Numerical solution of two-dimensional fingering patterns

Author:
Anika Remorie

Supervisors:
Dr. P.A. Zegeling
Dr. H.M. Schuttelaars

MSc. Thesis
Scientific Computing
Utrecht University

November 26, 2010

Abstract

In this report we numerically simulate fingering patters in unsaturated porous media using a combination of implicit and explicit finite difference methods. These fingering patterns can be modeled using the nonequilibrium Richards Equation (NERE). The NERE is an extended version of the Richards Equation (RE), the conventional equation for modeling flow in porous media. Theoretical research in the past has shown that a propagating water front that is uniform in the lateral direction is conditionally unstable to finite perturbations to the flow field. This instability causes the fingering patterns. Our numerical experiments reproduce the theoretical results: small perturbations start to grow if the wave number of the perturbations is small enough and the parameter τ is large enough, where τ is a measure for the nonequilibrium contribution in the NERE. Initially our numerical experiments were carried out on a fixed grid. To allow for a more accurate numerical solution of the fingering patterns we solve the NERE on an adaptive grid as well.

Preface

I wrote this Master Thesis as the final part of the Master's programme Scientific Computing at the Utrecht University. During the Bachelor's programme Applied Earth Sciences in Delft I became interested in fluid flow and during my studies in Utrecht I mainly focused on the numerical solution of partial differential equations. When the supervisor of this project, Paul Zegeling, introduced the subject of fingering patterns to me I was enthusiastic immediately, because this subject covers two of my main interests.

When I started my research, the first goal was to simulate a finger on a simple stationary grid. If we would obtain good results, the ultimate goal would be to simulate the fingering patterns on an adaptive grid. My first attempt was to model a finger using a strongly simplified model that was completely linear. Eventually we had to conclude that this simplified model was not good enough for generating fingers. Therefore the next attempt was to use a more complicated model, which appeared to be able to generate fingers indeed. I compared the stability properties of the numerical solution with the results of a theoretical stability analysis and the results compared surprisingly well. Because of these promising results we decided that during the final stage of my research I would try to simulate the fingering phenomenon on an adaptive grid.

I would like to thank Henk Schuttelaars (TU Delft) for his clear explanations of the stability properties of the model, both theoretically and numerically. Furthermore I would like to thank Paul Zegeling, who intensively guided me through the entire research during our weekly meetings. And last but not least I want to thank the people at home for their great support and interest in my progress.

Contents

List of Figures

Chapter 1

Introduction

Five decades ago fingering patterns in unsaturated porous media were observed in experiments [4]. These patterns are gravity-driven instabilities in the flow of a fluid through the porous medium. We can imagine the fingering patterns as follows: suppose that we have a porous medium in which a fluid is moving downwards under the influence of gravity. Below the moving front the medium is dry. Under the correct circumstances it is possible that the moving front of the fluid is not a straight line but instead looks like fingers of which the tip has a larger saturation than the rest of the finger.

In practice, the fingering patterns can cause fluids and contaminants to flow faster through the vadose zone to the ground water compared to flow without fingering patterns; this can happen in many kinds of field conditions [1]. Therefore it is important to have knowledge about when fingers can arise and how they behave.

Three decades ago scientists started to become interested in the finger phenomenon. Since then much research has taken place, experimentally in the laboratory and in the field as well as mathemetically and numerically. However, it is still not entirely clear what process causes the flow to become unstable so that fingers can arise. In [4] it is argued that an unstable flow may be caused by processes at the pore scale, processes that are not included in the Richards Equation (RE), the equation that is used conventionally for modeling flow in unsaturated porous media.

Suppose that we have an unsaturated porous medium with small irregularities (perturbations) at the wetting front. By means of stability analysis it can be shown that the flow field described by the RE is unconditionally stable to these small perturbations and hence cannot produce an unstable flow. Since fingering patterns are caused by unstable flows, the RE should be extended in order to model the fingering phenomenon. In [1] two extensions are investigated: the sharp front Richards Equation and the nonequilibrium Richards Equation (NERE). In this report we will consider the NERE.

In [1] and [4] it is shown that small perturbations of the initial saturation profile can grow when modeling unsaturated flow in porous media using the NERE. In other words, it is shown that the flow field described by the NERE is able to produce fingers. This is done using a linear stability analysis. In [4] the growth factor of a small perturbation to the initial saturation profile is determined and it appears that it depends on the parameters and the form of the perturbation whether fingers can be formed or not. The determination of this

growth factor is purely based on the linear stability analysis.

In this report we solve the NERE numerically in order to inspect the behavior of an initial perturbation to the flow field. Our intention is to see whether we also observe the theoretical findings for the growth factor numerically. To this end a finite difference method is used; we solve the NERE with a combination of implicit and explicit methods.

In Chapter 2 the RE and the NERE will be described; furthermore a summary of the stability analyses in [1] and [4] will be given. Chapter 3 focuses on the numerical approximation of the NERE using an Implicit-Explicit method. In Chapter 4 it will be explained how we can determine the growth factor of the perturbation. We will use the numerical approximation derived in Chapter 3 for inspecting the behavior of the wetting front: which parameters result in an unstable flow? The results of the simulations will be compared with the results of the theoretical stability analysis.

It is hard to model the behavior of a finger accurately because the tip of a finger can be very steep. Therefore it may be advantageous to use an adaptive grid for inspecting the finger phenomenon. This will be done in Chapter 5.

We implement the program that is used for the numerical simulations in C. The simulations were carried out on a laptop with an Intel Core2 Duo Processor of 1.0 GHz with 2.0 GB of RAM and a Linux 2.6.27 operating system.

Chapter 2

Modeling unsaturated flow in porous media

The Richards Equation (RE) is the equation that is usually used for modeling unsaturated flow in porous media. However, with the RE it is not possible to model the finger phenomenon. In [1] this is shown by linear stability analysis, by proving that small perturbations to the flow can never grow, but vanish in time. Here we will give a summary of this analysis.

By replacing the equilibrium relation between the water pressure and the saturation by a nonequilibrium relation we obtain the nonequilibrium Richards Equation (NERE). With this equation it is possible to simulate an unstable flow, as is shown in [4]. We will also give a review of their stability analysis. This analysis is linear and shows that small perturbations in the flow can grow under certain circumstances.

2.1 The Richards Equation

The RE can be found by combining Darcy's law and the threedimensional mass balance equation; its dimensionless form is given by

$$\begin{aligned} &\frac{\partial S}{\partial t} = \nabla \cdot (K(S)\nabla p) + \frac{\partial K(S)}{\partial z}, \\ &p = \mathcal{P}(S). \end{aligned} \tag{2.1}$$

In this equation S is the effective saturation, $S = \frac{s-s_r}{1-s_r}$, where s is the water saturation and s_r is the residual water saturation. $K(S)$ is the unsaturated hydraulic conductivity and p is the pressure. For the pressure many different functions can be chosen, however, for the RE an equilibrium pressure function that depends on S is used: $\mathcal{P}(S)$. Furthermore, t is the time and z is the vertical coordinate. We take z positive upwards, opposite to the direction of gravity.

When written in this form the RE depends on two variables: p and S. By introducing the diffusivity function $D(S) = K(S)\mathcal{P}'(S)$ we can reduce the number of variables to one; the RE is then written as follows:

$$\frac{\partial S}{\partial t} = \nabla \cdot (D(S)\nabla S) + \frac{\partial K(S)}{\partial z}. \tag{2.2}$$

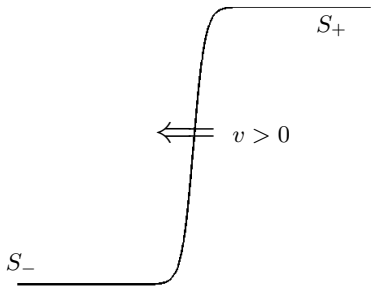


Figure 2.1: Sketch of the traveling wave.

An exact solution for this equation exists; in order to find it we rewrite the equation in its traveling wave form. To that end we define the following traveling wave coordinates:

$$S = S(\xi),$$
$$\xi = z + vt.$$

The boundary conditions for the traveling wave are

$$S(-\infty) = S_-, \qquad S(+\infty) = S_+, \qquad 0 < S_- < S_+ < 1,$$
$$\frac{dS}{d\xi}(-\infty) = \frac{dS}{d\xi}(+\infty) = 0;$$

note that these boundary conditions imply that S_+ is the saturation behind the wetting front, while S_- is the saturation ahead of the wetting front. Figure 2.1 shows a sketch of the traveling wave. The velocity v is defined as follows:

$$v = \frac{K(S_+) - K(S_-)}{S_+ - S_-}. \tag{2.3}$$

The traveling wave form of Equation (2.2) can be written as

$$v\frac{dS}{d\xi} = \frac{d}{d\xi}\left(D(S)\frac{dS}{d\xi}\right) + \frac{dK(S)}{d\xi}.$$

Integrating both sides of the equation once from S to $+\infty$, using the boundary conditions at $+\infty$, gives

$$vS_+ - vS = -D(S)\frac{dS}{d\xi} + K(S_+) - K(S).$$

According to Philip ([5]) the solution to this equation is given by

$$\xi(S) - \xi_* = \int_{S_*}^{S} \frac{D(\sigma)}{v\sigma - vS_+ - K(\sigma) + K(S_+)} d\sigma. \tag{2.4}$$

Here S_* is the saturation at an arbitrary point ξ_*, with $S_- < S_* < S_+$. The inverse of the function $\xi(S)$ is the traveling wave solution of Equation (2.2), which we call $S_0(\xi)$. To explicitely construct this solution, we followed the procedure in [1].

2.2 Stability analysis of the Richards Equation

In order to perform a stability analysis of the RE, we look at the response of the system to a small initial perturbation of the flow field. If this perturbation grows then the flow is unstable. For the analysis a three-dimensional domain is used. We can describe the perturbed saturation field as the sum of the ordinary saturation field, which is the basic solution $S_0(\xi)$, and the perturbation:

$$S = S_0(\xi) + \varepsilon S_1(x, y, \xi, t) + \mathcal{O}(\varepsilon^2).$$

In this equation S_1 is the perturbation and ε is the magnitude of this perturbation, with $0 < \varepsilon \ll 1$. S_1 is of the following form:

$$S_1(x, y, \xi, t) = e^{i\omega_x x + i\omega_y y + kt}\tilde{S}_1(\xi)$$

where $\omega^2 = \omega_x^2 + \omega_y^2$ is the wave number of the perturbation, ω_x and ω_y are the wave numbers in the x- and y-direction respectively. $\tilde{S}_1$ is a function that describes the variation of the perturbation as a function of ξ and vanishes at $\xi = \pm\infty$. The growth factor of the perturbation is represented by k: if k is positive then the perturbation grows, otherwise it declines. The stability analysis therefore comes down to determining the sign of the growth factor.

In order to perform the stability analysis, first we include the perturbed saturation field in Equation (2.2). We use the fact that $\frac{\partial \xi}{\partial z} = 1$ and $\frac{\partial \xi}{\partial t} = v$ and we ignore the terms of $\mathcal{O}(\varepsilon^2)$ and higher. Moreover, ∇ is defined as $\nabla = (\partial_x, \partial_y, \partial_\xi)$.

$$\begin{aligned}
&\frac{\partial\left(S_0(\xi) + \varepsilon e^{i\omega_x x + i\omega_y y + kt}\tilde{S}_1(\xi)\right)}{\partial t} = \\
&\qquad \nabla \cdot \left[D(S_0(\xi) + \varepsilon S_1(x, y, \xi, t))\nabla\left[S_0(\xi) + \varepsilon S_1(x, y, \xi, t)\right]\right] \\
&\qquad + \frac{\partial K(S_0(\xi) + \varepsilon S_1(x, y, \xi, t))}{\partial z}, \\
&\qquad \Longleftrightarrow \\
&v\frac{dS_0}{d\xi} + \varepsilon k S_1 + \varepsilon v\frac{dS_1}{d\xi} = \frac{d}{d\xi}\left(D(S_0)\frac{dS_0}{d\xi}\right) + \varepsilon\nabla \cdot (D(S_0)\nabla S_1) + \\
&\qquad \varepsilon\nabla \cdot \left(S_1 D'(S_0)\nabla S_0\right) + \frac{dK(S_0)}{d\xi} + \varepsilon\frac{dK'(S_0)S_1}{d\xi}.
\end{aligned}$$

Note that $S_0(\xi)$ is independent of x and y and that we expanded the functions $D(S_0(\xi) + \varepsilon S_1(x, y, \xi, t))$ and $K(S_0(\xi) + \varepsilon S_1(x, y, \xi, t))$ in a Taylor series.

We can split the above equation in two equations, such that the first equation only contains terms of order ε^0 and the second equation only contains terms of order ε^1. The first equation then has Equation (2.4) as solution while the second equation describes the behavior of the perturbation.

Rewriting the second equation, using the facts that $D'(S_0)\nabla S_0 = \nabla D(S_0)$ and that

$\nabla \cdot (S_1 \nabla D(S_0)) + \nabla \cdot (D(S_0) \nabla S_1) = \Delta (S_1 D(S_0))$ yields:

$$kS_1 + v\frac{dS_1}{d\xi} = \Delta (S_1 D(S_0)) + \frac{dK'(S_0)S_1}{d\xi},$$
$$\Longleftrightarrow$$
$$kS_1 = \frac{d^2}{d\xi^2}(S_1 D(S_0)) - \omega^2 S_1 D(S_0) + \frac{d}{d\xi}\left(\left(K'(S_0) - v\right) S_1\right).$$

By analyzing the spectrum of this problem we can find the sign of k, or in other words, we can find whether the flow field described by the RE is stable ($k < 0$) or unstable ($k > 0$). In [1] this is done in a very detailed way. The procedure comes down to replacing S_1 by another function and ξ by another coordinate to make the equation less complicated. Subsequently one arrives at the Schrödinger equation, which is a well known problem. Egorov et al. ([1]) analyzed this equation and found that $k < 0$ always whenever the perturbation is small. Hence the flow field described by the RE is unconditionally stable to all small perturbations. By nonlinear stability analysis they even showed that the flow field is unconditionally stable for all finite perturbations in heterogeneous media.

2.3 The nonequilibrium Richards Equation

The NERE is arrived at by replacing the equilibrium pressure $p = \mathcal{P}(S)$ in the RE by a nonequilibrium function. The general form of the NERE is [2]:

$$\frac{\partial S}{\partial t} = \nabla \cdot (K(S)\nabla p) + \frac{\partial K(S)}{\partial z}, \tag{2.5}$$
$$F\left(S, p, \dot{S}, \dot{p}, \ddot{S}, \ddot{p}, \dots\right) = 0, \tag{2.6}$$

where F is the general function for the pressure.

