story, the L_2-error is computed for the single grid case at first. For the values $s = 1.5$, $y_e = -300s$, $\frac{-\rho g}{2\mu} = -0.5$, $t = 100$ and $\delta t = 1 \cdot 10^{-4}$ figure 24 is generated as a reference to the single grid case.

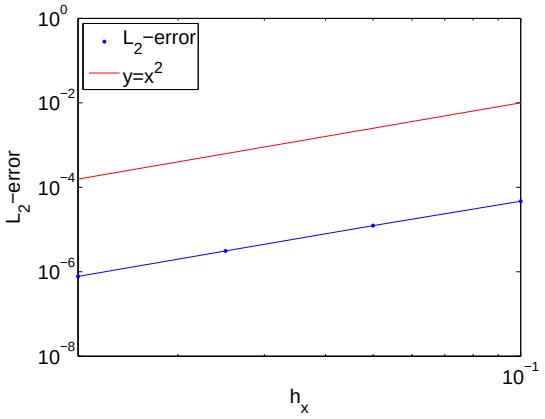
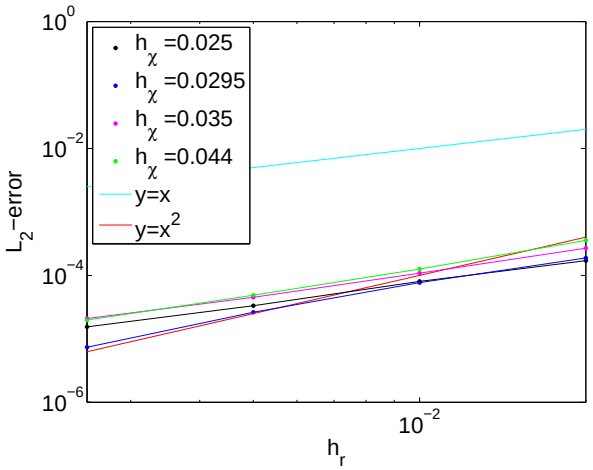
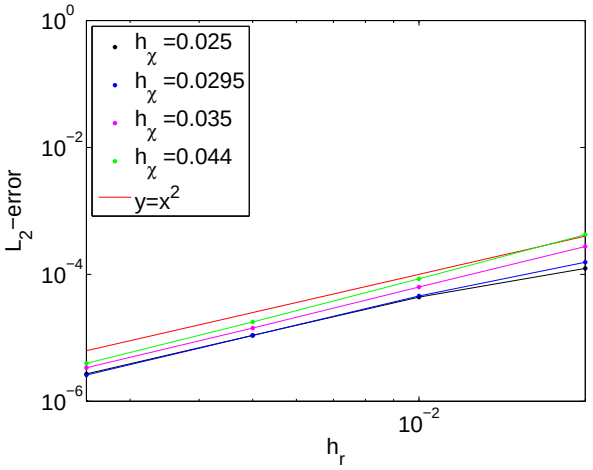


Figure 23: The L_2-error for $t = 100$; the smallest value of $h_x = 0.0125$, the smallest value of $h_y = -7.5$ and $\delta t = 1 \cdot 10^{-4}$

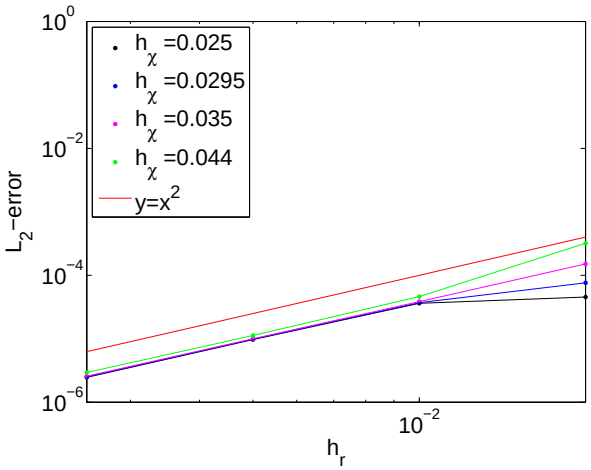
From figure 23 it can be concluded for the overlapping grids that in case of perfect coupling at maximum second order behavior of the L_2-error can be expected. With the values $s = 1.5$, $y_e = -300s$, $r_0 = 1.4$ (the starting point of the r-grid), $\frac{-\rho g}{2\mu} = -0.5$, $t = 100$ and $\delta t = 1 \cdot 10^{-4}$ the L_2-error behavior is determined for different orders of interpolation polynomials (the same order in both directions). Figure 24 shows the results of the simulations.



(a) 1st order polynomials



(b) 2nd order polynomials



(c) 3rd order polynomials

Figure 24: The L_2-error for $t = 100$; the smallest value of $h_r = 0.0025$, the smallest value of $h_y = -15$ and $\delta t = 1 \cdot 10^{-4}$

From figure 24 it can be concluded that at least a polynomial of second order is required to get second order L_2-error behavior. This is the same conclusion as for the one-dimensional problem. In the points the most on the right in figures 24b and 24c the Richardson error estimation formula does not hold anymore. The values for h_x, h_r and/or h_y are too large to make the assumption that the higher order terms in the Richardson error estimation formula may be ignored.

5. THE 2D PROBLEM WITH CIRCULAR PARTICLES

With everything done above it should now be possible to solve the convection-diffusion equation for two stationary particles. The basic idea is to wrap around the plates from the problem above, as illustrated in figure 22, so they form two particles on a concentration $c = c_0$. The r-grid will stick to the plates and thus wrap around those two particles. The basic setup of the problem, including the boundary conditions, is illustrated in figure 25. Note the problem is again symmetrical in the red line.

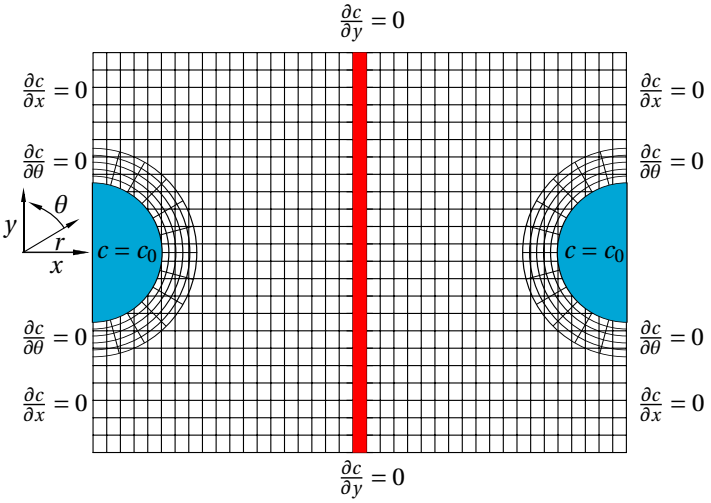


Figure 25: The setup for the 2D problem with circular particles

Now the r-grid is circular it is not the easiest way to describe it with Cartesian coordinates anymore. It is much easier to describe the grid in polar coordinates. However the drawback of using polar coordinates is that the convection-diffusion equation also must be rewritten in polar coordinates.

Another new problem that must be dealt with is the coupling. In all the problems above, it was needed to have only two Lagrange interpolation polynomials (one in every direction), because the x-coordinate of the coupling points was the same for every value of y. In this problem it will be a much more difficult task to localize the coupling points and to develop a proper coupling method for them.

