SJÄLVSTÄNDIGA ARBETEN I MATEMATIK

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET

On a Problem of Burgers' Equation with Homogeneous Neumann Boundary Conditions

av

Erik Boström

2014 - No 7

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET, 106 91 STOCKHOLM

On a Problem of Burgers' Equation with Homogeneous Neumann Boundary Conditions

Erik Boström

Självständigt arbete i matematik 15 högskolepoäng, Grundnivå

Handledare: Yishao Zhou

2014

On a Problem of Burgers' Equation with Homogeneous Neumann Boundary Conditions

by

Erik Boström

A thesis submitted in partial fulfillment for the degree of
Bachelor of Science in Mathematics
at Stockholm University

Abstract

This thesis is concerned with the occurrence of problems corresponding to the numerical treatment of the viscid Burgers' equation together with homogeneous Neumann boundary conditions. It has been shown that the steady state solutions of this system must be constants for an arbitrary initial condition, but for the same problem and for some specific initial conditions, the numerical solutions indeed converge to non-constant steady state solutions. We proved analytically that for arbitrary small non-zero Neumann conditions, the steady states are non-constant and in the shape of a tanh function. Since numerically treated derivatives must be approximated, the homogeneous Neumann conditions are in general approximated by a value up to the size of the machine epsilon of the used floating point format. Thus, these wrong non-constant solutions are existing numerical steady states for the homogeneous problem. It has also been shown that these non-constant steady states are indeed not uniquely defined. For each initial value problem there exist two steady state solutions that mainly depend on the size of the error of the Neumann conditions, the viscosity parameter and the magnitude of the initial condition. The convergence therefore depends on which floating point format we use, since the round off that can occur in the approximation of the Neumann conditions are larger for less accurate formats.

During numerical testing another problem was also found. Most likely it is also caused by the round off errors. Since all constants are steady state solutions, there are no globally defined attractor to the problem, and hence different initial conditions have different steady states. Because of that, small errors that occur in every iteration in the numerical process cannot be cured, since the solution in every iteration can be seen as an initial condition by itself. Hence, in some cases the solution seems to converge to the right steady state, but makes a drastic change to a wrong constant solution. Looking more closely, for an initial condition with an invariant point and for which the steady state solution is the zero constant we can see that the point is shifted by a small value in each iteration. We have approached this problem using an initial condition with a invariant point in which we imposed a Direchlet condition. With this setting the problem was eliminated, which indicates that the errors appear most likely due to the round off errors that occur when the zero value in the point is approximated.

We have also proved that the non-constant solutions does not exist for all numerical schemes. More detailed studies of the impact on specific numerical schemes might therefore be a topic for further studies.

Acknowledgements

I would like to thank my supervisor Yishao Zhou. Your support has meant a lot to me and has made me highly motivated for the task. And, even since it has been a little short of time both to get started with the thesis and to fit in the deadlines, everything has passed smoothly because of you. I will also thank Jan-Erik Björk for reviewing the thesis, and some valuable friends (you know who you are) who have liked to hear and comment about my work during the way.

Contents

1 Introduction **1**

2 Analytical solutions **5**
2.1 General solution using separation of variables 6
2.1.1 Direchlet boundary conditions 8
2.1.2 Neumann boundary conditions 8

3 Steady state solutions **9**
3.0.3 The equilibrium $u = 0$ 10
3.1 Linearization 11
3.2 The class of odd functions around $x = 1/2$ 14
3.3 Steady state solutions in finite precision 15
3.3.1 Existence of the zero steady state in finite precision 16

4 A bifurcation analysis of the finite precision steady states **19**
4.1 Consequences of non-zero Neumann conditions 19
4.2 Stability analysis 21
4.2.1 Linearization 21
4.2.2 Transformation into Sturm Liouville form 22
4.2.3 Eigenvalue approximation 24

5 Numerical results **26**
5.1 Existence of numerical solutions 26
5.1.1 Explicit Euler method 27
5.1.2 Crank Nicolson method 28
5.1.3 Finite element method 29
5.2 Results 30
5.2.1 ν fixed, varying magnitude of the initial condition 31
5.2.2 Magnitude of the initial condition fixed, varying ν 33
5.3 Approximation of roots and derivatives 34
5.3.1 Wrong solutions because of the approximation at $x = 1/2$ 37
5.4 Solutions in different decimal formats 38

5.5 Monotonically increasing odd initial conditions 38
5.6 Comparison with the non-homogeneous problem 39

6 Possible treatment on boundary conditions **41**

7 Conclusions **43**

8 Appendix: Implementation of numerical schemes **45**
8.1 Explicit Euler implementation 45
8.2 Crank Nicolson implementation 46
8.3 A piecewise linear finite element implementation 48

Chapter 1

Introduction

Named after the Dutch physicist Johannes Martinus Burgers (1895-1981), the Burgers' equation is a famous partial differential equation. It appears in applied mathematics as a fundamental model of non-linear phenomenon. Burgers presented the equation as a simple one-dimensional model for turbulent flow. The equation was later derived by James Lighthill (1924-1998) to a second order approximation of the Navier Stokes equation in fluid mechanics. One can consider the Burgers' equation as a simplification of the Navier Stokes equation, where the pressure term is dropped.

The uses of Burgers' equation are many. For instance it can be used to model flow problems such as shock flow and traffic flow. But depending on the nature of the problem it can also be used in areas of heat conduction, thermal radiation, chemical reactions etc. It is known as the simplest model that includes the non-linear and viscous effects of fluid dynamics.

Burgers' equation is also a useful equation for general testing. In this matter, the reason it is to prefer is that it is simple enough to give an insight into more complex problems. Hence, Burgers' equation is often the first choice as a test model in numerical analysis to illustrate accuracy and convergence of a particular scheme.