For the numerical simulations we will use the following dynamic nonequilibrium pressure function:

$$\tau\frac{\partial S}{\partial t} = p - \mathcal{P}(S),$$

where τ is a constant. Replacing the equilibrium pressure p in the RE (Equation (2.1)) with this function yields:

$$\begin{aligned}\frac{\partial S}{\partial t} &= \nabla \cdot \left[K(S)\nabla\left(\tau\frac{\partial S}{\partial t} + \mathcal{P}(S)\right)\right] + \frac{\partial K(S)}{\partial z} \\ &= \nabla \cdot [K(S)\nabla\mathcal{P}(S)] + \frac{\partial K(S)}{\partial z} + \tau\nabla \cdot \left[K(S)\nabla\left(\frac{\partial S}{\partial t}\right)\right].\end{aligned}$$

Similarly as for the RE we can introduce the diffusivity function $D(S) = K(S)\mathcal{P}'(S)$ in order to reduce the number of variables to one:

$$\frac{\partial S}{\partial t} = \nabla \cdot (D(S)\nabla S) + \frac{\partial K(S)}{\partial z} + \tau\nabla \cdot \left[K(S)\nabla\left(\frac{\partial S}{\partial t}\right)\right]. \tag{2.7}$$

This is the form in which we will use the NERE in the numerical simulations.

2.4 Stability analysis of the nonequilibrium Richards Equation

The linear stability analysis of the NERE consists of the same steps as for the RE. We follow the procedure as pointed out in [1] and [4]. We start with an analysis of the general NERE (Equation (2.5)) with the pressure function described by Equation (2.6).

We can rewrite the NERE in its traveling wave form. Suppose that the solution to this traveling wave equation is given by the saturation field $S_0(\xi)$ and by the pressure field $p_0(\xi)$. In order to investigate the stability properties of the NERE we add a small perturbation to the saturation field and to the pressure field, comparable to the perturbation added in the stability analysis for the RE. As a consequence the perturbed saturation field (S) and the perturbed pressure field (p) are given by:

$$S = S_0(\xi) + \varepsilon e^{i\omega_x x + i\omega_y y + kt}\tilde{S}_1(\xi) + \mathcal{O}(\varepsilon^2),$$
$$p = p_0(\xi) + \varepsilon e^{i\omega_x x + i\omega_y y + kt}\tilde{p}_1(\xi) + \mathcal{O}(\varepsilon^2).$$

By including these two functions in Equation (2.5) and (2.6) and by dropping the terms of order ε^2 and higher, we obtain the perturbation equations for the eigenvalues k and the eigenfunctions $\tilde{S}_1$ and $\tilde{p}_1$:

$$\frac{dA}{d\xi} + \omega^2 K(S_0)\tilde{p}_1 = -k\tilde{S}_1, \tag{2.8}$$
$$\mathcal{F}\left(\tilde{S}_1, \tilde{p}_1, \frac{d\tilde{S}_1}{d\xi}, \frac{d\tilde{p}_1}{d\xi}, \ldots; S_0, p_0, \frac{dS_0}{d\xi}, \frac{dp_0}{d\xi}, \ldots; k\right) = 0,$$

where

$$A = -K(S_0)\frac{d\tilde{p}_1}{d\xi} - K'(S_0)\left(1 + \frac{dp_0}{d\xi}\tilde{S}_1\right) + v\tilde{S}_1.$$

By integrating Equation (2.8) and by letting the flux A go to zero as ξ goes to infinity we obtain the following equation:

$$\omega^2 \int_{-\infty}^{+\infty} K(S_0)\tilde{p}_1 d\xi = -k \int_{-\infty}^{+\infty} \tilde{S}_1 d\xi.$$

From this equation we should determine the sign of k: only if k can be positive, the NERE is able to produce an unstable flow. In [1] this is done by inspecting an eigenvalue k_0 and its corresponding eigenfunctions at very small wave number of the perturbation, i.e. ω_x and ω_y infinitesimal. It is found that for instability it is required that $C < 0$ with

$$C = \int_{-\infty}^{+\infty} K(S_0)\frac{dp_0}{d\xi} d\xi.$$

A negative value for C occurs when the pressure field is sufficiently nonmonotonic [4]. It is necessary that $\frac{dp_0}{d\xi} < 0$; this occurs when the pressure gradient is opposite to the flow direction.

In [4] a stability analysis for a specific form of the NERE was performed. For the pressure the following relation was used:

$$\tau(s,p)\frac{\partial S}{\partial t} = p - \mathcal{P}(S),$$

where $\tau(s,p) = \tau_0\tau(S_0,p_0)$, with τ_0 a constant. This function resembles the function we are using in our numerical simulations. The only difference is that in [4] τ is a function of the pressure and the saturation, while we take it to be a positive constant.

For doing a linear stability analysis we need to find the perturbation equations. We can do this in an equivalent way as for the RE and the general NERE. After including the specific relation for the pressure into Equation (2.6) and the perturbed saturation and pressure field into Equation (2.5) and (2.6), we obtain the following two equations:

$$\frac{dA}{d\xi} + \omega^2 K(S_0)\tilde{p}_1 = -k\tilde{S}_1,$$
$$v\tau_0\frac{d\tilde{S}_1}{d\xi} + \left(\mathcal{P}'(S_0) + v\frac{\partial\tau(S_0,p_0)}{\partial S}\frac{dS_0}{d\xi}\right)\tilde{S}_1 + \left(v\frac{\partial\tau(S_0,p_0)}{\partial p}\frac{dS_0}{d\xi} - 1\right)\tilde{p}_1 = -k\tau_0\tilde{S}_1;$$

hereby the terms of order ε^2 and higher were ignored. A is the same as for the stability analysis of the general NERE.

Nieber et al. [4] solved this spectral problem numerically for various values of τ_0 between 0 and 1.0; the results are shown in Figure 2.2. We see that for curve 1, corresponding to $\tau_0 = 0$, $k_0 < 0$ for all values of ω and hence the flow is stable. This is indeed what we would expect considering the results of the stability analysis for the RE. For τ_0 very small (curve 2) the growth factor is still negative for all wave numbers, apparently the saturation profile is not sufficiently nonmonotonic yet for enabling an unstable flow. However, further increment of τ_0 (curves 3 to 6) yields a positive growth factor for wave numbers that are not too large. For these values of τ_0 behind the wetting front oscillations appear. These oscillations cause the pressure gradient to be opposite to the flow, which is a requirement for the growth factor to be positive, as shown in the stability analysis. This nonmonotonicity in the saturation profile causes small perturbations to grow such that fingers arise.

In Chapter 3 we will pay more attention to the oscillations. In our numerical experiments we are going to investigate whether we observe a comparable behavior for the conditions under which perturbations can grow.

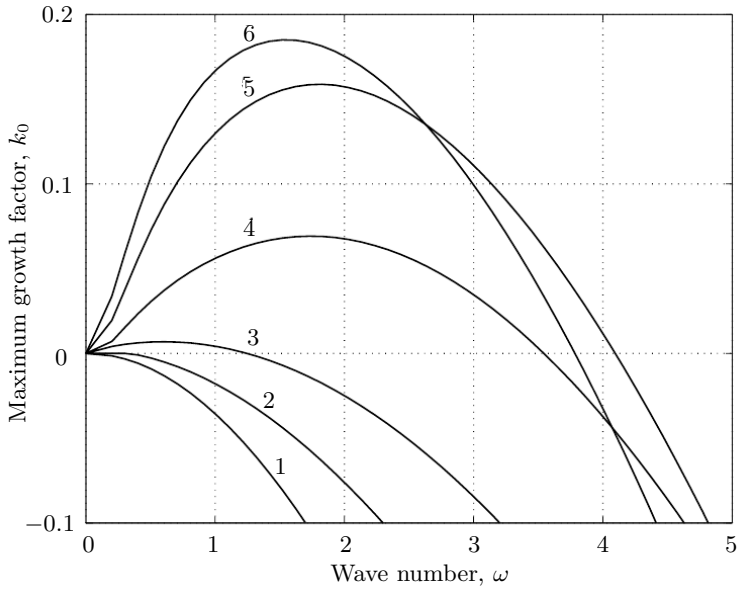


Figure 2.2: The maximum growth factor k_0 as a function of the wave number of the perturbation ω. The curve labels 1 to 6 represent different values of τ_0, increasing from 0 to 1.0. This plot was made by Nieber et al. [4].

Chapter 3

Numerical simulation of the nonequilibrium Richards Equation

For numerically solving the NERE we use the finite difference method. The NERE contains stiff terms and therefore implicit time integration is convenient. However, the equation includes nonlinear terms as well, which makes it rather complicated to use a fully implicit scheme. Therefore we will use an implicit-explicit (IMEX) scheme. We will explain briefly what an IMEX scheme is and apply one possible scheme to the NERE.

Using the finite difference approximation resulting from the IMEX scheme, we will look at the behavior of the wetting front without any perturbations in this chapter. Perturbations will be added in the next chapter.

3.1 Implicit-Explicit scheme

Suppose that we have to solve a partial differential equation with both stiff and non-stiff terms numerically with the finite difference method. Consider for example the following PDE:

$$\frac{\partial u}{\partial t} = f(u) + d\Delta u. \tag{3.1}$$

This PDE corresponds to a reaction-diffusion equation. The diffusion term in the equation is generally a stiff term [3, pp.6]; as a consequence, if we want to solve the PDE with an explicit scheme a very small time step is required in order to obtain a stable numerical solution. In more than one dimension this might lead to very long computation times. Implicit schemes are more suitable for solving PDEs with stiff terms, they do not require such a small time step.

However, the reaction term may be nonlinear. Solving the entire equation implicitely then yields solving a nonlinear system of equations during each time step, which is very time consuming in case that the function $f(u)$ has unfavorable properties [6]. To overcome these difficulties one could consider to use an implicit-explicit scheme.

In fact, an IMEX scheme is nothing more than a scheme that treats one part of the PDE implicitely while the other part is treated explicitely. IMEX schemes are widely used in

computational mathematics and many different schemes are available. In [6] a selection of IMEX methods is described. The Crank-Nicolson method is, for example, combined with the Adams-Bashfort and Leap Frog method.

For the reaction-diffusion equation mentioned above using an IMEX scheme is advantageous because the (nonlinear) reaction term can be calculated very easily using explicit time integration, while the diffusion term can be treated implicitely without having to solve a nonlinear system of equations during every time step. As a consequence a time step of the same order as for a fully implicit scheme can be used, which often yields a smaller computation time than for a fully implicit or explicit scheme [6].

3.1.1 The IMEX-θ method

One of the simplest IMEX methods is the IMEX-θ method: it combines the Euler Forward method with the implicit θ-method [3, pp.383]. We write an arbitrary PDE with both stiff and non-stiff terms as follows:

$$\frac{\partial u}{\partial t} = F(t, u(t)) + G(t, u(t)),$$

where F contains the non-stiff terms of the PDE that are suitable for explicit treatment, while G contains the stiff terms that should be integrated implicitely. The IMEX-θ method then reads as follows:

$$u^{n+1} = u^n + \Delta t F(t^n, u^n) + (1-\theta)\Delta t G(t^n, u^n) + \theta \Delta t G(t^{n+1}, u^{n+1}), \tag{3.2}$$

where Δt is the size of the time step and $u^n \approx u(n\Delta t)$. If we consider Equation (3.1), then $F(t, u(t)) = f(u)$ and $G(t, u(t)) = d\Delta u$.

In order to get an idea of the stability properties of the IMEX-θ method we consider the complex scalar test equation:

$$\frac{\partial u}{\partial t} = \lambda_F u(t) + \lambda_G u(t)$$

where the λ's are the eigenvalues of F and G after linearization. Including this test equation into Equation (3.2) yields

$$\begin{aligned} u^{n+1} &= u^n + \Delta t \lambda_F u^n + (1-\theta)\Delta t \lambda_G u_n + \theta \Delta t \lambda_G u^{n+1} \\ &\iff \\ u^{n+1} &= \frac{1 + \Delta t \lambda_F + (1-\theta)\Delta t \lambda_G}{1 - \theta \Delta t \lambda_G} u^n. \end{aligned}$$

For the IMEX-θ method to be stable it is required that

$$\left| \frac{1 + \Delta t \lambda_F + (1-\theta)\Delta t \lambda_G}{1 - \theta \Delta t \lambda_G} \right| \leq 1.$$

In case that $\theta = 1$ stability is guaranteed if

$$|1 + \Delta t \lambda_F| \leq 1 \quad \wedge \quad |1 - \Delta t \lambda_G| \geq 1,$$

which means that it is sufficient if the explicit part satisfies the stability requirements for the Euler Forward method and the implicit part satisfies the stability requirements for the Euler Backward method. Note that this is not true for $\theta \neq 1$. For an explanation of the stability requirements for $\theta \neq 1$ we refer to [3, pp.383]. This explanation is not relevant here because we will be using the IMEX-θ method with $\theta = 1$.

3.1.2 Application to the nonequilibrium Richards Equation

In order to solve the NERE we apply the IMEX-θ method with $\theta = 1$. Recall that the form of the NERE we are going to solve is given by:

$$\frac{\partial S}{\partial t} = \nabla \cdot (D(S)\nabla S) + \frac{\partial K(S)}{\partial z} + \tau \nabla \cdot \left[K(S) \nabla \left(\frac{\partial S}{\partial t} \right) \right]. \tag{3.3}$$

The second term on the right-hand side of the equation, $(K(S))_z$, is a nonlinear advection term and therefore suitable for explicit time integration. The other two terms, $\nabla \cdot (D(S)\nabla S)$ and $\tau \nabla \cdot (K(S)\nabla S_t)$, are stiff terms and should be integrated implicitely. However, both terms contain functions that depend on S: $D(S)$ and $K(S)$. Hence, applying the IMEX-θ method directly to Equation (3.3) would still force us to solve a nonlinear system of equations during each time step. In order to prevent this we evaluate the functions $D(S)$ and $K(S)$ at time step t^n instead of at time step t^{n+1}. In this way we are able to fully exploit the advantages of the IMEX-θ method: we solve the stiff terms of the PDE semi-implicitely, which gives us the possibility to use a relatively large time step, but without having to solve a nonlinear system of equations during each time step.

We solve the NERE in two dimensions; x is the horizontal coordinate perpendicular to the direction of gravity and z is the vertical coordinate, taken positive in the opposite direction of gravity. Below the discretization for each of the terms in Equation (3.3) will be discussed separately for an arbitrary internal grid point i, j, for the time step from t^n to t^{n+1}. We use a fixed, equally distributed mesh of which Δx is the horizontal spacing and Δz is the vertical spacing.