5.1. THE CONVECTION-DIFFUSION EQUATION IN POLAR COORDINATES

As stated above, the convection-diffusion equation must be developed in polar coordinates. The expressions for x and y in polar coordinates are given by:

$$x(r,\theta) = r\cos\theta \quad y(r,\theta) = r\sin\theta \tag{93}$$

$$c(x,y) = c(r\cos\theta, r\sin\theta) \tag{94}$$

Then from the chain rule of differentiation is follows:

$$\begin{cases} \dfrac{\partial c}{\partial r} = \dfrac{\partial c}{\partial x}\dfrac{\partial x}{\partial r} + \dfrac{\partial c}{\partial y}\dfrac{\partial y}{\partial r} \\ \dfrac{\partial c}{\partial \theta} = \dfrac{\partial c}{\partial x}\dfrac{\partial x}{\partial \theta} + \dfrac{\partial c}{\partial y}\dfrac{\partial y}{\partial \theta} \end{cases} \tag{95}$$

Further from (93):

$$\frac{\partial x}{\partial r} = \cos\theta \quad \frac{\partial y}{\partial r} = \sin\theta \quad \frac{\partial x}{\partial \theta} = -r\sin\theta \quad \frac{\partial y}{\partial \theta} = r\cos\theta \tag{96}$$

Substitution leads to:

$$\begin{cases} \dfrac{\partial c}{\partial r} = \cos\theta\dfrac{\partial c}{\partial x} + \sin\theta\dfrac{\partial c}{\partial y} \\ \dfrac{\partial c}{\partial \theta} = -r\sin\theta\dfrac{\partial c}{\partial x} + r\cos\theta\dfrac{\partial c}{\partial y} \end{cases} \tag{97}$$

And in matrix notation:

$$\begin{pmatrix} \dfrac{\partial c}{\partial r} \\ \dfrac{\partial c}{\partial \theta} \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -r\sin\theta & r\cos\theta \end{bmatrix} \begin{pmatrix} \dfrac{\partial c}{\partial x} \\ \dfrac{\partial c}{\partial y} \end{pmatrix} \tag{98}$$

Multiplication with the inverse of the matrix of both left and right sides lead to:

$$\begin{pmatrix} \frac{\partial c}{\partial x} \\ \frac{\partial c}{\partial y} \end{pmatrix} = \frac{1}{r} \begin{bmatrix} r\cos\theta & -\sin\theta \\ r\sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} \frac{\partial c}{\partial r} \\ \frac{\partial c}{\partial \theta} \end{pmatrix} \tag{99}$$

Or:

$$\begin{cases} \frac{\partial c}{\partial x} = \cos\theta \frac{\partial c}{\partial r} - \frac{\sin\theta}{r} \frac{\partial c}{\partial \theta} \\ \frac{\partial c}{\partial y} = \sin\theta \frac{\partial c}{\partial r} + \frac{\cos\theta}{r} \frac{\partial c}{\partial \theta} \end{cases} \tag{100}$$

Now $\frac{\partial c}{\partial x}$ and $\frac{\partial c}{\partial y}$ can be treated the same as c, so be differentiated with respect to r and θ, to find the expressions for $\frac{\partial^2 c}{\partial x^2}$ and $\frac{\partial^2 c}{\partial y^2}$ in polar coordinates. This will lead to:

$$\frac{\partial^2 c}{\partial x^2} = \cos\theta \frac{\partial}{\partial r}\left(\cos\theta \frac{\partial c}{\partial r} - \frac{\sin\theta}{r}\frac{\partial c}{\partial \theta}\right) - \frac{\sin\theta}{r}\frac{\partial}{\partial \theta}\left(\cos\theta \frac{\partial c}{\partial r} - \frac{\sin\theta}{r}\frac{\partial c}{\partial \theta}\right) \tag{101}$$

$$= \cos^2\theta \frac{\partial^2 c}{\partial r^2} + \frac{\sin^2\theta}{r^2}\frac{\partial^2 c}{\partial \theta^2} + \frac{\sin^2\theta}{r}\frac{\partial c}{\partial r} + \frac{2\cos\theta\sin\theta}{r^2}\frac{\partial c}{\partial \theta} - \frac{2\cos\theta\sin\theta}{r^2}\frac{\partial^2 c}{\partial r \partial \theta} \tag{102}$$

and

$$\frac{\partial^2 c}{\partial y^2} = \sin^2\theta \frac{\partial^2 c}{\partial r^2} + \frac{\cos^2\theta}{r^2}\frac{\partial^2 c}{\partial \theta^2} + \frac{\cos^2\theta}{r}\frac{\partial c}{\partial r} - \frac{2\cos\theta\sin\theta}{r^2}\frac{\partial c}{\partial \theta} + \frac{2\cos\theta\sin\theta}{r^2}\frac{\partial^2 c}{\partial r \partial \theta} \tag{103}$$

So that

$$\frac{\partial^2 c}{\partial x^2} + \frac{\partial^2 c}{\partial y^2} = \frac{\partial^2 c}{\partial r^2} + \frac{1}{r^2}\frac{\partial^2 c}{\partial \theta^2} + \frac{1}{r}\frac{\partial c}{\partial r} \tag{104}$$

And the convection-diffusion equation can be written in polar coordinates as

$$\frac{\partial c}{\partial t} = D\left(\frac{\partial^2 c}{\partial r^2} + \frac{1}{r^2}\frac{\partial^2 c}{\partial \theta^2} + \frac{1}{r}\frac{\partial c}{\partial r}\right) - v_x\left(\cos\theta \frac{\partial c}{\partial r} - \frac{\sin\theta}{r}\frac{\partial c}{\partial \theta}\right) - v_y\left(\sin\theta \frac{\partial c}{\partial r} + \frac{\cos\theta}{r}\frac{\partial c}{\partial \theta}\right) \tag{105}$$

The only remaining issue is to find expressions for v_x and v_y. It does not matter if these expressions are in the basis of x and y of r and θ, because (93) can transform to the other basis.

5.2. The stream function

In equation (105) the terms v_x and v_y still have to be specified. Define $\boldsymbol{V} = (v_x, v_y)$. Assuming conservation of mass along with the assumption of a constant mass density, i.e. the fluid is incompressible, yields $\nabla \cdot \boldsymbol{V} = 0$. The steam function ψ is introduced that satisfies:

$$v_x = \frac{\partial \psi}{\partial y} \quad \text{and} \quad v_y = -\frac{\partial \psi}{\partial x} \tag{106}$$

Further assuming that the fluid is inviscid, the flow can said to be irrotational: $\nabla \times \boldsymbol{V} = 0$. This means that the stream function ψ satisfies Laplace's equation:

$$\nabla^2 \psi = 0 \tag{107}$$

The easiest way to solve this equation is to work with polar coordinates so v_x and v_y are expressed in polar coordinates suitable for equation (105). The first step is to rewrite the Laplace equation in polar coordinates and to find out how v_x and v_y can be derived from ψ. From (104) it can be concluded directly that

$$\nabla^2 \psi = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = \frac{\partial^2 \psi}{\partial r^2} + \frac{1}{r^2}\frac{\partial^2 \psi}{\partial \theta^2} + \frac{1}{r}\frac{\partial \psi}{\partial r} = 0 \tag{108}$$

is the expression of the Laplace equation in polar coordinates. From (93), the definition of x and y in polar coordinates, the relation between the components of the velocity in Cartesian coordinates and the components of the velocity in polar coordinates can be found.

$$\frac{\partial \psi}{\partial y} = v_x = \frac{\partial x}{\partial t} = \frac{\partial r(t)\cos\theta(t)}{\partial t} = \cos\theta \frac{\partial r}{\partial t} - r\sin\theta \frac{\partial \theta}{\partial t} = \cos\theta u_r - \sin\theta u_\theta \tag{109}$$

$$-\frac{\partial \psi}{\partial x} = v_y = \frac{\partial y}{\partial t} = \frac{\partial r(t)\sin\theta(t)}{\partial t} = \sin\theta \frac{\partial r}{\partial t} + r\cos\theta \frac{\partial \theta}{\partial t} = \sin\theta u_r + \cos\theta u_\theta \tag{110}$$

Where

$$u_r = \frac{\partial r}{\partial t} \quad \text{and} \quad u_\theta = r\frac{\partial \theta}{\partial t} \tag{111}$$

u_r is the velocity in the r-direction, the radial velocity, and u_θ the velocity in the θ-direction, also called the tangential velocity. The gradient of ψ is defined

$$\nabla\psi = \frac{\partial \psi}{\partial x}\hat{\boldsymbol{\imath}} + \frac{\partial \psi}{\partial y}\hat{\boldsymbol{\jmath}} = -v_y\hat{\boldsymbol{\imath}} + v_x\hat{\boldsymbol{\jmath}} \tag{112}$$

From the results of equation (100) we have our expressions for $\frac{\partial \psi}{\partial x}$ and $\frac{\partial \psi}{\partial y}$. These expressions can be substituted in the expression for the gradient as from (109) and (110) the results can be substituted for v_x and v_y.