The left-hand side of Equation (3.3) is approximated as follows:

$$\left. \frac{\partial S}{\partial t} \right|_{i,j}^{n} \approx \frac{S_{i,j}^{n+1} - S_{i,j}^{n}}{\Delta t}.$$

In order to make a discretization for the first term of the right-hand side, we work it out:

$$\begin{aligned} \nabla \cdot (D(S)\nabla S) &= \nabla \cdot \left(D(S) \begin{pmatrix} \frac{\partial S}{\partial x} \\ \frac{\partial S}{\partial z} \end{pmatrix} \right) \\ &= \frac{\partial}{\partial x} \left(D(S) \frac{\partial S}{\partial x} \right) + \frac{\partial}{\partial z} \left(D(S) \frac{\partial S}{\partial z} \right). \end{aligned}$$

Next we use a central spatial discretization and approximate the function $D(S)$ in the desired grid point using the trapezoidal rule:

$$\begin{aligned} \left. \frac{\partial}{\partial x} \left(D(S) \frac{\partial S}{\partial x} \right) \right|_{i,j}^{n} &\approx \frac{D\left(S_{i+\frac{1}{2},j}^{n}\right) \left. \frac{\partial S}{\partial x} \right|_{i+\frac{1}{2},j}^{n+1} - D\left(S_{i-\frac{1}{2},j}^{n}\right) \left. \frac{\partial S}{\partial x} \right|_{i-\frac{1}{2},j}^{n+1}}{\Delta x} \\ &\approx \frac{D\left(S_{i+\frac{1}{2},j}^{n}\right) \left(S_{i+1,j}^{n+1} - S_{i,j}^{n+1}\right) - D\left(S_{i-\frac{1}{2},j}^{n}\right) \left(S_{i,j}^{n+1} - S_{i-1,j}^{n+1}\right)}{(\Delta x)^2} \end{aligned}$$

$$\approx \frac{D(S^n_{i+1,j}) + D(S^n_{i,j})}{2} \frac{S^{n+1}_{i+1,j} - S^{n+1}_{i,j}}{(\Delta x)^2} - \frac{D(S^n_{i,j}) + D(S^n_{i-1,j})}{2} \frac{S^{n+1}_{i,j} - S^{n+1}_{i-1,j}}{(\Delta x)^2}.$$

For the z-direction the same procedure is followed, resulting in a similar expression as for the x-direction.

The second term, the term that will be integrated explicitely, is discretized with an upwind approximation. We do not use a central approximation because that might lead to an unstable numerical scheme. The reason for this is that the advection equation is unconditionally unstable when approximated with a forward time, central space approximation. Hence the discretization for the second term is:

$$\left.\frac{\partial K(S)}{\partial z}\right|^n_{i,j} \approx \frac{K(S^n_{i,j+1}) - K(S^n_{i,j})}{\Delta z}.$$

Finally, the third term on the right-hand side is the most complicated term. Working it out in the same way as the first term gives:

$$\tau \nabla \cdot \left[K(S) \nabla \left(\frac{\partial S}{\partial t} \right) \right] = \tau \frac{\partial}{\partial x} \left(K(S) \frac{\partial^2 S}{\partial x \partial t} \right) + \tau \frac{\partial}{\partial z} \left(K(S) \frac{\partial^2 S}{\partial z \partial t} \right)$$

For the finite difference approximation we can use the same strategy as for the first term, but now there is a time derivative involved as well. This time derivative is handled by calculating the spatial derivative between the brackets at two different time steps: at time step n and at time step $n+1$:

$$\begin{aligned}
\left.\frac{\partial}{\partial x} \left(K(S) \frac{\partial^2 S}{\partial x \partial t} \right)\right|^n_{i,j} &\approx \frac{K\left(S^n_{i+\frac{1}{2},j}\right) \left.\frac{\partial^2 S}{\partial x \partial t}\right|^n_{i+\frac{1}{2},j} - K\left(S^n_{i-\frac{1}{2},j}\right) \left.\frac{\partial^2 S}{\partial x \partial t}\right|^n_{i-\frac{1}{2},j}}{\Delta x} \\
&\approx K(S^n_{i+\frac{1}{2},j}) \left(\frac{S^{n+1}_{i+1,j} - S^{n+1}_{i,j}}{(\Delta x)^2 \Delta t} - \frac{S^n_{i+1,j} - S^n_{i,j}}{(\Delta x)^2 \Delta t} \right) \\
&\quad - K(S^n_{i-\frac{1}{2},j}) \left(\frac{S^{n+1}_{i,j} - S^{n+1}_{i-1,j}}{(\Delta x)^2 \Delta t} - \frac{S^n_{i,j} - S^n_{i-1,j}}{(\Delta x)^2 \Delta t} \right) \\
&\approx \frac{K(S^n_{i+1,j}) + K(S^n_{i,j})}{2} \left(\frac{S^{n+1}_{i+1,j} - S^{n+1}_{i,j}}{(\Delta x)^2 \Delta t} - \frac{S^n_{i+1,j} - S^n_{i,j}}{(\Delta x)^2 \Delta t} \right) \\
&\quad - \frac{K(S^n_{i,j}) + K(S^n_{i-1,j})}{2} \left(\frac{S^{n+1}_{i,j} - S^{n+1}_{i-1,j}}{(\Delta x)^2 \Delta t} - \frac{S^n_{i,j} - S^n_{i-1,j}}{(\Delta x)^2 \Delta t} \right).
\end{aligned}$$

For the z-direction the discretization is similar.

By combining all these discretizations we obtain the IMEX-θ scheme for the NERE. Bringing the terms that should be approximated at time t^{n+1} to the left-hand side of the equation and bringing the other terms to the right-hand side leads to the following system of equations:

$$A(\mathbf{S}^n)\mathbf{S}^{n+1} = \mathbf{b}(\mathbf{S}^n).$$

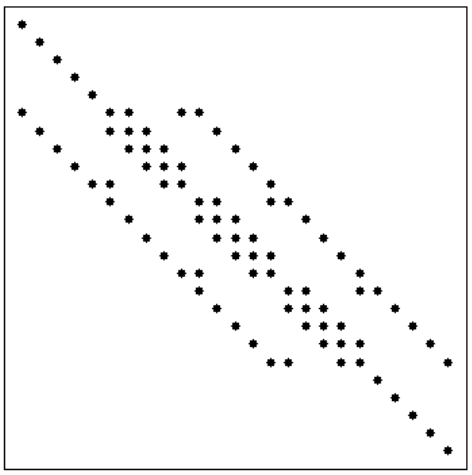


Figure 3.1: The structure of the discretization matrix for the IMEX-θ method applied to the NERE. For this example a grid of size $N_x \times N_z = 5 \times 5$ was used.

In this equation, the vector $\mathbf{S}$ is defined as:

$$\mathbf{S} = (S_{0,0},\ S_{1,0},\ \ldots,\ S_{N_x,0},\ S_{0,1},\ S_{1,1},\ \ldots,\ S_{N_x,N_z})',$$

where N_x is the number of grid points in the x-direction and N_z is the number of grid points in the z-direction. The vector $\mathbf{b}$ contains the terms of the right-hand side of the equation, while the discretization matrix A contains the terms of the left-hand side of the equation. Note that the matrix A depends on $\mathbf{S}$ at time step t^n, this means that we have to calculate A during each time step.

At the top and bottom of the domain, that will be discussed in more detail in the next section, we choose Dirichlet boundary conditions: the saturation is kept fixed at S_+ and S_- respectively. On the left and right boundary of the domain we choose periodic boundary conditions. This implies that the boundary points are approximated with the same accuracy as the inner grid points. Taking into account the boundary conditions, the structure of A is as shown in Figure 3.1.

By solving the system during all time steps we solve the NERE. It is too time consuming to use a direct method because the discretization matrix will be large; therefore the system is solved using an iterative method. In Figure 3.1 we see that A is asymmetric due to its structure. Therefore we cannot use a method like Conjugate Gradients or MINRES (Minimal Residual). Instead we use the Bi-CGSTAB (Bi-Conjugate Gradient Stabilized) method, which is suitable for asymmetric matrices and has a regular convergence pattern[1]. To this end we use the package LASPACK [7], a package written in C that contains various methods for solving linear systems. In order to calculate $\mathbf{S}^{n+1}$ from $\mathbf{S}^n$ we use $\mathbf{S}^n$ as an initial guess for the iteration process.

[1]By a regular convergence pattern we mean that, for the Bi-CGSTAB method, the norm of the residual at a certain iteration step never becomes much larger than the norm at the previous iteration step.

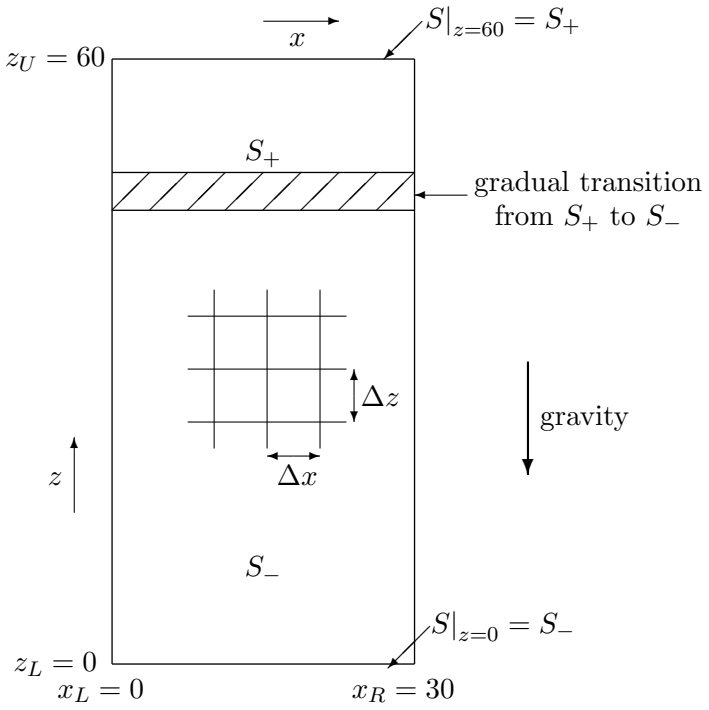


Figure 3.2: The computational domain for solving the NERE.

3.2 Domain, initial condition and parameters

The domain in which we solve the NERE is shown in Figure 3.2. We choose a rectangular domain of size 30×60. As an initial condition, the saturation at the top equals $S_+ = 0.5$ and the saturation at the bottom equals $S_- = 0.05$. In between there is a small transition zone from S_+ to S_-. When integrating the NERE in time, this transition zone may change into fingering patterns. The spacing of the grid is as follows: $\Delta x = 0.25$ and $\Delta z = 0.25$. We use a time step of $\Delta t = 0.05$.

In order to determine a suitable initial condition we start with a saturation profile that looks like a tanh-function from the side of the domain; we do not add any perturbations yet. This function already resembles the profile we want to use for an initial condition, but the steepness of the wetting front is rather random. Therefore we solve the NERE with τ set to zero for a number of time steps using the tanh-function as an initial condition: this is the initialization time. After the initialization time the steepness of the wetting front has adjusted itself to the domain and parameters we chose. If we would perform more time steps, the form of the wetting front would remain the same. The resulting saturation profile is used as the initial condition for our further simulations.

Finally, we have to specify which functions we are going to use for $K(S)$ and $P(S)$. We use the Brooks-Corey relations as mentioned in [4]:

$$\begin{aligned} K(S) &= S^{\alpha}, \qquad \alpha > 2, \\ P(S) &= -S^{-\beta}, \quad 0 < \beta < 1. \end{aligned}$$

From $P(S)$ and $K(S)$ we can calculate $D(S)$:

$$D(S) = K(S)P'(S) = \beta S^{\alpha-\beta-1},$$

and by including $\alpha = 3$ and $\beta = \frac{1}{2}$ we obtain

$$K(S) = S^3, \qquad D(S) = \frac{1}{2}S^{3/2}.$$

The domain and the parameters described so far remain fixed during all simulations described in this report, unless stated otherwise. Only the parameter τ and the perturbation will be varied.

3.3 Behavior of the wetting front

When modeling flow in porous media for an unperturbed wetting front using the NERE, oscillations may appear behind the wetting front if the parameter τ is large enough. These oscillations are necessary for generating fingering patterns: recall from Chapter 2 that fingering patterns can only arise when the pressure gradient opposes the direction of the flow.

In this section we show the behavior of the wetting front for various values of τ and by phase plane analysis we explain our observations. Since we are using an unperturbed initial condition, the saturation profile is independent of x during the simulations in this section. Therefore we perform them in one dimension instead of two. For the phase plane analysis we use a one-dimensional saturation profile as well.

In Figure 3.3 the saturation profile for various values of τ is shown at $t = 150$[2]. For $\tau = 0$ we obtain the RE. We observe that the saturation profile does not contain any irregularities, it varies smoothly from S_- to S_+. However, if we increase τ to a value of 1, we see one small wiggle at the wetting front. Increasing τ even further results in a saturation profile with multiple oscillations behind the wetting front.

This behavior can easily be understood by inspecting the phase plane of the traveling wave solution of the NERE. We write the NERE in its traveling wave form by introducing the traveling wave coordinate $\xi = z + vt$; the traveling wave solution of the saturation $S(z)$ is called $\phi(\xi)$. Transforming Equation (3.3) into its traveling wave form then yields:

$$v\frac{d\phi}{d\xi} = \frac{d}{d\xi}\left(D(\phi)\frac{d\phi}{d\xi}\right) + \frac{dK(\phi)}{d\xi} + v\tau\frac{d}{d\xi}\left(K(\phi)\frac{d^2\phi}{d\xi^2}\right). \tag{3.4}$$

In this equation v is again given by Equation (2.3). Integration of Equation (3.4) gives:

$$\int_{-\infty}^{\xi}\left\{v\frac{d\phi}{d\xi}\right\}d\hat{\xi} = \int_{-\infty}^{\xi}\left\{\frac{d}{d\xi}\left(D(\phi)\frac{d\phi}{d\xi}\right) + \frac{dK(\phi)}{d\xi} + v\tau\frac{d}{d\xi}\left(K(\phi)\frac{d^2\phi}{d\xi^2}\right)\right\}d\hat{\xi}$$

$$\Longleftrightarrow$$

$$v\phi - vS_- = D(\phi)\frac{d\phi}{d\xi} + K(\phi) - K(S_-) + v\tau K(\phi)\frac{d^2\phi}{d\xi^2}.$$

[2]In this report we solve the dimensionless version of the NERE, hence the time variable t is dimensionless.