$$\nabla\psi = \left(\cos\theta\frac{\partial \psi}{\partial r} - \frac{\sin\theta}{r}\frac{\partial \psi}{\partial \theta}\right)\hat{\boldsymbol{\imath}} + \left(\sin\theta\frac{\partial \psi}{\partial r} + \frac{\cos\theta}{r}\frac{\partial \psi}{\partial \theta}\right)\hat{\boldsymbol{\jmath}} = -\left(\sin\theta u_r + \cos\theta u_\theta\right)\hat{\boldsymbol{\imath}} + \left(\cos\theta u_r - \sin\theta u_\theta\right)\hat{\boldsymbol{\jmath}} \tag{113}$$

$$\Rightarrow \quad \left(\cos\theta\hat{\boldsymbol{\imath}} + \sin\theta\hat{\boldsymbol{\jmath}}\right)\frac{\partial \psi}{\partial r} + \frac{1}{r}\left(\cos\theta\hat{\boldsymbol{\jmath}} - \sin\theta\hat{\boldsymbol{\imath}}\right)\frac{\partial \psi}{\partial \theta} = \left(\cos\theta\hat{\boldsymbol{\jmath}} - \sin\theta\hat{\boldsymbol{\imath}}\right)u_r - \left(\sin\theta\hat{\boldsymbol{\jmath}} + \cos\theta\hat{\boldsymbol{\imath}}\right)u_\theta \tag{114}$$

$$\Rightarrow \quad \frac{\partial \psi}{\partial r}\hat{\boldsymbol{r}} + \frac{1}{r}\frac{\partial \psi}{\partial \theta}\hat{\boldsymbol{\theta}} = u_r\hat{\boldsymbol{\theta}} - u_\theta\hat{\boldsymbol{r}} \tag{115}$$

So there may be concluded that

$$u_r = \frac{1}{r}\frac{\partial \psi}{\partial \theta} \quad \text{and} \quad u_\theta = -\frac{\partial \psi}{\partial r} \tag{116}$$

Unfortunately it is not possible to solve the stream function. The reason for this is that the domain has both circular and Cartesian boundaries. The conclusion is that it is not possible to obtain an analytical solution for the flow profile. An option would be to solve the Laplace equation numerically and from there calculate the flow profile. This would require some additional time, which is not available for this research. Another option would now be to just choose a random flow profile to test the numerical scheme. The flow profile that is everywhere equal to zero is chosen and just the diffusion equation for the setup in figure 25 is solved.

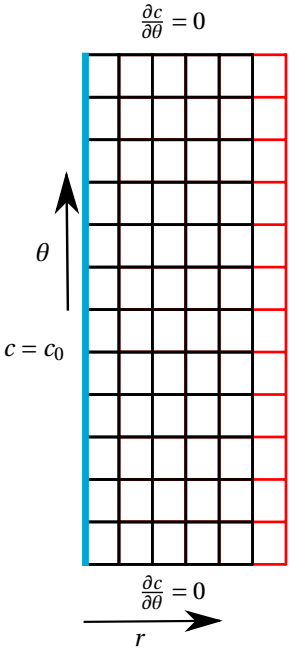


Figure 26: A sketch of the θ, r-grid

5.3. Solving the diffusion equation on a circular grid

Now we return to the setup in figure 25. It is our goal to solve the diffusion equation, while the convective term is chosen to be zero, on the circular grid as on the Cartesian grid. For the Cartesian grid the convection-diffusion equation was solved in section 4. On the circular grid the diffusion equation is given by:

$$\frac{\partial c}{\partial t} = D\left(\frac{\partial^2 c}{\partial r^2} + \frac{1}{r^2}\frac{\partial^2 c}{\partial \theta^2} + \frac{1}{r}\frac{\partial c}{\partial r}\right) \tag{117}$$

After this transformation the circular grid is transformed in a Cartesian grid, see figure 26. Second order cental discretization of the terms on the right side of the diffusion equation leads upon:

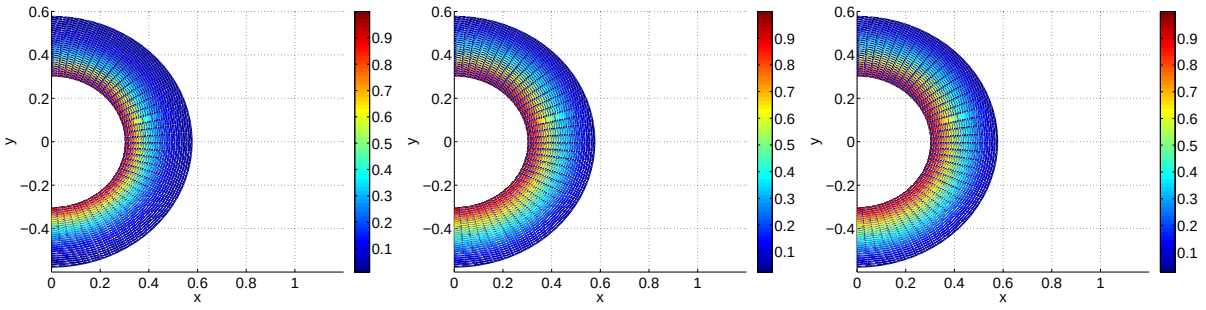
$$\frac{\partial c}{\partial t} \approx D\left(\frac{\boldsymbol{c}_{i+1} - 2\boldsymbol{c}_{i-1} + \boldsymbol{c}_i}{h_r^2} + \frac{1}{\boldsymbol{r}_j^2}\frac{\boldsymbol{c}_{j+1} - 2\boldsymbol{c}_j + \boldsymbol{c}_{j-1}}{h_\theta^2} + \frac{1}{\boldsymbol{r}_i}\frac{\boldsymbol{c}_{i+1} - \boldsymbol{c}_{i-1}}{2h_r}\right) \tag{118}$$

The first and the third term solely depend on r and thus can be converted in a matrix representation as was done in section 4.1 for the Cartesian grid case; call this matrix $\boldsymbol{A}_r$. The second term depends on θ and on r, but if the factor $\frac{1}{\boldsymbol{r}_i^2}$ is ignored for the moment the remainder of the second term depends only on θ and can be converted to a matrix; call this matrix $\boldsymbol{A}_\theta$. The total matrix for the circular grid is then given by $\boldsymbol{A}_{\theta,r} = \text{kron}\left(\boldsymbol{I}_r^{'}, \boldsymbol{A}_\theta\right) + \text{kron}\left(\boldsymbol{A}_r, \boldsymbol{I}_\theta\right)$. Here I_θ is just the identity matrix, but $I_r^{'}$ contains the values of $\frac{1}{\boldsymbol{r}_j^2}$ on its diagonal so $\boldsymbol{A}$ is the matrix representation of equation (118). Call N_θ the number of grid points in the θ-direction. It should be clear that the first N_θ rows of $\boldsymbol{A}$ account for the same values of r; in this case the $\boldsymbol{r}_1$ value closest to the particle. While there are only Neumann boundary conditions in the θ-direction, the boundary vector $\boldsymbol{b}$, with only the first N_θ unequal to zero, is given by:

$$\boldsymbol{b}_{2D} = Dc_0 \begin{pmatrix} \frac{1}{h_r^2} + \frac{1}{\boldsymbol{r}_1 2h_r} \\ \vdots \\ \frac{1}{h_r^2} + \frac{1}{\boldsymbol{r}_1 2h_r} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{119}$$

As it can be seen it is assumed that the particle is on a concentration c_0 and that the boundary condition in r_e, which is the red region of figure 26 and the biggest value of the r-grid, is zero.

For $a = 0.3$, $r_e = 0.58$ the problem is solved with the Modified Euler method for $r = a, ..., r_e$ and $\theta = \frac{-\pi}{2}, ..., \frac{\pi}{2}$. Here $h_r = 0.01$, $h_\theta = \frac{\pi}{40}$ and $\delta t = 1 \cdot 10^{-4}$. With zero concentration everywhere at $t = 0$ the following results are obtained at various times:



(a) $t = 1$ (b) $t = 3$ (c) $t = 5$

Figure 27: The concentration plot on the circular grid for various times; $h_r = 0.01$, $h_\theta = \frac{\pi}{40}$ and $\delta t = 1 \cdot 10^{-4}$

The next step is to check if just as on the Cartesian grid the L_2-error behavior is of second order. While all the discretizations are of second order there is no reason to expect the error behavior anything else than second order. As in section 4.5 an arbitrary function could be chosen to be plugged in the diffusion equation with source term. The resulting source term could then be used in the numerical problem. The code can then be verified by comparing the numerical solution with the chosen function. There are only two things that work slightly different than in section 4.5. In the first place the boundaries in section 4.5 where all Cartesian boundaries, which meant a Cartesian function with time independent boundary conditions could be chosen, equation (89). For this problem there are no Cartesian boundaries, so if the function in (89) is chosen that leads upon time dependent boundary conditions for the θ, r-grid. The consequence of the these time dependent boundary conditions is that the boundary condition vector $\boldsymbol{b}$ must be updated for every time step. For the reason that hereafter the order of the total problem is examined, equation (89) with the time dependent boundary conditions is used. The second thing that changes is that the definition of the L_2-error in polar coordinates is slightly different. The definition of the L_2-error in polar coordinates is given by:

$$L_2 = \sqrt{\frac{\sum_{j=1}^{N_\theta} \sum_{i=1}^{N_r} \left(c(r_i, \theta_j, t) - \boldsymbol{c}_{ij}\right)^2 r_i}{N_\theta N_r}} \tag{120}$$

$c(r_i, \theta_j, t)$ is the exact solution, $\boldsymbol{c}_{ij}$ the interpolated value, N_θ and N_r the number of points in the θ- respectively r-direction. The main difference is the factor r_i in the sum. This factor is caused by the Jacobian for the transformation from Cartesian coordinates to polar coordinates.