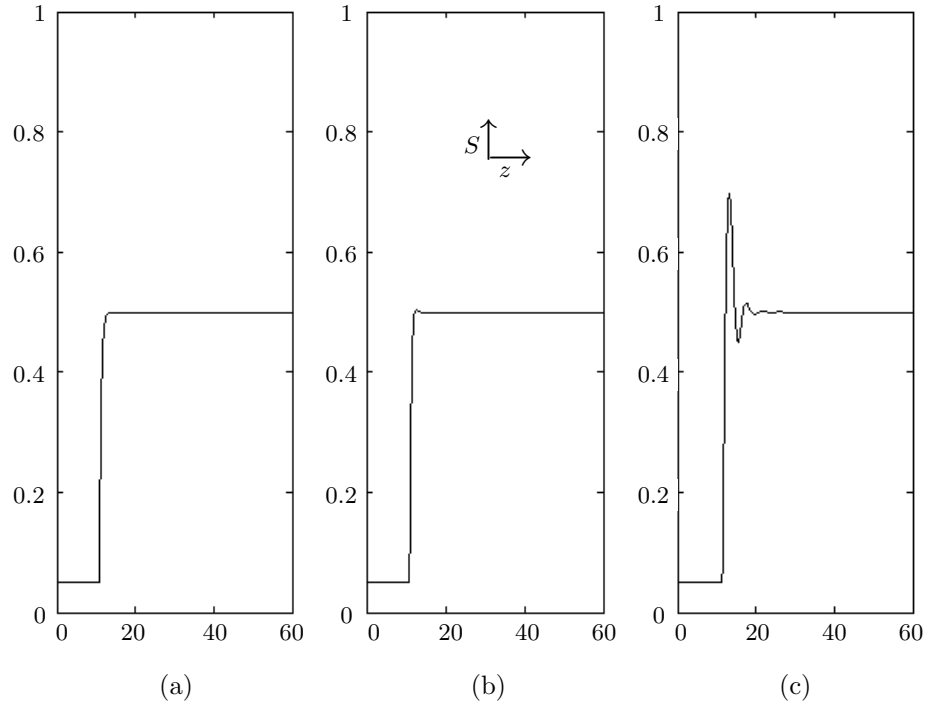


Figure 3.3: The one-dimensional saturation profile for various values of τ; (a) $\tau = 0$; (b) $\tau = 1$; (c) $\tau = 5$.

Hereby we used the following boundary conditions:

$$\phi(-\infty) = S_-,$$
$$\frac{d\phi}{d\xi}(-\infty) = \frac{d^2\phi}{d\xi^2}(-\infty) = 0.$$

We can rewrite this second order ordinary differential equation (ODE) into a system of two first order ODEs as follows:

$$\begin{cases} \dfrac{d\phi}{d\xi} &= \psi \\ \dfrac{d\psi}{d\xi} &= \dfrac{v\phi - vS_- - D(\phi)\psi - K(\phi) + K(S_-)}{v\tau K(\phi)}. \end{cases}$$

Substituting the functions for $K(S)$ and $D(S)$, as mentioned in the previous section, yields:

$$\begin{cases} \dfrac{d\phi}{d\xi} &= \psi \\ \dfrac{d\psi}{d\xi} &= \dfrac{v\phi - vS_- - \beta\phi^{1.5}\psi - \phi^3 + S_-^3}{v\tau\phi^3}. \end{cases} \tag{3.5}$$

This system has two critical points, namely $(\phi, \psi) = (S_-, 0)$ and $(\phi, \psi) = (S_+, 0)$. Before we solve the system, first we inspect the behavior of these critical points. We can determine the

type and stability of the critical points by inspecting the eigenvalues of the Jacobian matrix:

$$\mathcal{J} = \begin{bmatrix} \frac{\partial \phi'}{\partial \phi} & \frac{\partial \phi'}{\partial \psi} \\ \frac{\partial \psi'}{\partial \phi} & \frac{\partial \psi'}{\partial \psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{-2v\phi + 3vS_- + 1.5\beta\psi\phi^{1.5} - 3S_-^3}{v\tau\phi^4} & -\frac{\beta}{v\tau\phi^{1.5}} \end{bmatrix}.$$

At the critical points the Jacobian matrix equals

$$\mathcal{J} = \begin{bmatrix} 0 & 1 \\ \frac{-2vS_\pm + 3vS_- - 3S_-^3}{v\tau S_\pm^4} & -\frac{\beta}{v\tau S_\pm^{1.5}} \end{bmatrix}.$$

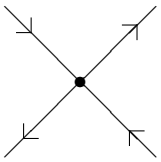
We inspect both critical points separately.

Critical point $(\phi, \psi) = (S_-, 0)$

For this point the eigenvalues of the Jacobian matrix equal

$$\lambda = -\frac{\beta}{2v\tau S_-^{1.5}} \pm \sqrt{\frac{\beta^2}{4v^2\tau^2 S_-^3} + \frac{v - 3S_-^2}{v\tau S_-^3}}.$$

Both terms in the square root are positive and hence the eigenvalues are always real. Moreover the eigenvalues have opposite signs. Therefore the critical point $(\phi, \psi) = (S_-, 0)$ is a saddle point regardless the value of τ. In the picture below a sketch of the point is shown where the dot in the middle represents the coordinate $(\phi, \psi) = (S_-, 0)$.



Critical point $(\phi, \psi) = (S_+, 0)$

For this critical point the eigenvalues of the Jacobian matrix are equal to:

$$\lambda = -\frac{\beta}{2v\tau S_+^{1.5}} \pm \sqrt{\frac{\beta^2}{4v^2\tau^2 S_+^3} + \frac{3vS_- - 2vS_+ - 3S_-^3}{v\tau S_+^4}}. \tag{3.6}$$

Figure 3.4 shows the term in the square root as a function of τ. Note that it depends on the value of τ whether this term is negative or positive, or in other words, whether the eigenvalues have a nonzero imaginary part or not. By including the values for S_+, S_- and β as mentioned in the previous section it is evident that

1. the term in the square root is positive for

   $$\tau < \tau^* \approx 0.476667.$$

   In this case both eigenvalues are real and negative, which means that $(\phi, \psi) = (S_+, 0)$ is a stable node;

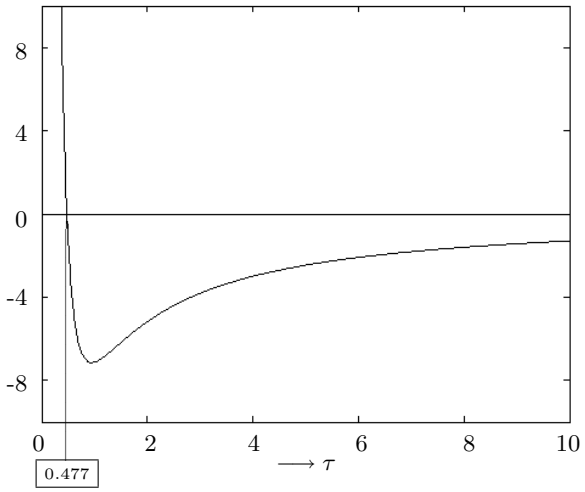


Figure 3.4: The term in the square root of Equation (3.6) as a function of τ for the eigenvalues of the critical point $(\phi, \psi) = (S_+, 0)$.

2. the term in the square root is negative for

   $$\tau > \tau^*.$$

   In this case the eigenvalues are complex conjugate with negative real part. As a consequence $(\phi, \psi) = (S_+, 0)$ is a stable focus.

We solve System (3.5) with Matlab, using the built-in function ode45. Figure 3.5 shows the behavior of orbits starting near $(\phi, \psi) = (S_-, 0)$ for various values of τ. Note that on the horizontal axis the traveling wave solution of the NERE is shown, while the vertical axis shows the derivative with respect to ξ of the traveling wave solution.

In case that $\tau < \tau^*$ the orbits directly go to $(\phi, \psi) = (S_+, 0)$ and hence the resulting saturation profile is monotonic as in Figure 3.3(a). However, in case $\tau > \tau^*$ the orbits spiral around $(\phi, \psi) = (S_+, 0)$ which results in the nonmonotonic saturation profiles shown in Figures 3.3(b) and (c). The larger the magnitude of the spiraling movement in the phase plane, the larger the magnitude of the oscillations in the saturation profile is.

From this analysis we can conclude that only for $\tau > \tau^*$ we obtain a saturation profile with oscillations behind the wetting front. Hence for values of τ smaller than τ^* there is no non-monotonicity in the saturation profile. This means that the pressure gradient does not oppose the flow direction anywhere in the domain and as a consequence no fingering patterns can be generated for $\tau < \tau^*$.

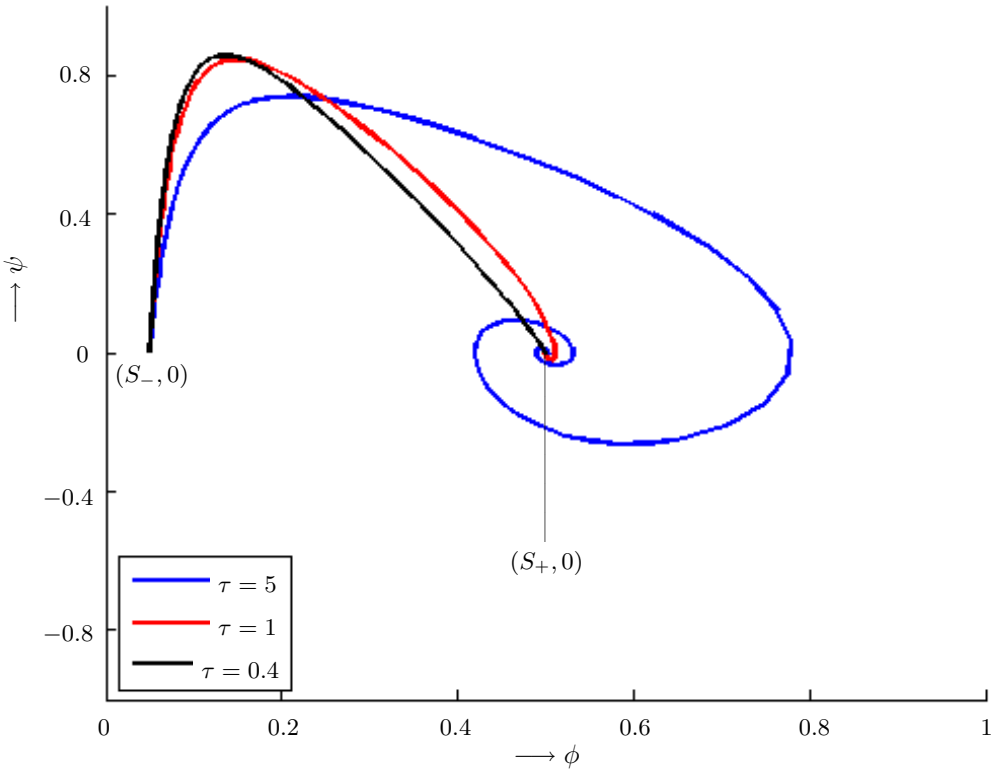


Figure 3.5: The behavior of the orbits in the phase plane for various values of the parameter τ. When $\tau < \tau^*$ the critical point $(S_+, 0)$ is clearly a stable node, but when $\tau > \tau^*$ it is a stable focus.

Chapter 4

The growth factor of the perturbation

In Chapter 2 it was argued that the NERE is conditionally unstable to small perturbations and in Chapter 3 a numerical scheme for solving the NERE was discussed. Using this numerical scheme we are now going to examine whether we indeed observe the unstable behavior that is expected from the theoretical stability analysis.

To this end we first describe how we add a perturbation to the flow field and how we determine whether this perturbation grows or decays, and at which rate. Then we solve the NERE for various values of τ and of the wave number of the perturbation. Finally we will compare the result of our numerical simulations with the results of the theoretical stability analysis.

4.1 Addition of a perturbation to the flow field

After the initialization time the saturation profile looks like the profile shown in Figure 4.1. Since we did not add any perturbation yet, the saturation profile is constant in the x-direction. We perturb the flow field by slightly changing the initial condition.

The perturbation to the flow field is constructed using a perturbation in the z-direction and a perturbation in the x-direction. The perturbation in the z-direction is chosen such that only the wetting front is perturbed. It is zero in the regions where the saturation is (approximately) equal to S_+ or S_- and in the transition zone it has the form of a half period of a cosine, see Figure 4.2 for a sketch. The amplitude of the cosine equals ε.

In the x-direction the perturbation has the form of a cosine as well. The cosine is added along the entire width of the domain and has amplitude one. We interpret the wave number ω as the number of periods of the cosine that fits inside the domain. In the simulations only integer values of ω are used since this is most convenient regarding the periodic boundary conditions.

The final perturbation equals the product of the perturbations in the x- and z-direction. Figure 4.3(a) shows the perturbed initial condition with $\omega = 3$; Figure 4.3(b) enlarges the region where the perturbation was added. For these figures the amplitude of the perturbation

is very large. In our simulations the amplitude is smaller, namely $\varepsilon = 0.025$, which is so small that we can hardly see it in the saturation profile.

4.2 Determination of the maximum derivation

By solving the NERE with the perturbed initial condition we can determine whether the size of the perturbation increases or decreases in time. To that end we solve the NERE from $t = 0$ to $t_{end} = 150$, first with the unperturbed initial condition and then with the perturbed initial condition[1]. We compare the results of both simulations with each other. There are two possibilities:

1. the perturbation decays, this means that the perturbed saturation profile resembles the unperturbed profile more and more as time proceeds;
2. the perturbation grows, the difference between the perturbed and unperturbed saturation profile increases with time.

In order to quantify the increment or decrease of the magnitude of the perturbation we subtract the unperturbed saturation profile from the perturbed profile at 50 equally distributed moments during the simulation. This is done for each grid point, see Figure 4.4 for an illustration. At each moment of comparison the maximum difference between both saturation profiles is determined. In case of a growing perturbation we clearly see an exponential increase of the maximum difference (Figure 4.5(a)) and in case of a decreasing perturbation we see an exponential decrease (Figure 4.5(b)). By fitting an exponential function through the data points we can determine the growth factor. In the next section a fitting procedure will be described.

In both plots of Figure 4.5 we see that the maximum difference of the perturbation first decreases rapidly: the exponential increase or decrease only starts from the fourth or fifth dot. This phenomenon is caused by the fact that initially the perturbation is rather arbitrary, it is not an eigenfunction. Besides, the difference between the saturation profiles cannot keep on growing permanently because the saturation cannot grow unlimited. Therefore we should be careful only to include the dots that represent the exponential growth or decline. The exponential part can be distinguished quite accurately by plotting the natural logarithm of the maximum difference: only the points that lie on a straight line belong to the exponential part.

4.3 Exponential least squares fitting

In order to quantify the growth factor we should make an exponential fit through the data points that were collected during the simulation. This can be done by making an exponential least squares fit [9, 10]. Suppose that the exponential function that fits through the data points is of the following form:

$$y = Ae^{Bt},$$

[1] We choose t_{end} to be equal to 150 because at that time, the wetting front has moved along a large part of the domain, but is not influenced by the Dirichlet boundary condition at the bottom of the domain yet.

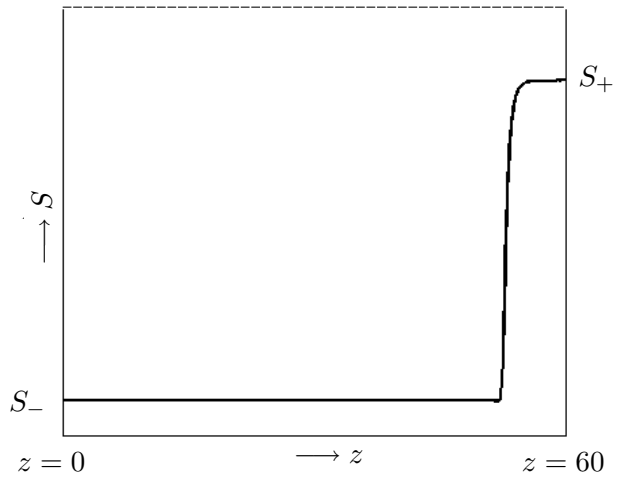


Figure 4.1: Sideview of the unperturbed initial condition.