So with the time dependent boundary conditions at a and r_e and the source term in equation (91) with $v_y = 0$, of course all after substitution of the definitions from (93), the numerical problem is solved for $a = 0.3$, $r_e = 0.5$, $h_r = 0.0025; 0.005; 0.01; 0.02$, $h_\theta = \frac{\pi}{80}; \frac{\pi}{40}; \frac{\pi}{20}; \frac{\pi}{10}$ and $\delta t = 1 \cdot 10^{-4}$. This result is compared with the prescribed function according to the definition of the L_2-error in (120). Figure 28 shows the second order behavior that was expected from the theory.

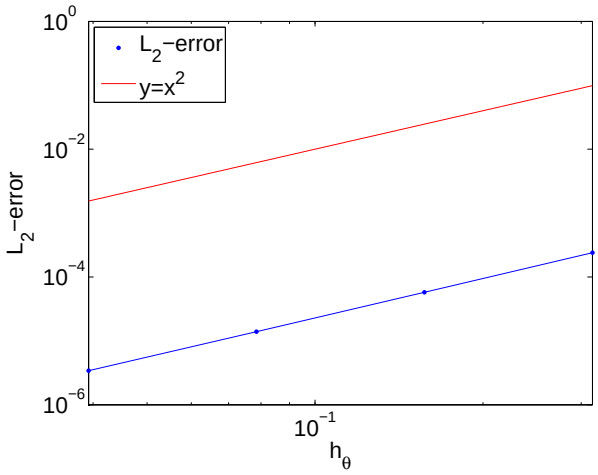


Figure 28: The L_2-error at $t = 0.1$ for $a = 0.3$, $r_e = 0.5$; $h_r = 0.0025; 0.005; 0.01; 0.02$, $h_\theta = \frac{\pi}{80}; \frac{\pi}{40}; \frac{\pi}{20}; \frac{\pi}{10}$ and $\delta t = 1 \cdot 10^{-4}$

Now the diffusion equation has been solved on the Cartesian grid and on the circular grid separated both with second order error behavior. What rests is the coupled problem where the diffusion equation is solved on the Cartesian and circular grid in one go. In the next section the coupling mechanism is initiated.

5.4. BILINEAR INTERPOLATION

Where there are two grids, the x, y-grid and the θ, r-grid, some communication between those grids is needed. Consider figure 29. The red dots represent grid points of the θ, r-grid. The green dot is a point of the x, y-grid that is overlapped by the θ, r-grid. Assume the concentrations in the θ, r-grid, thus the red dots, are all known. The concentration in the x, y-grid, the green dot, can now be interpolated from the concentrations in the θ, r-grid in the following manner. The first step is linear interpolation in the θ-direction:

$$\mathscr{L}_1 = \frac{\theta_2 - \theta_{\mathscr{L}}}{\theta_2 - \theta_1} c_{11} + \frac{\theta_{\mathscr{L}} - \theta_1}{\theta_2 - \theta_1} c_{21} \tag{121}$$

$$\mathscr{L}_2 = \frac{\theta_2 - \theta_{\mathscr{L}}}{\theta_2 - \theta_1} c_{12} + \frac{\theta_{\mathscr{L}} - \theta_1}{\theta_2 - \theta_1} c_{22} \tag{122}$$

After that linear interpolation in the r-direction between $\mathscr{L}_1$ and $\mathscr{L}_2$ will result in the concentration in $\mathscr{L}$:

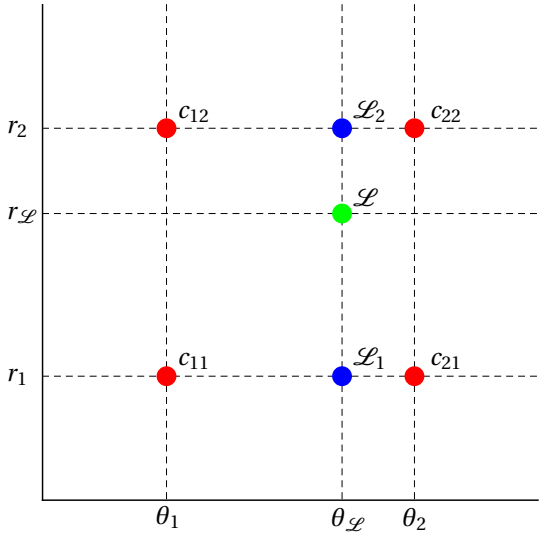


Figure 29: Bilinear interpolation

$$\mathscr{L} = \frac{r_2 - r_{\mathscr{L}}}{r_2 - r_1} \mathscr{L}_1 + \frac{r_{\mathscr{L}} - r_1}{r_2 - r_1} \mathscr{L}_2 \tag{123}$$

It will be clear that the total interpolation is the product of two linear functions. From the derivation of the linear interpolation error in section 3.8 it follows that the the product of the two linear functions has error

$\mathcal{O}\left(\min\left\{h_\theta^2, h_r^2\right\}\right)$. As seen before at least interpolation with a polynomial of second order ($\mathcal{O}(h^3)$) can result in a second order error behavior. So it seems somewhat counter-intuitive to continue with bilinear interpolation. So in advance it is already possible to say that the L_2-error will not have second order behavior, but as earlier mentioned the time span for this research is limited. Interpolation with 9, second order polynomials, or even 16, third order polynomials, is certainly preferable to use, but would simply cost too much time to implement. Bilinear interpolation will be used to interpolate the concentrations from the θ, r-grid to the x, y-grid as well as to interpolate the concentrations from the x, y-grid to the θ, r-grid.

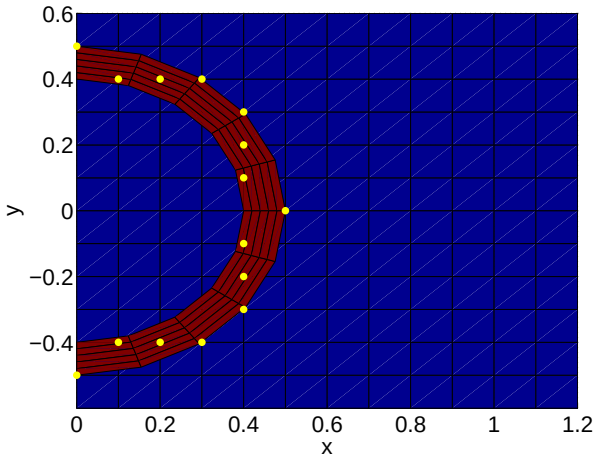
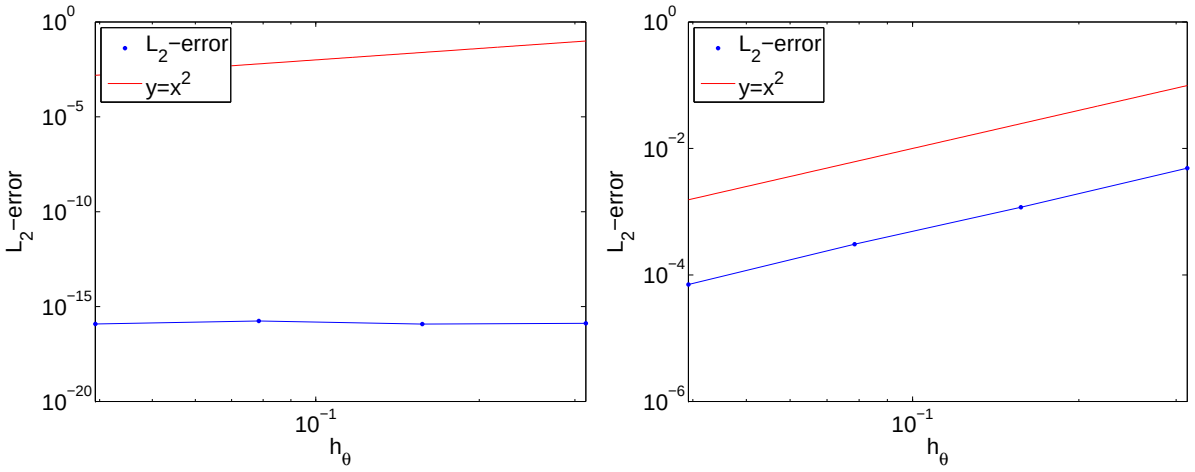


Figure 30: The boundary points of the x, y-grid that lie inside the θ, r-grid, $a = 0.4, r_e = 0.5, h_x = h_y = 0.1, h_r = 0.02$ & $h_\theta = \frac{\pi}{10}$

To check if the bilinear interpolation is well implemented in the program that solves equation (117), the bilinear interpolation can be checked separately. Consider in figure 30 an arbitrary case of the setup in figure 25. The yellow dots are points of the x, y-grid that lie inside the θ, r-grid. The bilinear interpolation works as follows. Consider one particular point of the x, y-grid that is marked with a yellow dot. The coordinates of this point will be transformed to polar coordinates. Then the bilinear interpolation from the θ, r-grid to this point is in terms of θ and r. The four points of the θ, r-grid that surround the considered point will then be used for the interpolation. A function that is linear in θ and r should interpolate exactly, so with error equal to zero, from the θ, r-grid to the points marked with yellow dots in the x, y-grid. Any non-linear function in θ and/or r should be interpolated with an error behavior of second order.