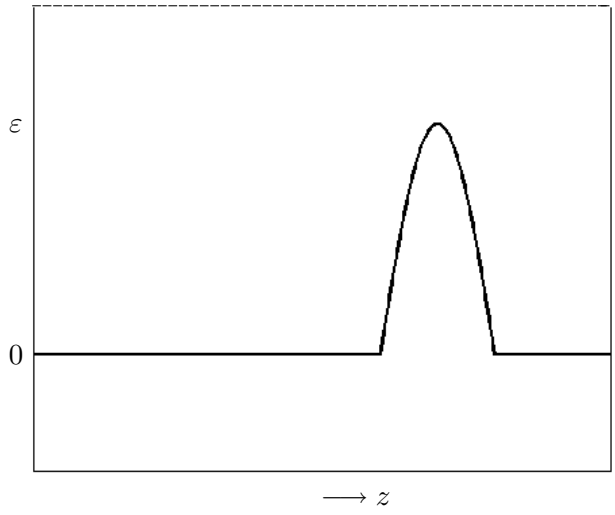
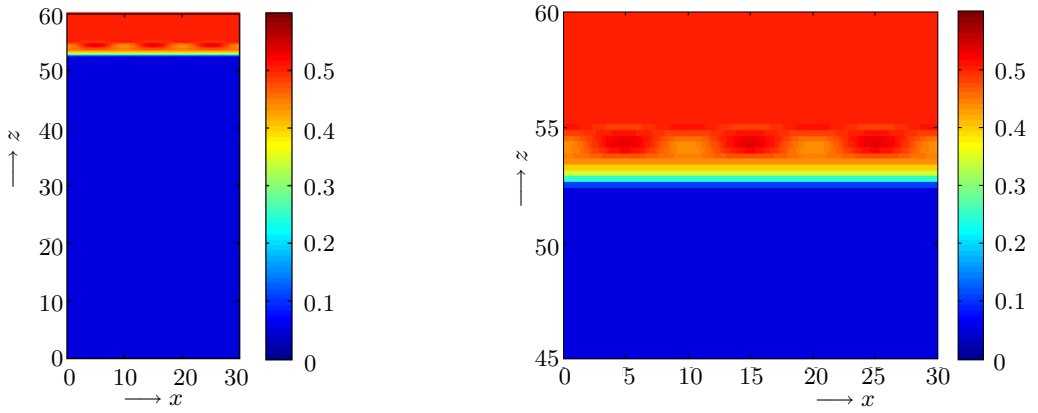
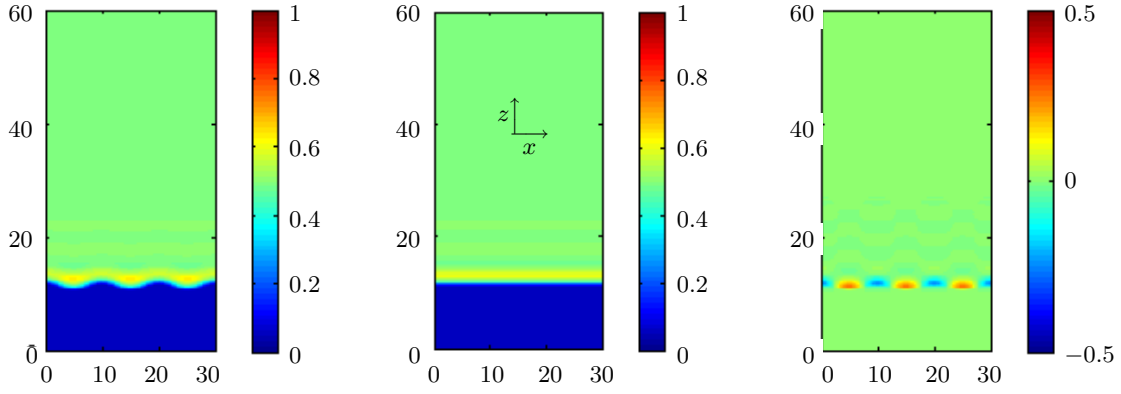


Figure 4.2: Sketch of the magnitude of the perturbation in the z-direction.



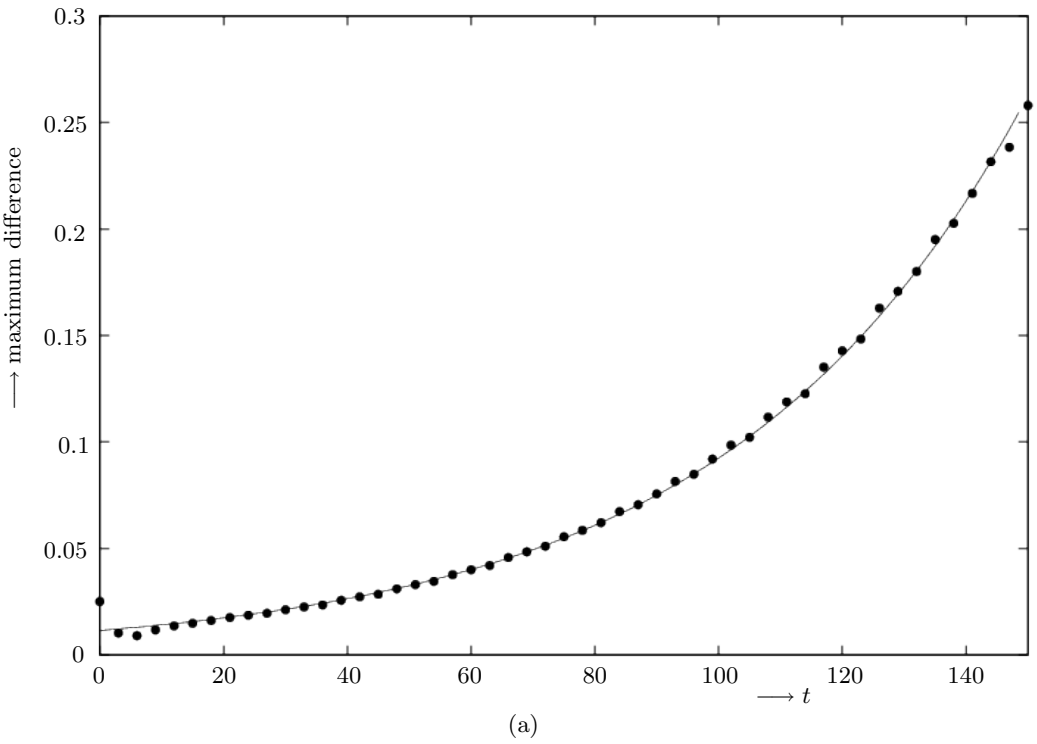
(a) (b)

Figure 4.3: The perturbed initial saturation profile; (a) the entire domain; (b) zoom of the perturbation.

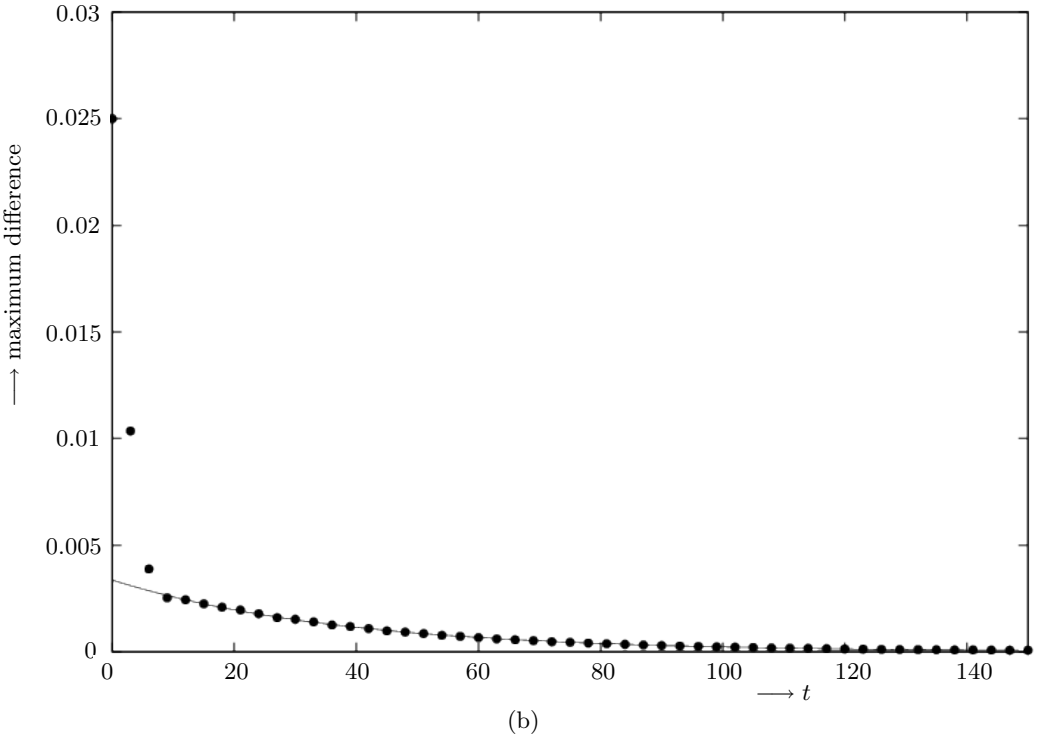


(a) (b) (c)

Figure 4.4: (a) The perturbed saturation profile at $t = 150$; (b) the unperturbed saturation profile at $t = 150$; (c) the difference between the two profiles. Note that the color scale for the right figure differs from the other two figures. The parameters that were used are $\tau = 5$, $\omega = 3$.



(a)



(b)

Figure 4.5: Example of the maximum difference of a perturbed and unperturbed saturation profile as a function of time; (a) a growing perturbation ($\tau = 5$ and $\omega = 3$); (b) a declining perturbation ($\tau = 5$ and $\omega = 9$). The lines through the dots are exponential fits.

where y represents the maximum difference between the saturation profiles, A respresents the y-intercept, B represents the growth factor k and t represents the time. This equation can be rewritten as follows:

$$\begin{aligned} &\ln(y) = \ln(A) + Bt, \\ &\ln(y) = a + bt, \qquad\qquad a = \ln(A), \quad b = B. \end{aligned}$$

If we have n data points that are given by (t_i, y_i), then the squared error equals:

$$R^2 = \sum_{i=1}^{n} \left[\ln(y_i) - (a + bt_i)\right]^2 . \tag{4.1}$$

R^2 is minimized for $\dfrac{\partial R^2}{\partial a} = 0$ and $\dfrac{\partial R^2}{\partial b} = 0$:

$$\begin{aligned} \frac{\partial R^2}{\partial a} &= -2 \sum_{i=1}^{n} \left[\ln(y_i) - (a + bt_i)\right] = 0, \\ \frac{\partial R^2}{\partial b} &= -2 \sum_{i=1}^{n} \left[t_i \left[\ln(y_i) - (a + bt_i)\right]\right] = 0. \end{aligned}$$

Rewriting these two equations gives:

$$\begin{aligned} na + b \sum_{i=1}^{n} \left[t_i\right] &= \sum_{i=1}^{n} \left[\ln(y_i)\right], \\ a \sum_{i=1}^{n} \left[t_i\right] + b \sum_{i=1}^{n} \left[t_i^2\right] &= \sum_{i=1}^{n} \left[t_i \ln(y_i)\right]; \end{aligned}$$

we can write this result in matrix-vector form as follows:

$$\begin{bmatrix} n & \sum\limits_{i=1}^{n} \left[t_i\right] \\ \sum\limits_{i=1}^{n} \left[t_i\right] & \sum\limits_{i=1}^{n} \left[t_i^2\right] \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum\limits_{i=1}^{n} \left[\ln(y_i)\right] \\ \sum\limits_{i=1}^{n} \left[t_i \ln(y_i)\right] \end{bmatrix}.$$

By solving this small system of equations we find the values for a and b that minimize the squared error R^2 and hence give the exponential least squares fit through the data points. Solving the system yields:

$$\begin{aligned} a &= \frac{\sum\limits_{i=1}^{n} \left[\ln(y_i)\right] \sum\limits_{i=1}^{n} \left[t_i^2\right] - \sum\limits_{i=1}^{n} \left[t_i\right] \sum\limits_{i=1}^{n} \left[t_i \ln(y_i)\right]}{n \sum\limits_{i=1}^{n} \left[t_i^2\right] - \left[\sum\limits_{i=1}^{n} \left[t_i\right]\right]^2}, \\ b &= \frac{n \sum\limits_{i=1}^{n} \left[t_i \ln(y_i)\right] - \sum\limits_{i=1}^{n} \left[t_i\right] \sum\limits_{i=1}^{n} \left[\ln(y_i)\right]}{n \sum\limits_{i=1}^{n} \left[t_i^2\right] - \left[\sum\limits_{i=1}^{n} \left[t_i\right]\right]^2}. \end{aligned}$$

Note that in this way the data points with small y-values contribute most to the least squares fit [9]. In order to give each data point an equal weight it is better to perform a weighted least squares fit: we do not minimize the error given in Equation (4.1), but instead we minimize

$$R^2 = \sum_{i=1}^{n} \left[y_i \left[\ln(y_i) - (a + bt_i) \right]^2 \right].$$

To this end we follow the same procedure as above. First we take the derivative of R^2 with respect to a and b:

$$\frac{\partial R^2}{\partial a} = -2 \sum_{i=1}^{n} \left[y_i \left[\ln(y_i) - (a + bt_i) \right] \right] = 0,$$
$$\frac{\partial R^2}{\partial b} = -2 \sum_{i=1}^{n} \left[t_i y_i \left[\ln(y_i) - (a + bt_i) \right] \right] = 0,$$

then we rewrite the equations and obtain a linear system:

$$a \sum_{i=1}^{n} \left[y_i \right] + b \sum_{i=1}^{n} \left[t_i y_i \right] = \sum_{i=1}^{n} \left[y_i \ln(y_i) \right],$$
$$a \sum_{i=1}^{n} \left[t_i y_i \right] + b \sum_{i=1}^{n} \left[t_i^2 y_i \right] = \sum_{i=1}^{n} \left[t_i y_i \ln(y_i) \right],$$
$$\Longleftrightarrow$$
$$\begin{bmatrix} \sum_{i=1}^{n} \left[y_i \right] & \sum_{i=1}^{n} \left[t_i y_i \right] \\ \sum_{i=1}^{n} \left[t_i y_i \right] & \sum_{i=1}^{n} \left[t_i^2 y_i \right] \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} \left[y_i \ln(y_i) \right] \\ \sum_{i=1}^{n} \left[t_i y_i \ln(y_i) \right] \end{bmatrix}.$$

By solving this system the weighted exponential least squares fit is obtained:

$$a = \frac{\sum_{i=1}^{n} \left[t_i^2 y_i \right] \sum_{i=1}^{n} \left[y_i \ln(y_i) \right] - \sum_{i=1}^{n} \left[t_i y_i \ln(y_i) \right] \sum_{i=1}^{n} \left[t_i y_i \right]}{\sum_{i=1}^{n} \left[t_i^2 y_i \right] \sum_{i=1}^{n} \left[y_i \right] - \left[\sum_{i=1}^{n} \left[t_i y_i \right] \right]^2},$$
$$b = \frac{\sum_{i=1}^{n} \left[y_i \right] \sum_{i=1}^{n} \left[t_i y_i \ln(y_i) \right] - \sum_{i=1}^{n} \left[t_i y_i \right] \sum_{i=1}^{n} \left[y_i \ln(y_i) \right]}{\sum_{i=1}^{n} \left[y_i \right] \sum_{i=1}^{n} \left[t_i^2 y_i \right] - \left[\sum_{i=1}^{n} \left[t_i y_i \right] \right]^2}.$$

The weighted fit generally gives better results than the non-weighted fit [9] and therefore we use it for fitting the exponential function through the data points. In Figure 4.5 we see that the exponential functions indeed lie very close to the data points. Finally the growth factor k equals the value of b resulting from this fitting procedure.