The preceding idea is implemented for $a = 0.3$, $r_e = 0.58$, $s = 1.5$ $y_0 = -s$ and $y_e = s$ and with grid spacings $h_x = h_y = 0.05$, $h_\theta = \frac{\pi}{80}; \frac{\pi}{40}; \frac{\pi}{20}; \frac{\pi}{10}$ and $h_r = 0.005; 0.01; 0.02; 0.04$. For figure 31a the function $r + \theta$ is interpolated from the θ, r-grid to the x, y-grid. The exact values of the function in the grid points of the x, y-gird is prescribed to those points. With the definition of the L_2-error from equation (92) the interpolated values in all the boundary points of the x, y-grid are compared with the exact values. As predicted for linear functions figure 31a gives (ignoring the MATLAB® machine epsilon) zero error. For figure 31b the function $r\cos(\theta) + r\sin(\theta)$ is interpolated from the θ, r-grid to the x, y-grid. This time the function is not linear and the interpolation gives second order error as predicted.

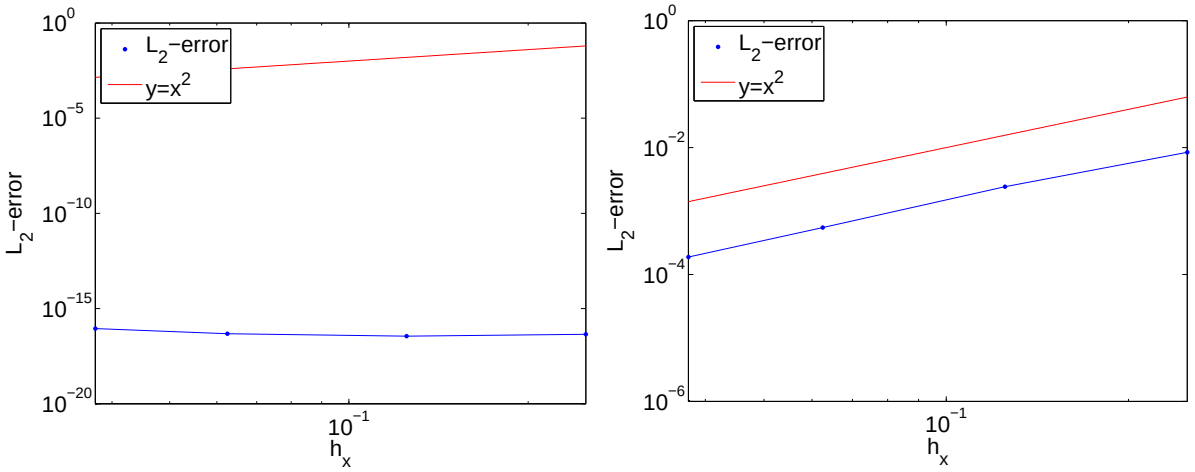
The same thing can be done in the opposite direction. The function $x + y$ is interpolated from the x, y-grid to boundary points of the θ, r-grid. The grid spacings are now $h_\theta = \frac{\pi}{100}$, $h_r = 0.005$ and $h_x = h_y = 0.0375; 0.0625; 0.125; 0.25$. With the definition of the L_2-error in polar coordinates, equation (120), the interpolated values are compared with the exact values in all the boundary points of the θ, r-grid. The result can be seen in figure 32a. Again the linear function gives zero error. For the result in figure 32b the function $\sqrt{x^2 + y^2} + \text{atan}\left(\frac{y}{x}\right)$ is interpolated. This non-linear function leads again to second order error behavior. It can be concluded that the interpolation in both directions works as it should.



(a) The linear function $r+\theta$ is interpolated

(b) The non-linear function $r\cos(\theta)+r\sin(\theta)$ is interpolated

Figure 31: The L_2-error for the interpolation from the θ, r-grid to the x, y-grid for a linear and a non-linear function; $h_x = h_y = 0.05$, $h_\theta = \frac{\pi}{80}; \frac{\pi}{40}; \frac{\pi}{20}; \frac{\pi}{10}$ and $h_r = 0.005; 0.01; 0.02; 0.04$



(a) The linear function $x+y$ is interpolated

(b) The non-linear function $\sqrt{x^2+y^2}+\text{atan}\left(\frac{y}{x}\right)$ is interpolated

Figure 32: The L_2-error for the interpolation from the x, y-grid to the θ, r-grid for a linear and a non-linear function; $h_\theta = \frac{\pi}{100}$, $h_r = 0.005$ and $h_x = h_y = 0.0375; 0.0625; 0.125; 0.25$

5.5. Finding boundary points and the corresponding interpolation points

Like in all earlier cases the problem is symmetrical, so only half of figure 25 can be used. Figure 33 shows the actual setup, where the origin lies in the center of the particle. As earlier mentioned the θ, r-grid can be treated in the same way as the x, y-grid. The only difference is that on the θ, r-grid equation (117) has to be solved where on the x, y-grid it is equation (62) that has to be solved. The (Neumann) boundary conditions for θ are directly obtained from the boundary conditions for x, the one Dirichlet boundary condition for r is obtained from the particle and the other, the one in the red region of figure 26, is obtained by bilinear interpolation from the x, y-grid. In the preceding section about bilinear interpolation there was already a little comment on the coupling, but the full mechanism will be explained below.

For the coupling there arise two problems: which points of the x, y-grid are considered boundary points and which points from the θ, r-grid are needed to interpolate to those boundary points? The first problem is tackled by an algorithm that selects all the points where it applies that $\sqrt{x^2+y^2} \leq r_e$. Then from this selection it selects for every y the points where x is maximal and for every x the points where $|y|$ is maximal. This algorithm ensures that every point of the x, y-grid for which $\sqrt{x^2+y^2} > r_e$, has four surrounding points, so the

numerical integration works. For $a = 0.4$, $r_e = 0.5$, $h_x = h_y = 0.1$, $h_r = 0.02$ and $h_\theta = \frac{\pi}{10}$ an example was shown in figure 30. As earlier mentioned the yellow dots are the boundary points of the x, y-grid. Some of the yellow dots lie outside the red region of the θ, r-grid, but those dots are actually on $r = 0.5$. So they may be considered as points that overlap the θ, r-grid.

The next step is to select four points, for every boundary point of the x, y-grid, in the θ, r-grid that can interpolate their concentration to that boundary point. This is again done by an algorithm. This algorithm works as follows. From the matrix of all the θ-coordinates of the points in the θ, r-grid it subtracts the θ-coordinate of the boundary point. The same thing is done for the r-coordinates. Then the absolute value of both results is summed to one big matrix. Now this new matrix can be sorted by MATLAB® along both directions. After sorting the values at $(1,1)$, $(1,2)$, $(2,1)$ and $(2,2)$ belong to the original points that surround the boundary point. A similar treatment is done by coupling from the x, y-grid to the θ, r-grid. It will be determined where the boundary points of the θ, r-grid lie inside the x, y-grid and which are the four points, that are used for interpolation, of the x, y-grid surround this boundary point.

As seen in section 3.8 it is preferable to solve the whole numerical integration problem, including the coupling, with one matrix $\boldsymbol{A}$. Although the idea is the same the representation of $\boldsymbol{A}$ is slightly different:

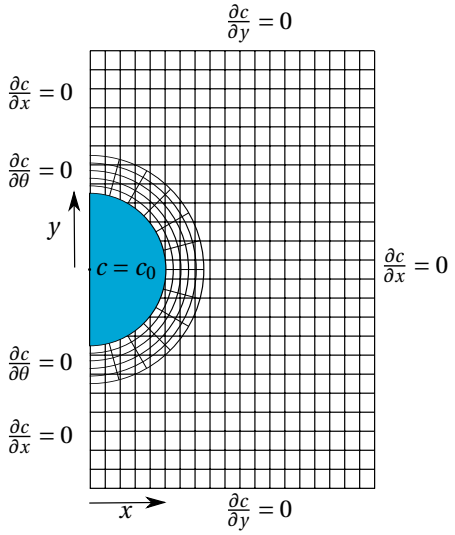
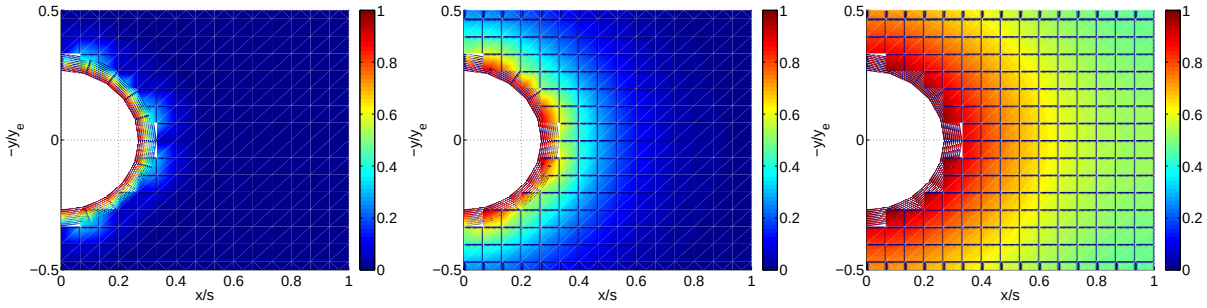


Figure 33: The grid of the whole problem

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{\theta,r} & \mathscr{L}_{x,y} \\ \mathscr{L}_{\theta,r} & \boldsymbol{A}_{x,y} \end{bmatrix} \tag{124}$$

Here $\boldsymbol{A}_{\theta,r}$ is the matrix from section 5.3 that solves the diffusion equation on the θ, r-grid, i.e. the matrix that contains the discretization of equation (117) and the boundary conditions for θ. $\boldsymbol{A}_{x,y}$ is the matrix that holds the discretization for all the points in the x, y-grid. $\mathscr{L}_{x,y}$ contains the interpolation coefficients from the x, y-grid to the θ, r-grid and $\mathscr{L}_{\theta,r}$ contains the interpolation coefficients from the θ, r-grid to the x, y-grid. In practice $\mathscr{L}_{x,y}$ is mixed up with $\boldsymbol{A}_{\theta,r}$ while some of the interpolation points (points used for interpolation) of the x, y-grid are boundary points to which is interpolated from the θ, r-grid. All entries of the rows and columns of $\boldsymbol{A}_{x,y}$ for the points that lie inside the θ, r-grid are equal to zero. Most of these points are not considered in the problem and the boundary points are represented by $\mathscr{L}_{\theta,r}$. The exact construction of $\boldsymbol{A}$ is quite an administrative task. One minimal condition to check if $\boldsymbol{A}$ is properly constructed is to add the entries along every row of matrix $\boldsymbol{A}$ plus the boundary vector $\boldsymbol{b}$. If the sum of the entries on each row is not equal to zero, there may be concluded that the representation of matrix $\boldsymbol{A}$ is incorrect.

If matrix $\boldsymbol{A}$ is constructed properly and boundary vector $\boldsymbol{b}$ contains the boundary concentrations $c = c_0$ for the concentration on the particle, again the method of Modified Euler can be used directly to solve the problem. Figure 34a, 34b and 34c are created with $a = 0.4$, $r_e = 0.5$, $s = y_0 = y_e = 1.5$, $h_\theta = \frac{\pi}{10}$, $h_r = 0.01$, $h_x = h_y = 0.1$ and $\delta t = 1 \cdot 10^{-4}$



(a) $t = 1$ (b) $t = 10$ (c) $t = 100$

Figure 34: Concentration profile of coupled circular and Cartesian grids with $a = 0.4$, $r_e = 0.5$, $s = y_0 = y_e = 1.5$, $h_\theta = \frac{\pi}{10}$, $h_r = 0.01$, $h_x = h_y = 0.1$ and $\delta t = 1 \cdot 10^{-4}$. The white region represents the points of the x, y-grid where the problem is not defined.

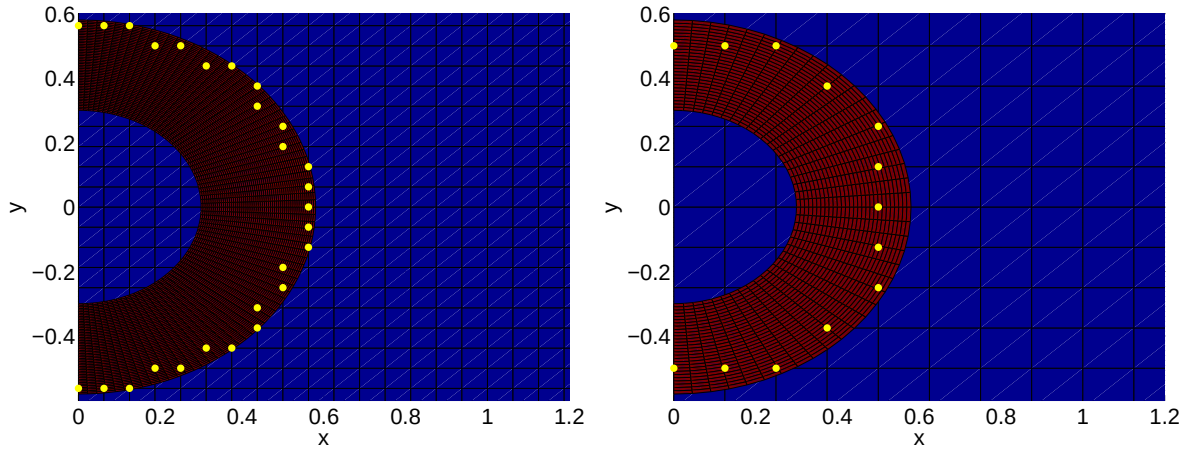
Another simple check for whether the code is correct is to start with an initial concentration of c_0 everywhere. As time progresses the solutions should not change. This is indeed the case to machine accuracy. For a final check of the code we need to look at the L_2-error of the coupled system.

5.6. THE L_2-ERROR

As before the results can be verified by comparing a numerical solution with an exact solution of the problem. As mentioned in section 5.3 equation (89) is used as the exact solution. This results in time dependent boundary conditions on the particle and time independent boundary conditions on the other boundaries. The actual L_2-error of the total problem is a combination of the definitions from equation (92), for the Cartesian grid, and (120), for the circular grid:

$$L_2 = \sqrt{\frac{\sum_{j=1}^{N_\theta}\sum_{i=1}^{N_r}\left(c(r_i,\theta_j,t)-\boldsymbol{c}_{ij}\right)^2 r_i + \sum_{k=1}^{N_x}\sum_{l=1}^{N_y}\left(c(x_k,y_l,t)-\boldsymbol{c}_{kl}\right)^2}{N_\theta N_r + N_{xy}}} \tag{125}$$

Where $c(r_i,\theta_j,t)$ and $c(x_k,y_l,t)$ are the exact solutions and $\boldsymbol{c}_{ij}$ and $\boldsymbol{c}_{kl}$ are the numerical solutions. N_{xy} are the number of grid points of the x,y-grid without the boundary points and the points of the x,y-grid that are excluded from the problem. Of course the boundary points and the points of the x,y-grid that are excluded from the problem do not contribute in the L_2-error, $\boldsymbol{c}_{ij}$ and $\boldsymbol{c}_{kl}$ are set zero for these points. Figure 35 shows four overlap situations, where the grid spacing are repeatedly doubled. These four cases are used to determine the L_2-error for various times.