4.4 Results

In this section we inspect whether we observe the same behavior as in Figure 2.2 by numerically solving the NERE with a perturbed saturation profile. We run simulations for various values of τ and ω. The results of these simulations are shown in Figure 4.6.

Figure 4.6 clearly shows the same behavior as the theoretically derived plot in Figure 2.2. For small values of τ the growth factor k is negative for all frequencies of the perturbation. Increasing τ gives positive values for k if ω is not too large, and the maximum value for k increases as τ increases.

Even though the curves have the same form in both figures, the values for k, τ and ω do not match. This is because we used a finite domain in our simulations and we did not use the same functions for $K(S)$ and $D(S)$.

The parameters and functions we chose influence all values in the graph of Figure 4.6. For example, a smaller domain would give us less frequencies at which the growth factor is positive and a smaller value for β would decrease the amount of diffusion in the model, which would result in a larger growth factor. An example of a simulation with a smaller value for β is shown in Figure 4.7: the fingering patterns for $\beta = 0.1$ are much more "spectacular" than for $\beta = 0.5$ and the perturbation clearly grows at a faster rate.

However, it was not our aim to match the results of the theoretical stability analysis exactly in the numerical simulations, nor was it our aim to generate spectacular fingering patterns or patterns that resemble the ones observed in laboratory experiments. With the simulations in this section we just showed that, for a rather arbitrary choice of functions and parameters, the NERE clearly satisfies the stability properties derived theoretically.

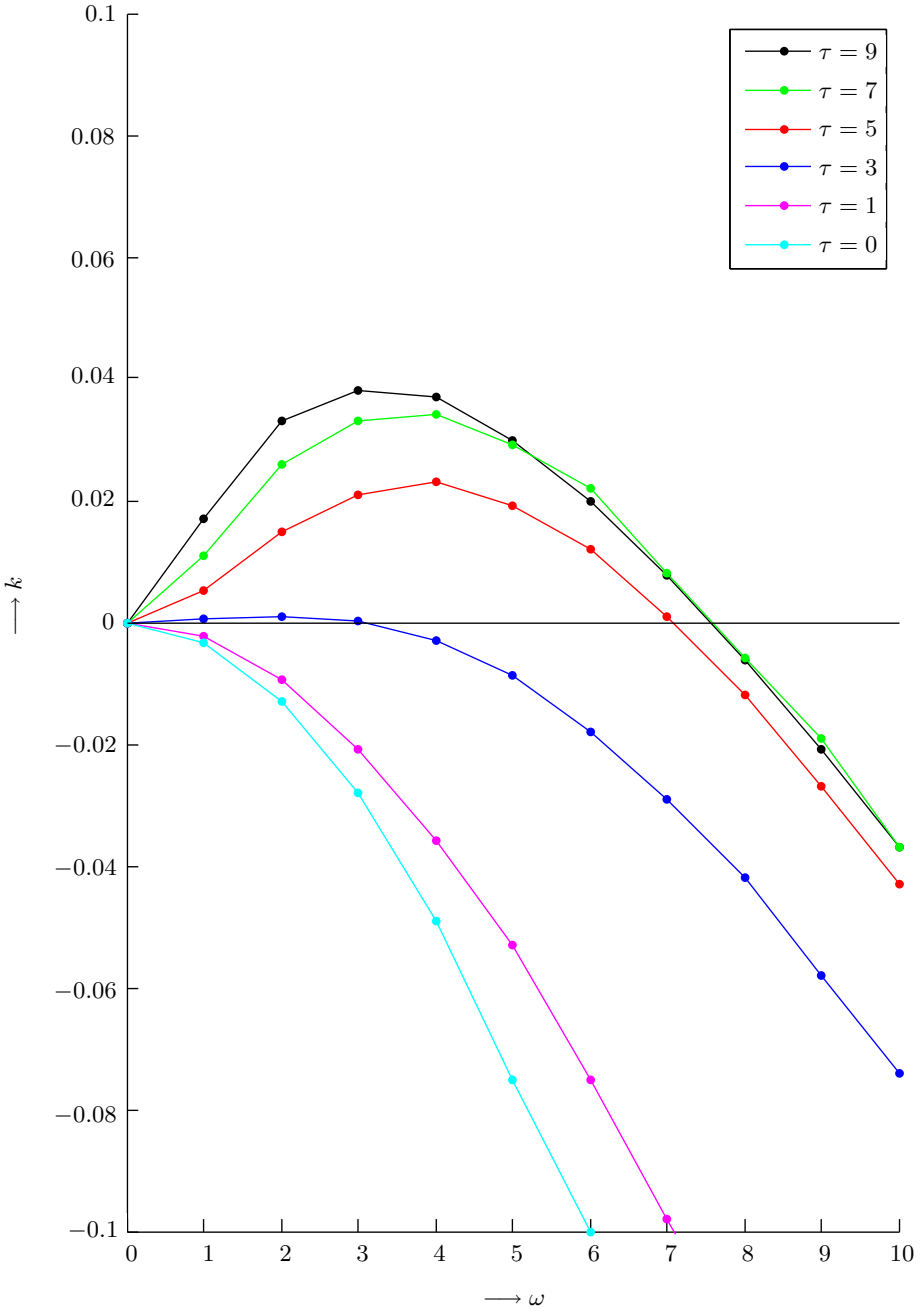


Figure 4.6: The growth factor, numerically determined, as a function of the wave number of the perturbation for various values of τ.

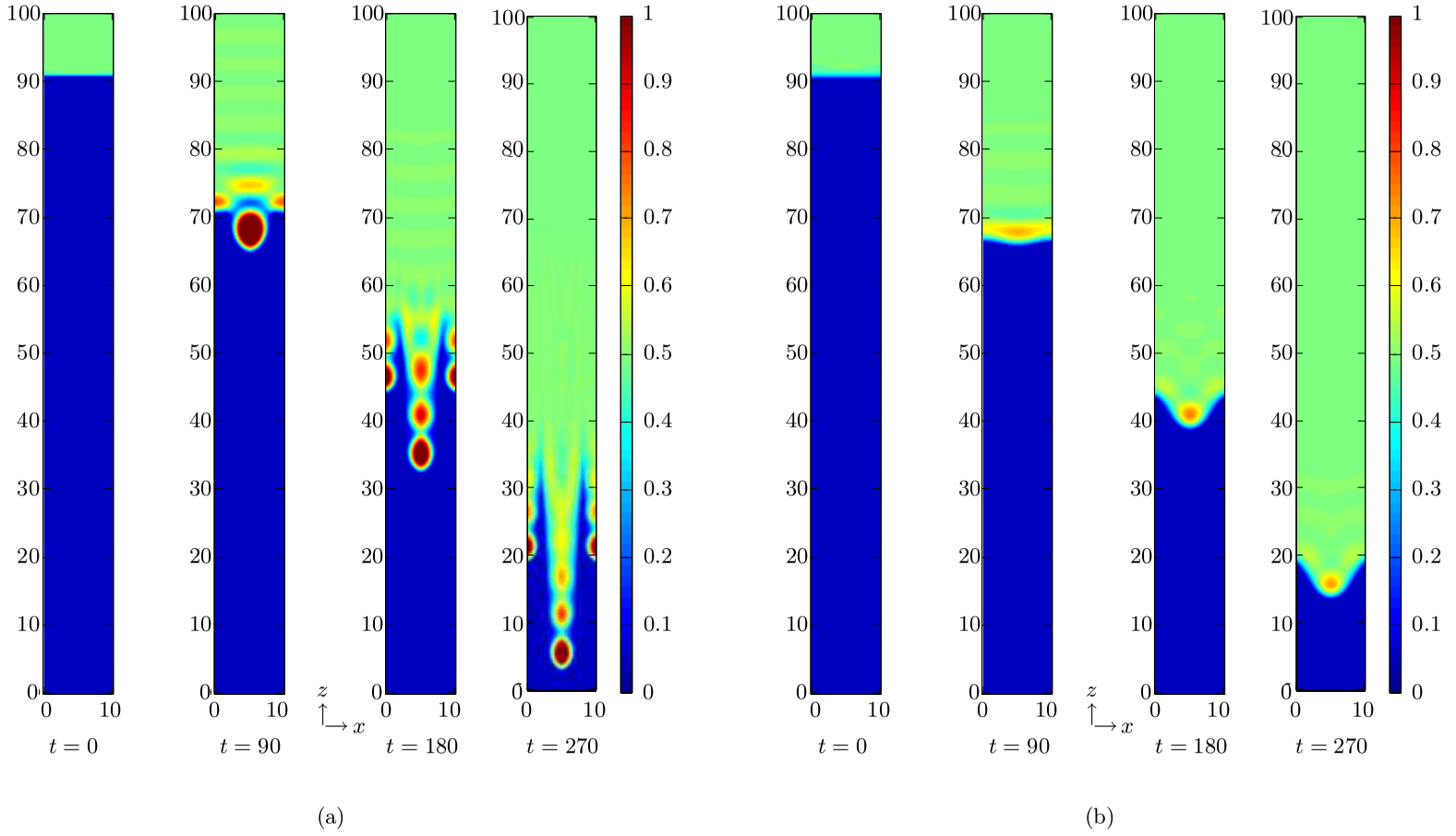


Figure 4.7: The saturation profile for two different values of β; (a) $\beta = 0.1$; (b) $\beta = 0.5$. For a smaller β the fingering patterns look much more spectacular because there is less diffusion. For obtaining this figure we used a domain of size 10×100 and the parameters were as follows: $\tau = 7$, $\omega = 1$ and $\alpha = 3$. The simulation was carried out until $t_{end} = 270$.

Chapter 5

Adaptive scheme

In the experiments that were done up to now a uniformly distributed fixed mesh was used. In order to model the fingering structures accurately, the spacing between the grid points had to be very small because the tips of the fingers can become very steep. However, in regions with less spatial variation of the saturation, a grid with larger spacing would suffice.

Therefore it is convenient to use an adaptive grid for modeling the fingering phenomenon, so that most grid points are concentrated in regions with high spatial variation of the saturation. In this way we could either save computation time by using less grid points, or we could model the fingers even more accurately by concentrating more grid points around the finger tips than with a fixed mesh.

In this chapter we solve the NERE on an adaptive grid. We start with explaining which steps should be taken for solving the NERE on a moving mesh and then we will focus on each step in more detail. The last section of this chapter shows the results of solving the NERE on an adaptive grid.

5.1 Solving the nonequilibrium Richards Equation on an adaptive grid

In the previous chapters, where we integrated the saturation profile on an equally spaced grid, during each time step we calculated the discretization matrix and right-hand side vector and solved the resulting linear system. In order to integrate the saturation profile on an adaptive grid we have to do some extra work.

During each time step the mesh should be adapted to the saturation profile. This adaptation is based on the spatial variation of the saturation profile, regions with a high gradient obtain more grid points, while regions with a low gradient obtain less grid points. The division of grid points over the domain is determined using a monitor function, which will be described in more detail later on. When, in a certain time step, the mesh has been adapted to the saturation profile, we calculate the saturation profile at time $t+\Delta t$ using the IMEX-θ method. Summarizing, performing one time step for time integration on an adaptive grid comes down to performing the following steps:

- evaluate the monitor function;

- move the mesh;
- perform time integration using the IMEX-θ method.

5.1.1 Discretization of the nonequilibrium Richards Equation

The discretization of the saturation profile for the time integration is not the same as described in Chapter 3. In Chapter 3 we had to do with rectangular mesh cells because Δx and Δz were constant over the entire mesh, while now the cells can have any form. The question is: how can we discretize the NERE for non-rectangular mesh cells? We do this by applying a coordinate transformation. We transform the nonuniformly distributed (x, z)-grid with $N_x \times N_z$ grid points to the uniformly distributed (ξ, η)-grid with $(\xi, \eta) \in [0, 1] \times [0, 1]$ with the same number of grid points. This uniform grid is called the computational domain, while the (x, z)-grid is called the physical domain. We calculate the saturation profile on the physical domain, thereby making use of the coordinate transformation so that we can discretize the NERE on the computational domain.

We apply the following coordinate transformation:

$$\begin{aligned} x &= x(\xi, \eta, \theta), \\ z &= z(\xi, \eta, \theta), \\ t &= \theta, \\ S(x, z, t) &= S(\xi, \eta, \theta). \end{aligned}$$

We only transform the spatial variables x and z, therefore we choose θ to be equal to t.

Applying the chain rule to, for example, the derivatives $\frac{\partial S}{\partial t}$, $\frac{\partial S}{\partial x}$ and $\frac{\partial S}{\partial z}$ gives:

$$\begin{aligned} \frac{\partial S}{\partial t} &= \frac{\partial S}{\partial \xi}\frac{\partial \xi}{\partial t} + \frac{\partial S}{\partial \eta}\frac{\partial \eta}{\partial t} + \frac{\partial S}{\partial \theta}\frac{\partial \theta}{\partial t} \\ &= \frac{\partial S}{\partial \xi}\frac{\partial \xi}{\partial t} + \frac{\partial S}{\partial \eta}\frac{\partial \eta}{\partial t} + \frac{\partial S}{\partial \theta}, \end{aligned} \tag{5.1}$$

$$\begin{aligned} \frac{\partial S}{\partial x} &= \frac{\partial S}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial S}{\partial \eta}\frac{\partial \eta}{\partial x} + \frac{\partial S}{\partial \theta}\frac{\partial \theta}{\partial x} \\ &= \frac{\partial S}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial S}{\partial \eta}\frac{\partial \eta}{\partial x}, \end{aligned} \tag{5.2}$$

$$\begin{aligned} \frac{\partial S}{\partial z} &= \frac{\partial S}{\partial \xi}\frac{\partial \xi}{\partial z} + \frac{\partial S}{\partial \eta}\frac{\partial \eta}{\partial z} + \frac{\partial S}{\partial \theta}\frac{\partial \theta}{\partial z} \\ &= \frac{\partial S}{\partial \xi}\frac{\partial \xi}{\partial z} + \frac{\partial S}{\partial \eta}\frac{\partial \eta}{\partial z}. \end{aligned} \tag{5.3}$$

If we now find expressions for ξ_t, η_t, ξ_x, η_x, ξ_z and η_z, we are able to express these three derivatives above entirely on the computational domain. This would give us the possibility to use ordinary finite differences for the discretization for the NERE, because the mesh cells of the computational domain are rectangular.

The expressions we are looking for can be found by assuming that the transformation is invertible, for in that case the product of the Jacobian matrix of the transformation and its

inverse should equal the identity matrix [8]:

$$\begin{bmatrix} \xi_x & \xi_z & \xi_t \\ \eta_x & \eta_z & \eta_t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\xi & x_\eta & x_\theta \\ z_\xi & z_\eta & z_\theta \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

By working out this product we obtain six equations with six unknowns. Solving these equations gives us the following expressions:

$$\xi_x = \frac{z_\eta}{\mathcal{J}}, \qquad \xi_z = -\frac{x_\eta}{\mathcal{J}}, \qquad \xi_t = -\frac{x_\theta z_\eta - z_\theta x_\eta}{\mathcal{J}},$$
$$\eta_x = -\frac{z_\xi}{\mathcal{J}}, \qquad \eta_z = \frac{x_\xi}{\mathcal{J}}, \qquad \eta_t = \frac{x_\theta z_\xi - z_\theta x_\xi}{\mathcal{J}},$$

where $\mathcal{J} = x_\xi z_\eta - x_\eta z_\xi$, the determinant of the Jacobian matrix. By including these expressions into Equations (5.1)-(5.3), we end up with expressions for $\frac{\partial S}{\partial t}$, $\frac{\partial S}{\partial x}$ and $\frac{\partial S}{\partial z}$ that can be discretized by using a finite difference method on a uniform grid.

Recall from previous chapters that the NERE is given by

$$\frac{\partial S}{\partial t} = \nabla \cdot (D(S)\nabla S) + \frac{\partial K(S)}{\partial z} + \tau \nabla \cdot \left[K(S) \nabla \left(\frac{\partial S}{\partial t} \right) \right].$$

As an example, we work out part of the term $\nabla \cdot (D(S)\nabla S)$. Recall that

$$\nabla \cdot (D(S)\nabla S) = \frac{\partial}{\partial x}\left(D(S)\frac{\partial S}{\partial x} \right) + \frac{\partial}{\partial z}\left(D(S)\frac{\partial S}{\partial z} \right).$$

We make a discretization for the term with derivatives with respect to x. To that end, we write [8]:

$$\begin{aligned} S_x &= S_\xi \xi_x + S_\eta \eta_x \\ &= S_\xi \frac{z_\eta}{\mathcal{J}} - S_\eta \frac{z_\xi}{\mathcal{J}} \\ &= \frac{1}{\mathcal{J}} \left[S_\xi z_\eta + S z_{\eta\xi} - S z_{\xi\eta} - S_\eta z_\xi \right] \\ &= \frac{1}{\mathcal{J}} \left[(S z_\eta)_\xi - (S z_\xi)_\eta \right]. \end{aligned} \tag{5.4}$$

Note that we assumed that the grid is smooth enough, so that $z_{\eta\xi} = z_{\xi\eta}$. We multiply Equation (5.4) by $D(S)$:

$$D(S) S_x = \frac{D(S)}{\mathcal{J}} \left[(S z_\eta)_\xi - (S z_\xi)_\eta \right].$$

Next, we substitute this expression instead of S into Equation (5.4) to find a discretization

for $(D(S)S_x)_x$:

$$\begin{aligned}(D(S)S_x)_x &= \frac{1}{\mathcal{J}}\left(\frac{D(S)}{\mathcal{J}}\left[S_\xi z_\eta - S_\eta z_\xi\right] z_\eta\right)_\xi - \frac{1}{\mathcal{J}}\left(\frac{D(S)}{\mathcal{J}}\left[S_\xi z_\eta - S_\eta z_\xi\right] z_\xi\right)_\eta \\ &= \frac{1}{\mathcal{J}}\left[\left(\frac{D(S)z_\eta^2}{\mathcal{J}}S_\xi\right)_\xi - \left(\frac{D(S)z_\xi z_\eta}{\mathcal{J}}S_\eta\right)_\xi - \right. \\ &\qquad\qquad \left.\left(\frac{D(S)z_\xi z_\eta}{\mathcal{J}}S_\xi\right)_\eta + \left(\frac{D(S)z_\xi^2}{\mathcal{J}}S_\eta\right)_\eta\right] \\ &= \frac{1}{\mathcal{J}}\left[(C_1S_\xi)_\xi - (C_2S_\eta)_\xi - (C_2S_\xi)_\eta + (C_3S_\eta)_\eta\right],\end{aligned}$$

with $C_1 = \frac{1}{\mathcal{J}}D(S)z_\eta^2$, $C_2 = \frac{1}{\mathcal{J}}D(S)z_\xi z_\eta$ and $C_3 = \frac{1}{\mathcal{J}}D(S)z_\xi^2$. We approximate this equation by central finite differences and we average the terms that contain half indices:

$$\begin{aligned}(D(S)S_x)_x|_{i,j}^n \approx \frac{1}{\mathcal{J}_{i,j}^n}\Bigg[&\frac{C_1|_{i+1,j}^n + C_1|_{i,j}^n}{2}\frac{S_{i+1,j}^{n+1} - S_{i,j}^{n+1}}{(\Delta\xi)^2} - \frac{C_1|_{i,j}^n + C_1|_{i-1,j}^n}{2}\frac{S_{i,j}^{n+1} - S_{i-1,j}^{n+1}}{(\Delta\xi)^2} \\ &- C_2|_{i+1,j}^n\frac{S_{i+1,j+1}^{n+1} - S_{i+1,j-1}^{n+1}}{4\Delta\xi\Delta\eta} + C_2|_{i-1,j}^n\frac{S_{i-1,j+1}^{n+1} - S_{i-1,j-1}^{n+1}}{4\Delta\xi\Delta\eta} \\ &- C_2|_{i,j+1}^n\frac{S_{i+1,j+1}^{n+1} - S_{i-1,j+1}^{n+1}}{4\Delta\xi\Delta\eta} + C_2|_{i,j-1}^n\frac{S_{i+1,j-1}^{n+1} - S_{i-1,j-1}^{n+1}}{4\Delta\xi\Delta\eta} \\ &+ \frac{C_3|_{i,j+1}^n + C_3|_{i,j}^n}{2}\frac{S_{i,j+1}^{n+1} - S_{i,j}^{n+1}}{(\Delta\eta)^2} - \frac{C_3|_{i,j}^n + C_3|_{i,j-1}^n}{2}\frac{S_{i,j}^{n+1} - S_{i,j-1}^{n+1}}{(\Delta\eta)^2}\Bigg].\end{aligned}$$

This is the discretization for one of the terms of the NERE. We do not work out the other terms here, as all other terms can be treated in a similar way.

Note that C_1, C_2 and C_3 are calculated at time step n, while S is calculated at time step $n+1$. Here, the same procedure as in Chapter 3 is applied: we linearize the nonlinear terms by approximating them at time step n, so that we can solve a linear system of equations in each time step for calculating S^{n+1} from S^n.

By making a discretization of the entire NERE in the way that was described here, thereby using the IMEX-θ method, we arrive at a system of linear equations. We solve this system at each time step in the same way as in Chapter 3, namely with the Bi-CGSTAB method.

5.1.2 The adaptation of the mesh

Now that we know the procedure to make a discretization for the NERE on an adaptive mesh, the question that remains is how we can actually adapt the mesh. This is done by solving the so called mesh PDEs. These mesh PDEs are partial differential equations that are derived by minimizing a mesh energy functional. Their derivation is discussed in more detail in [11].

The adaptive mesh PDEs

In order to move the mesh we use the following two adaptive mesh PDEs:

$$\begin{aligned} \nabla \cdot (\omega \nabla x) &= 0, \\ \nabla \cdot (\omega \nabla z) &= 0, \end{aligned} \tag{5.5}$$

where $\nabla = \begin{pmatrix} \partial/\partial\xi \\ \partial/\partial\eta \end{pmatrix}$ and $\omega > 0$ the monitor function. This monitor function determines how the grid points are divided over the domain.

In order to adapt the mesh these PDEs should be solved in each time step. Instead of solving the elliptic PDEs given by (5.5), we solve the following time dependent equations:

$$\begin{aligned} \nabla \cdot (\omega \nabla x) &= x_{t'}, \\ \nabla \cdot (\omega \nabla z) &= z_{t'} \end{aligned} \quad . \tag{5.6}$$

Here t' is an artificial time variable. Note that for the limit $t' \to \infty$ the right-hand side of both equations goes to zero and we obtain (5.5). We solve the PDEs separately by performing a small number of internal time steps, until the difference between two subsequent grids is below some tolerance value. As a boundary condition, we require that the grid points in the corners do not move. Moreover, grid points that are situated on a boundary can move along that boundary, but cannot leave it. In fact, for the boundaries we solve the one-dimensional version of Equation (5.6): $(\omega x_\xi)_\xi = x_{t'}$ for the horizontal boundaries and $(\omega z_\eta)_\eta = z_{t'}$ for the vertical boundaries.

For the simulations in this chapter we perform at most two internal time steps each time the mesh is adapted. We do not use more time steps because then the mesh might move to fast, which leads to errors or even an unstable numerical solution. It is possible that we deviate from the solution of Equation (5.5) in this way, but the results of our simulatons will show that the mesh still adapts itself well to the saturation profile.

For solving the PDEs we use a central finite difference approximation and the Euler Backward method. The monitor function ω is calculated on beforehand, so that the system of equations resulting from the discretization is linear. For the second internal time step, ω is not calculated again. The system of equations is solved by using the Bi-CGSTAB method.

The monitor function

The monitor function determines the division of the grid points over the physical domain. If the monitor function is relatively large in a certain area of the domain, then this area obtains many grid points compared to areas with a relatively low monitor value.

When simulating the finger phenomenon, in a large part of the domain the saturation is more or less constant in time and in space. These areas are not very interesting, we are mainly interested in the behavior of the wetting front and of the fingers. In these areas the saturation has a high gradient and therefore it is quite obvious to choose a monitor function that is based on the spatial variation of the saturation. We choose the following monitor function:

$$\omega = \gamma(t) + \sqrt{\nabla S \cdot \nabla S},$$

where

$$\gamma(t) = \int_0^1 \int_0^1 \sqrt{\nabla S \cdot \nabla S} d\xi d\eta.$$

The idea for this monitor function comes from [11]. The function $\gamma(t)$ controls the amount of adaptivity: a larger value for γ decreases the amount of adaptation and vice versa. The entire monitor function is calculated on the computational domain. The double integral in the calculation of $\gamma(t)$ is approximated using the trapezoidal rule.

The transition from high to low spatial variation in the saturation profile is rather abrupt, as becomes clear by inspecting the figures in Chapter 4. This means that the monitor function contains abrupt changes as well, which results in an unsmooth grid. In order to prevent this, we apply a filter to the monitor function: each monitor value is averaged with the monitor values adjacent to it. The following Gaussian filter is applied:

$$\omega_{i,j}^{\text{filtered}} = \frac{1}{4}\omega_{i,j} + \frac{1}{8}\left(\omega_{i-1,j} + \omega_{i+1,j} + \omega_{i,j-1} + \omega_{i,j+1}\right) + \frac{1}{16}\left(\omega_{i-1,j-1} + \omega_{i+1,j-1} + \omega_{i-1,j+1} + \omega_{i+1,j+1}\right).$$

In case that the mesh is still not smooth enough, this filter can be applied multiple times. We apply it five times, because in case that we choose a small value for β like in Figure 4.7(a), the mesh becomes very irregular as a result of the steep finger tips. An additional benefit of filtering is that the grid just before the moving front is refined already as well.

5.2 Results

For simulating the fingering patterns on an adaptive grid we use a domain of size 10×60 and we reduce the number of grid points in the z-direction by a factor two compared to the experiments for obtaining Figure 4.6. The number of grid points in the x-direction is reduced by a factor three (note that the width of the domain was reduced by a factor three as well). The functions $K(S)$ and $D(S)$ remain unchanged and the same holds for the time step: $\Delta t = 0.05$. We choose $\alpha = 3$; the value for β is varied in this section. The wave number of the perturbation is set to $\omega = 1$.

As a first experiment we solve the NERE for a perturbed initial condition with the following parameters: $\tau = 0.0$ and $\beta = 0.5$. From Chapter 4 we know that the perturbation does not grow in this case and therefore the saturation will not have a high gradient in the x-direction anywhere in the domain. The mesh will mainly have to adapt itself in the z-direction.

Figure 5.1 shows the results of the simulation until $t_{end} = 150$. We see that the adaptive mesh method seems to work well: the mesh is clearly refined around the wetting front. The refinement of the grid mainly comes from points directly below the wetting front.

In the next experiment τ is increased to 7.0 so that the initial perturbation is expected to grow and become a finger, which complicates the problem. In Figure 5.2 it is shown that the mesh is able to follow the fingering structures well: the wetting front as well as the finger tip obtain a finer mesh than the rest of the domain. If the mesh would not adapt itself, the number of grid points in the z-direction would equal 20 in the zoomed-in regions. However, because of the adaptation these regions obtained much more grid points: approximately 40.

We can obtain a numerically more complicated problem by decreasing the amount of diffusion in the model. We do this by setting β equal to 0.2. The results of this simulation, which runs until $t_{end} = 200$, are shown in Figure 5.3. The mesh points cluster around the entire wetting front and the regions close to to the finger tip where the gradient is high. However, the refinement still mainly comes from grid points below the wetting front. The adaptive grid method is able to handle this problem; however, it could be improved so that the grid points of the top of the domain are used for the refining the grid in the regions with high gradient as well.