(a) $h_\theta = \frac{\pi}{80}$, $h_r = 0.005$ and $h_x = h_y = 0.0625$

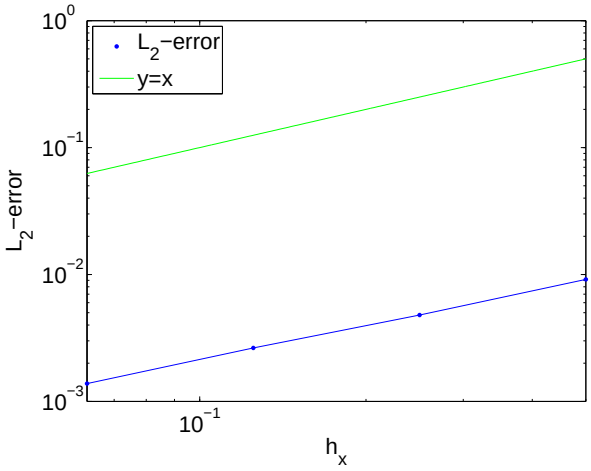
(b) $h_\theta = \frac{\pi}{40}$, $h_r = 0.01$ and $h_x = h_y = 0.125$

(c) $h_\theta = \frac{\pi}{20}$, $h_r = 0.02$ and $h_x = h_y = 0.25$

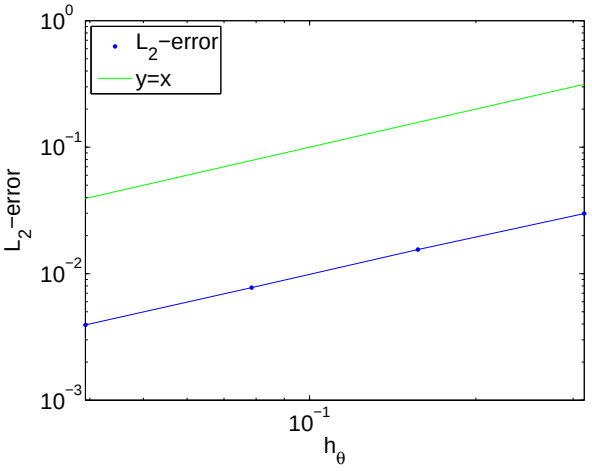
(d) $h_\theta = \frac{\pi}{10}$, $h_r = 0.04$ and $h_x = h_y = 0.5$

Figure 35: The yellow points are again the boundary points of the x,y-grid in the θ,r-grid. Figure 35a is generated with $a = 0.3$, $r_e = 0.58$, $h_\theta = \frac{\pi}{80}$, $h_r = 0.005$ and $h_x = h_y = 0.0625$. In figure 35b the values of h_θ, h_r, h_x and h_y are doubled according to the values in 35a, in figure 35c those values are doubled according to the values in 35b and so on.

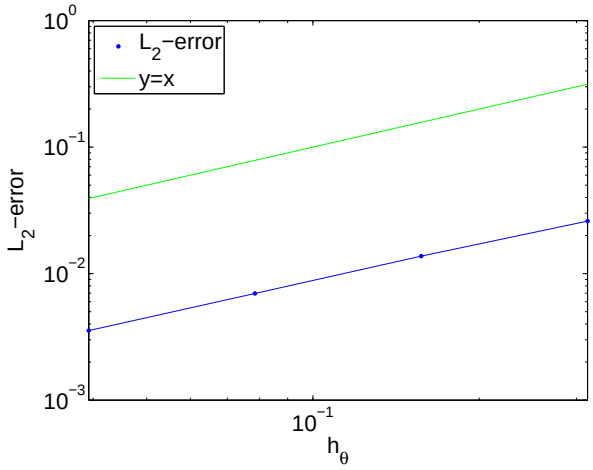
In section 3.8 it was derived that linear interpolation, i.e. interpolation with an error of $\mathcal{O}(h^2)$, will not lead to second order error behavior of the numerical problem and at least a second order interpolation polynomial, i.e. with an error of $\mathcal{O}(h^3)$, is needed for second order error behavior of the numerical problem. Bilinear interpolation has an error of order $\mathcal{O}(h^2)$. It can thus be predicted that the L_2-error behavior of the total problem is of first order to the utmost. For $t = 20$ and $\delta t = 1 \cdot 10^{-4}$ the L_2-error is determined for the cases from figure 35.



(a) x, y-grid with definition (92)



(b) θ, r-grid with definition (120)



(c) both grids with definition (125)

Figure 36: The L_2-error for the x, y-grid (figure 36a), for the θ, r-grid (figure 36b) and for both grids (36c; all for $t = 20$)

All figures show the first order L_2-error behavior that was theoretically predicted. As earlier stated it is due to the lack of time that it is not shown that by using nine points (3×3 points), which means second order interpolation polynomials, second order error behavior will be reached for the total problem.

6. CONCLUSION AND DISCUSSION

The main topic of this report was to find and verify a method to couple two grids without introducing an extra error caused by the coupling, i.e. the error behavior for the coupled grid must be the same as the error behavior for the single grid case. It was seen that for a second and a third order interpolation polynomial the error behavior for the coupled grid was second order as for the single grid. Linear interpolation led upon one order less in the error behavior, so it is not recommended to use linear interpolation or bilinear interpolation when all the terms of the convection-diffusion equation are discretized to second order.

Although not researched due to the lack of time, it will be sufficient to use nine points, 3×3 points, instead of the four points in the case of bilinear interpolation. The point that is interpolated to must lie somewhere in this 3×3 point region. The interpolation polynomials will then be of second order, which means an error of $\mathcal{O}(h^3)$, in both directions and from the results of this research it may be concluded that the total interpolation has also an error of $\mathcal{O}(h^3)$. So the use of this second order interpolation polynomial would lead to second order behavior in figure 36.

In the problem with the circular particles the flow profile was chosen to be zero everywhere, no flow. From the two-dimensional problem with the flat plates it can be concluded that a flow profile and thus a convective term does not change the error behavior as long as the discretization of the convective term is of second order. The inclusion of a convective term in the problem with the circular particles would cost some time to implement, but would not cause an error behavior different from second order. It should even be possible for the particles to move as long as for every discrete value of x and for every discrete value of y there is one grid point overlapping the circular grid. The boundary points of the x, y-grid should then be determined each integration step and thus every integration step a new matrix should be created.

Like the extension from the one-dimensional problem to the two-dimensional problem it should be possible to extend the conclusions from the two-dimensional problem to a three-dimensional problem. In that case 27 points, $3 \times 3 \times 3$, are needed to interpolate from to a single point in order to obtain second order error behavior. This would be quite cumbersome to program, but the biggest problem is the memory required for the storage of the matrices, although the use of sparse matrices would extremely decreases the amount of memory required.

Further research could still be done for the case that the spherical/circular grids of two particles overlap each other, which was the main topic of [2]. If the particles are at fixed positions and $r_e - a < 2s < (r_e - a)$ (the circular grids overlap, but do not overlap the particles), the coupling is still relatively simple. The concentrations of the circular grid can be coupled to the boundary points of the Cartesian grid that overlap the circular grid(s). The concentration should be averaged in the boundary points that overlap both grids. To the boundary points of a circular grid that lie inside the other circular grid is interpolated from that circular grid and to the other boundary points of the circular grid is interpolated from the Cartesian grid. This is a bit more complicated, but still solvable. When $2s < r_e - a$, the circular grid overlaps the other particle, the problem gets really complicated. The boundary points of the circular grids are no more on the same circle. If the particles are now also allowed to move and multiple particles are introduced the problem gets soon too complicated to implement and too big to solve.

BIBLIOGRAPHY

[1] J.J. Derksen, *Simulations of solid–liquid scalar transfer for a spherical particle in laminar and turbulent flow*, AIChE Journal, Vol 60, p1202, 2014.

[2] J.J. Derksen, *Simulations of solid–liquid mass transfer in fixed and fluidized beds*, Chemical Engineering Journal, Vol 255, p233, 2014.

[3] C. Vuik, P. van Beek, F. Vermolen, J. van Kan *Numerieke Methoden voor Differentiaalvergelijkingen*. VSSD, Delft, Eerste Druk, 2006.