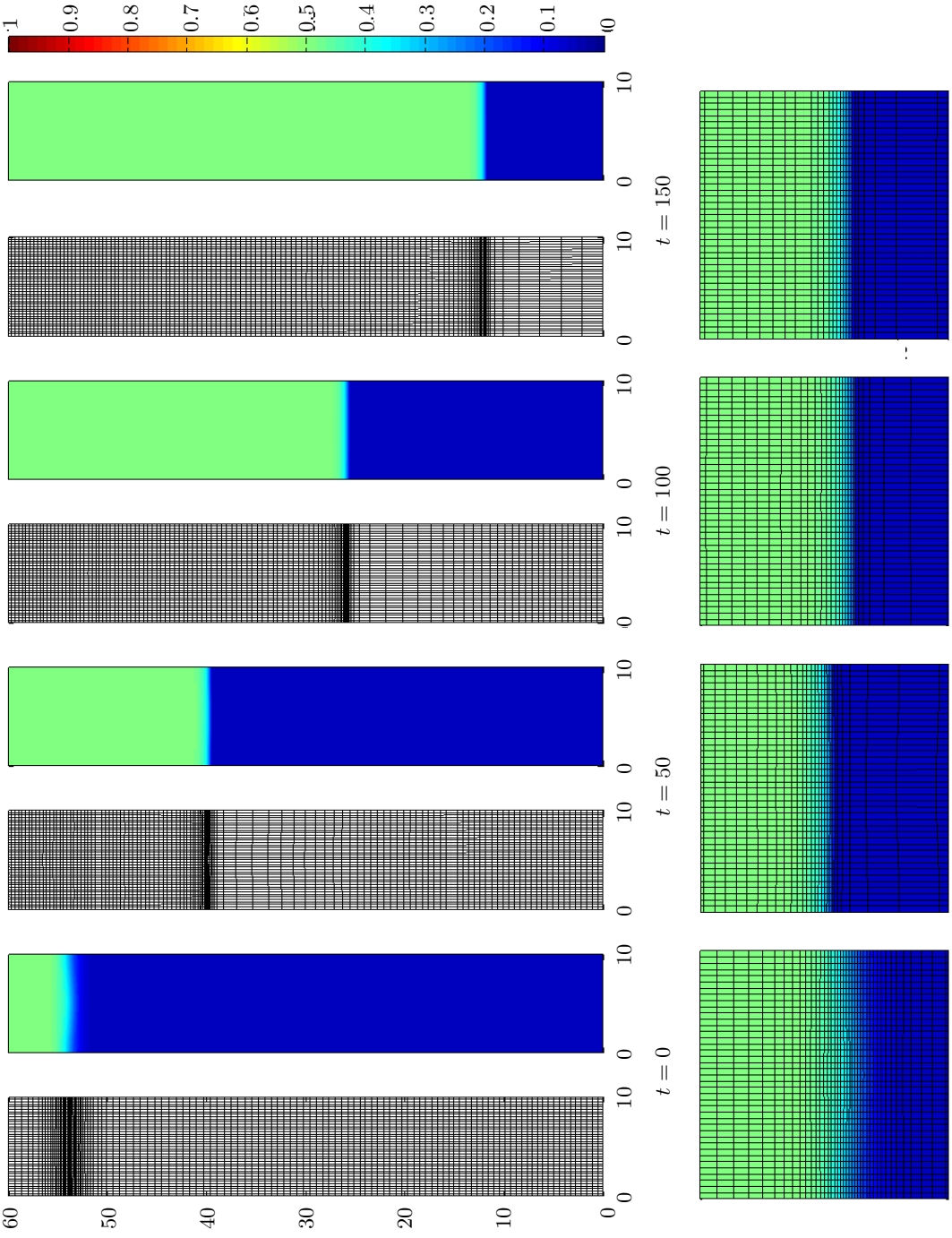


Figure 5.1: The saturation profile and corresponding mesh at different times for $\tau = 0.0$ and $\beta = 0.5$. The figures below the saturation profiles zoom in at the region around the wetting front; their size is 10×10.

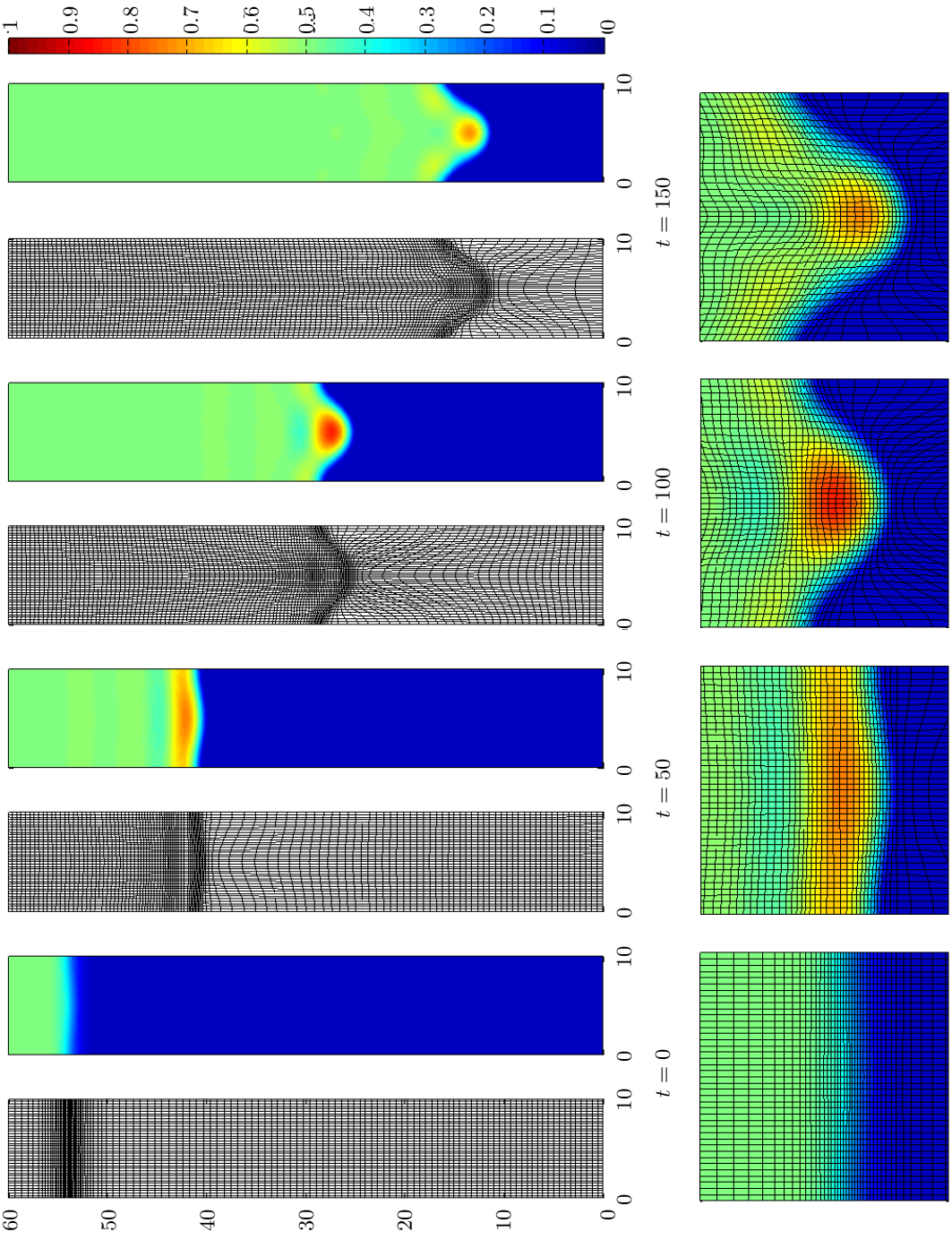


Figure 5.2: The saturation profile and corresponding mesh at different times for $\tau = 7.0$ and $\beta = 0.5$. The figures below the saturation profiles zoom in at the region around the wetting front; their size is 10×10.

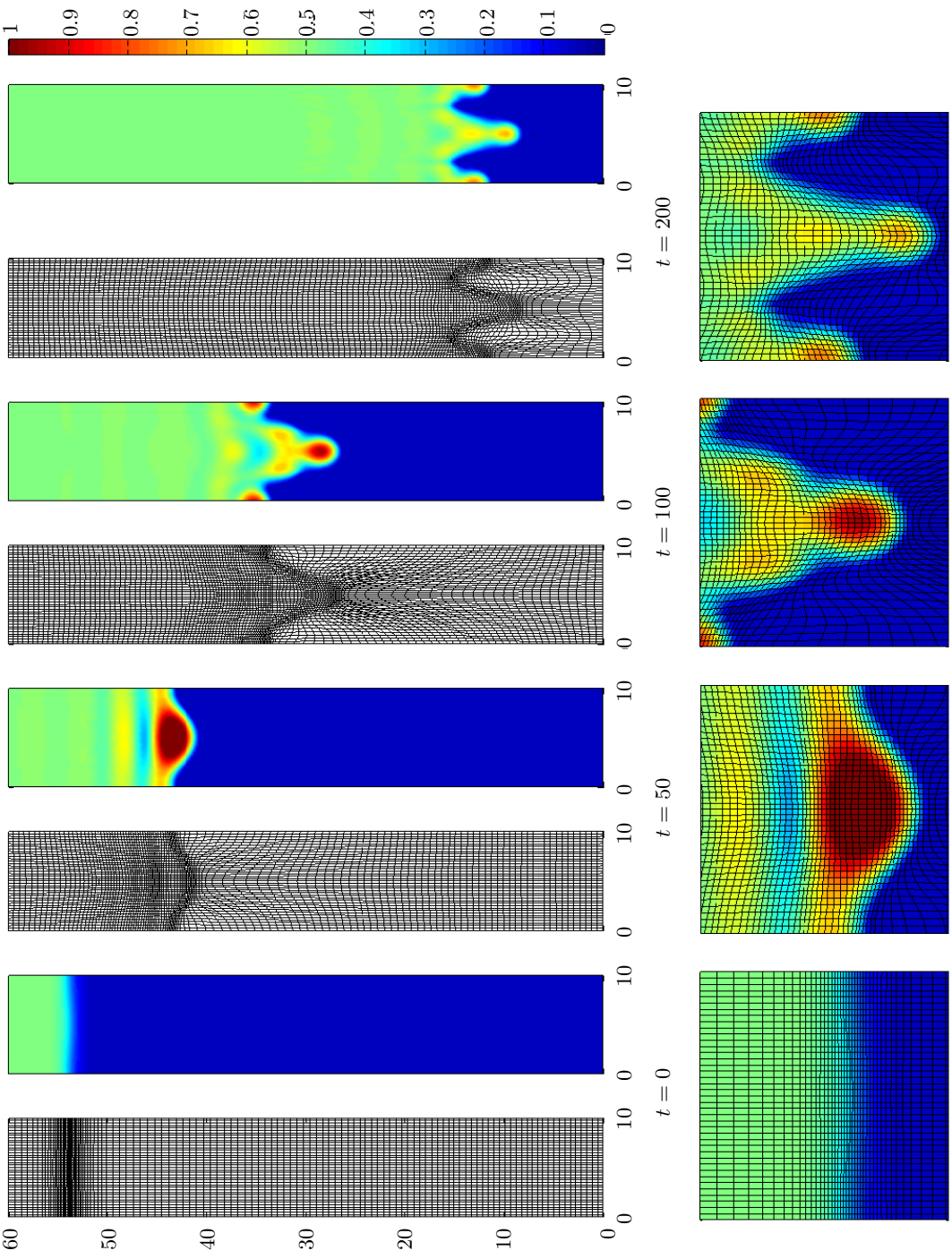


Figure 5.3: The saturation profile and corresponding mesh at different times for $\tau = 7.0$ and $\beta = 0.2$. The figures below the saturation profiles zoom in at the region around the wetting front; their size is 10×10.

Chapter 6

Conclusion

In this report we investigated the effect of small perturbations to the flow field of the nonequilibrium Richards Equation. From theoretical research it was already known that the flow described by the NERE is conditionally unstable to small perturbations. We examined whether we observed the same behavior when solving the NERE numerically.

We can easily answer the question whether a flow is stable or unstable by examining the saturation profile in time. In case that the small initial perturbations grow and become fingering patterns, the flow is unstable. On the other hand, in case that the perturbations fade away, the flow is stable. In order to compare the numerical results with the theoretical results we had to somehow quantify the amount of stability or instability. We did this by calculating the growth factor of the perturbation by means of comparing an unperturbed flow field with a perturbed one.

Using simulations of the NERE for various values of the parameter τ, we constructed a diagram of the growth factor of the perturbation as a function of the wave number of the perturbation. This diagram showed that, in case that the wave number is low and in case that τ is large enough, the flow becomes unstable. A larger value for τ generally yielded a larger growth factor.

The diagram we constructed did not match exactly with the diagram that was constructed by means of theoretical stability analysis. The reason for this is that we did not use the same functions and domain. However, our conclusions are the same:

1. there is a critical value for τ above which the flow can become unstable;
2. the flow only becomes unstable in case the wave number of the initial perturbation is small enough. The definition of small enough depends on the value of τ.

As a conclusion, we can say that our experiments clearly confirm the behavior of the flow field described by the NERE with respect to small initial perturbations that is expected by the theoretical stability analysis.

The diagram was constructed using a finite difference method on a regularly spaced grid. However, a very small grid was required in order to do accurate computations, which resulted in long computation times. Since the saturation profile contains both regions with high and low spatial variation when simulating fingering patterns, it seemed convenient to use an

adaptive grid. The numerical solution of the NERE on an adaptive grid appeared to work well: when lowering the number of grid points by a factor two, the wetting front and finger tips still could be modelled with a fine grid.

In the future, our work could be extended by using a more complicated and realistic model for simulating the fingering phenomenon, so that the numerical experiments agree better with laboratory experiments. Moreover, the adaptive program could be optimized further in order to obtain a better speed up compared to the program that uses a fixed mesh and in order to improve the refinement of the grid.

Bibliography

[1] Egorov, A. G., Dautov, R. Z., Nieber, J. L., and Sheshukov, A. Y. Stability analysis of gravity-driven infiltrating flow. *Water Resources Research Vol. 39*, No. 9 (2003), 1266.

[2] Gray, W. G., and Hassanizadeh, S. M. Unsaturated flow theory including interfacial phenomena. *Water Resources Research Vol. 27*, No. 8 (August 1991), 1855–1863.

[3] Hundsdorfer, W., and Verwer, J. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, vol. 33 of *Springer Series in Computational Mathematics*. Springer, Berlin, 1993.

[4] Nieber, J. L., Dautov, R. Z., Egorov, A. G., and Sheshukov, A. Y. Dynamic capillary pressure mechanism for instability in gravity-driven flows; review and extension to very dry conditions. *Transp. Porous Media Vol. 58* (2005), 147–172.

[5] Philip, J. R. The theory of infiltration: 4. Sorptivity analysis of algebraic infiltration equations. *Soil Science Vol. 84* (1957), 257–264.

[6] Ruuth, S. J. Implicit-explicit methods for reaction-diffusion problems in pattern formation. *Journal of Mathematical Biology Vol. 34* (1995), 148–176.

[7] Skalický, T. LASPack Version 1.12.2, 1992-1995.

[8] Sypkens Smit, M. Three-dimensional adaptive moving meshes and their visualization. Master's thesis, Universiteit Utrecht, 2005.

[9] Weisstein, E. W. Least squares fitting–exponential. From MathWorld–A Wolfram Web Source, http://mathworld.wolfram.com/LeastSquaresFittingExponential.html. Visited August 12, 2010.

[10] Weisstein, E. W. Least squares fitting. From MathWorld–A Wolfram Web Source, http://mathworld.wolfram.com/LeastSquaresFitting.html. Visited August 12, 2010.

[11] Zegeling, P. A. On resistive MHD models with adaptive moving meshes. *Journal of Scientific Computing Vol. 24*, No. 2 (August 2005), 263–284.