[4] Harry van den Akker, Rob Mudde *Fysische Transportverschijnselen*. VSSD, Delft, Derde Druk, 2008

[5] Richard Haberman *Applied Partial Differential Equations*. Pearson, New Jersey, Derde Druk, 1998

[6] Patrick Knupp, Kambiz Salari *Verification of Computer Codes in Computational Science and Engineering*. Chapman & Hall/CRC, Boca Raton, Eerste Druk, 2003

[7] Theory about the numerical scheme in two dimensions:
https://www.youtube.com/watch?v=r1-r1t5i58g#t=154 23-6-2015

[8] Theory about sparse matrices:
https://en.wikipedia.org/wiki/Sparse_matrix 23-6-2015

[9] Theory about bilinear interpolation:
https://en.wikipedia.org/wiki/Bilinear_interpolation 23-6-2015

[10] Theory about Lax equivalence theorem:
https://en.wikipedia.org/wiki/Lax_equivalence_theorem 23-6-2015

[11] Theory about nondimensionalization:
http://banach.millersville.edu/~bob/math467/NonDimHeat.pdf 24-6-2015

APPENDIX

A. THE ANALYTICAL SOLUTION OF THE ONE-DIMENSIONAL PROBLEM

Using [5, p.35-50] as a guideline (16) can also be solved analytically. To start with the coordinate transform $y = x + s$ in order to reduce the math:

$$\begin{aligned} \frac{\partial c(y,t)}{\partial t} &= D\frac{\partial^2 c(y,t)}{\partial y^2},\ 0 < y < 2s,\ \forall\ 0 < t \\ c(2s,t) &= c(0,t) = c_0,\ \forall\ 0 < t \\ c(y,0) &= 0,\ \forall\ y \neq 0, 2s \end{aligned} \tag{126}$$

Actually it is easier to first solve the problem for $u(y,t) = c(y,t) - c_0$:

$$\begin{aligned} \frac{\partial u(y,t)}{\partial t} &= D\frac{\partial^2 u(y,t)}{\partial y^2},\ 0 < y < 2s,\ \forall\ 0 < t \\ u(2s,t) &= u(0,t) = 0,\ \forall\ 0 < t \\ u(y,0) &= -c_0,\ \forall\ y \neq 0, 2s \end{aligned} \tag{127}$$

A method called separation of variables is used to solve for $u(y,t)$. This method assumes that the solution can be written as a product of a solely time-dependent part and a solely spacial-dependent part: $u(y,t) = \phi(y)h(t)$. Now equation (127) can be rewritten as:

$$\frac{\partial \phi(y)h(t)}{\partial t} = D\frac{\partial^2 \phi(y)h(t)}{\partial y^2} \tag{128}$$

$$\Leftrightarrow \frac{1}{Dh(t)}\frac{\partial h(t)}{\partial t} = \frac{1}{\phi(y)}\frac{\partial^2 \phi(y)}{\partial y^2} = -\lambda \tag{129}$$

$\lambda > 0$ is called the separation constant and the minus sign is chosen for convenience as well as for the problem being physical. Solving for $h(t)$ gives:

$$h(t) = C_1 e^{-D\lambda t} \tag{130}$$

With C_1 an integration constant. Solving for $\phi(y)$ gives:

$$\phi(y) = C_2 \cos(\sqrt{\lambda}y) + C_3 \sin(\sqrt{\lambda}y) \tag{131}$$

With C_2 and C_3 arbitrary constants. Now the boundary conditions from (127) are used to determine the value of λ:

$$\phi(0) = C_2 \cos(\sqrt{\lambda}0) + C_3 \sin(\sqrt{\lambda}0) = 0 \tag{132}$$

$$\Rightarrow C_2 = 0 \tag{133}$$

$$\phi(2s) = C_3 \sin(\sqrt{\lambda}2s) = 0 \tag{134}$$

$$\Rightarrow \sqrt{\lambda}2s = n\pi \Leftrightarrow \lambda = \left(\frac{n\pi}{2s}\right)^2 \text{ with } n \in \mathbb{N} \tag{135}$$

From the principle of superposition now it can be concluded that the solution of (127) for $u(y,t)$ can be written as:

$$u(y,t) = \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi}{2s}y\right) e^{-D\left(\frac{n\pi}{2s}\right)^2 t} \tag{136}$$

The constants B_n can be determined with the aid of the initial condition:

$$u(y,0) = \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi}{2s}y\right) \tag{137}$$

$$u(y,0)\sin\left(\frac{m\pi}{2s}y\right) = \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi}{2s}y\right)\sin\left(\frac{m\pi}{2s}y\right) \tag{138}$$

$$\int_0^{2s} u(y,0)\sin\left(\frac{m\pi}{2s}y\right)dy = \sum_{n=1}^{\infty} B_n \int_0^{2s} \sin\left(\frac{n\pi}{2s}y\right)\sin\left(\frac{m\pi}{2s}y\right)dy \tag{139}$$

Considering the integral on the right hand side in the equation above, assume $n = m$:

$$\int_0^{2s} \sin\left(\frac{n\pi}{2s}y\right)\sin\left(\frac{m\pi}{2s}y\right)dy = \int_0^{2s} \sin^2\left(\frac{n\pi}{2s}y\right)dy \tag{140}$$

$$= \int_0^{2s} \left(\frac{1}{2} - \frac{1}{2}\cos\left(\frac{n\pi}{s}y\right)\right)dy \tag{141}$$

$$= \left[\frac{y}{2} - \frac{s}{2n\pi}\sin\left(\frac{n\pi}{s}y\right)\right]_0^{2s} \tag{142}$$

$$= s \tag{143}$$

Now assume $n \neq m$:

$$\int_0^{2s} \sin\left(\frac{n\pi}{2s}y\right)\sin\left(\frac{m\pi}{2s}y\right)dy = \frac{1}{2}\int_0^{2s}\left[\cos\left(\frac{(n-m)\pi}{s}y\right) - \cos\left(\frac{(n+m)\pi}{s}y\right)\right]dy \tag{144}$$

$$= \frac{1}{2}\left[\frac{s}{(n-m)\pi}\sin\left(\frac{(n-m)\pi}{s}y\right) - \frac{s}{(n+m)\pi}\sin\left(\frac{(n+m)\pi}{s}y\right)\right]_0^{2s} \tag{145}$$

$$= 0 \tag{146}$$

Computation of the left hand side of (139):

$$\begin{aligned}\int_0^{2s} u(y,0)\sin\left(\frac{m\pi}{2s}y\right)dy &= -c_0\int_0^{2s}\sin\left(\frac{m\pi}{2s}y\right)dy \\ &= c_0\left[\frac{2s}{m\pi}\cos\left(\frac{m\pi}{2s}y\right)\right]_0^{2s} \\ &= \frac{c_0 2s}{m\pi}\left[\cos(m\pi) - 1\right] \\ &= \begin{cases} -\dfrac{4c_0 s}{m\pi} & \text{if } m \text{ is odd} \\ 0 & \text{else} \end{cases}\end{aligned}$$

Now it can be concluded that the B_n in (136) can be written as:

$$B_n = \begin{cases} -\dfrac{4c_0}{n\pi} & \text{if } n \text{ is odd} \\ 0 & \text{else} \end{cases} \tag{147}$$

c_0 is added to $u(y,t)$ to solve for $c(y,t)$. After the inverse of the coordinate transform earlier done the result for $c(x,t)$ is:

$$c(x,t) = c_0 + \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi}{2s}(x+s)\right)e^{-D\left(\frac{n\pi}{2s}\right)^2 t} \tag{148}$$

Where the B_n are given by (147). There is no problem this exact solution being an infinite series, because the terms for n very large approximate zero. The term for $n = 10000$ is that small that MATLAB® returns 0, so terminating the summation after 10000 terms does not cause any extra error.

B. SPARSE MATRICES

An important property of a numerical system is the amount of time it costs to evaluate the problem. With the numerical scheme from section 4.2 equation (19) is integrated with Modified Euler. The input values that affect the computation time are as follows: $\delta t = 0.01,\ t_e = 200,\ s = 1.5,\ h_x = 0.1,\ h_y = -2,\ y_e = -300s$. The time it takes MATLAB® to solve the problem including initialization, i.e. specifying variables and building matrices, is about 617 seconds on my computer.

In the first attempt to tackle the $2D$ problem a direct approach with for-loops instead of matrices was used. Every time step the concentration in every grid point was calculated explicitly with the difference equations and recalculated with a Modified Euler correction step. Although this seems a rather lazy manner of solving the problem, it took my computer approximately 120 seconds, which is much less than the 617 seconds the numerical scheme from section 4.2 took. The striking difference in computation time can be explained by the fact that the matrix $\boldsymbol{A}_{2D}$ holds especially zeros. Every matrix multiplication all these zeros are also multiplied. In other words, for the computation of the concentration in a specific grid point every other gird point is taken into account. In the 'lazy' direct approach with for-loops only the direct neighbors and the grid point itself were taken into account. To rule out all those multiplications with 0 when $\boldsymbol{A}_{2D}$ is multiplied with the concentration vector $\boldsymbol{c}$, the theory of sparse matrices could be used.

A sparse matrix is a matrix in which most of its elements are zero. When such a matrix is stored on the memory of a computer it is useful not to save all elements but only the non-zero elements. The non-zero elements are stored in such a way that the row and column numbers of the elements are known. To store a matrix in this way saves a lot of memory that can be used for the computation and omits the multiplications with 0. In MATLAB® the command 'sparse$(\boldsymbol{A}_{2D})$' makes $\boldsymbol{A}_{2D}$ sparse. With this adaption the problem with the numerical scheme of section 4.2 is solved in about 5 and a halve seconds.