Burgers' equation is mainly stated in two forms: The viscous Burgers' equation is the complete form, written as

$$u_t + uu_x = \nu u_{xx}, \tag{1.1}$$

where $\nu > 0$ is the viscosity parameter, u is the solution variable; u_t defines the derivative in time and u_x the derivative in space. The physical interpretation of the terms is that uu_x controls convection and νu_{xx} diffusion. The second form is the inviscid Burgers' equation, where the viscosity constant is set to zero:

$$u_t + uu_x = 0. \tag{1.2}$$

The solutions of the inviscid equation can be considered by studying the characteristics of

the equation. As long as the characteristics does not cross there will be unique solutions. But due to the non-linear term, the characteristics may cross at some time, which will cause non-unique solutions. Non-unique solutions cannot exist in most physical situations.

If the viscosity parameter is non-zero, then the diffusion term works as a control term, restricting existence of non-unique solutions. As the wave starts to break the second derivative u_{xx} grows much faster than u_x and νu_{xx} starts to influence the solution. It shows that the νu_{xx} term keeps the solution smooth for all time. Hence the viscous Burgers' equation does not generate any shock wave solutions.

This thesis only deals with results corresponding to the viscous Burgers' equation. However, the solutions of the inviscid Burgers' equation are a widely studied subject. See e.g. [9].

Consider again the shape of the viscous equation (1.1); the equation is written in quasi linear form. Sometimes the uu_x term is not handy to use. One may instead rewrite the term to conservative form: $(F(u))_x = \left(u^2/2\right)_x$. Hence, the equation (1.1) is reformed to:

$$u_t + \left(\frac{u^2}{2}\right)_x = \nu u_{xx}. \tag{1.3}$$

This form will be used several times in this thesis.

The existence of explicitly given analytical solutions to a differential equation makes the use of numerical approximations crucial in many areas. Also, there is a need of simulations in science which cannot for sure be done analytically. Even since the performance of the computers has been increasing a lot over the recent decades, the precision of an arithmetic operation will always be limited by the floating point format of the computer. The non-existence of infinite memory will always prevent the use of infinite numbers on a computer system.

The most used computer systems at date (2014) are single and double precision formats. I.e. Systems with 32 and 64 bits memory per number stored. Every part of the number must be stored in a different place in the memory. The distribution of the bits in single precision is stored as follow: one bit for the sign ($-$ or $+$), 8 bits for the exponent and 23 bits for the fraction. In double precision arithmetic the distribution is: one bit for the sign, 11 bits for the exponent, and 52 bits for the fraction. In Figure 1.1 the the distribution is shown in more detail. All numbers expressed on a computer system is rational

$$\underbrace{\times_2}_{\text{Sign}}\ \underbrace{\times_2\ \times_2 \cdots \times_2\ \times_2}_{\text{Exponent}}\ \underbrace{\times_2\ \times_2 \cdots \times_2\ \times_2}_{\text{Fraction}}$$

Figure 1.1: Bit occupation for one number in finite precision arithmetic.

numbers. For a number to be expressed exactly in base two, the denominator has to be powers of two. Numbers with a prime factor other than two as a denominator cannot be

represented with a finite binary expansion. In double precision arithmetic 52 bits for the fraction can be stored, the rest is truncated. Similarly, in single precision arithmetic there is only place for 23 bits, the rest is truncated. While an arithmetic operation is executed, the result may therefore be truncated. And the smallest number that is stored but not truncated after a arithmetic operation is performed is called the machine epsilon. The machine epsilon is for double respectively single precision

$$\epsilon_{double} = 2^{-53} \approx 1.1102 \cdot 10^{-16}, \tag{1.4}$$

$$\epsilon_{single} = 2^{-24} \approx 5.9605 \cdot 10^{-8}, \tag{1.5}$$

which is the representation where the last bit is 1 and the rest 0 for the fraction part. One may not mix this with the smallest number possible that can be represented, since that number also uses the exponential part of the memory. E.g. the smallest number in double precision is 10^{-325}. More about floating point arithmetic can be seen in [8].

In this thesis the impact of the round off errors that occur when we are dealing with values close to machine epsilon are treated. Consider the Neumann boundary value problem governed by the viscous Burgers' equation:

$$\begin{cases} u_t + uu_x = \nu u_{xx}, & (x,t) \in (0,1) \times (0,\infty),\ \nu > 0, \\ u_x = 0, & (x,t) \in \{0,1\} \times (0,\infty), \\ u = u_0, & (x,t) \in [0,1] \times \{0\}. \end{cases} \tag{1.6}$$

where u_0 is an initial condition over $[0,1]$. Solving (1.6) numerically, the Neumann conditions must be zero in each iteration using a numerical time-stepping method. Dealing with zeros against non-zeros, especially values that we expect being computed as zero, is most often giving round off errors. Hence, trying to compute something such that the result is zero can be changed to compute something to a value close to machine epsilon in finite precision. It is found that the viscid Burgers' equation with zero Neumann conditions and for an arbitrary initial condition does generate non-existing solutions. It is easy to show that for all initial conditions the steady state solution must be constant, but the numerical results shows something else. Instead of a constant solution, the solution converges to a non-constant shock-wave looking solution, see Figure 1.2.

There has been some research going on treating this problem over the years. Indeed, it seems to be the round off error while using the zero Neumann conditions that is the main source to these wrong solutions. We are considering this in more detail. The ground of this thesis follows from the research papers by Allen, Burns, Balogh, Gilliam, Hill and Shubow [2]-[3], which first approached the problem at the nineties. Also, the master students Pugh and Nguyen has done numerical testing of different finite element schemes in their master theses [11], [10], which has been valuable. Other papers that has influenced the content of this thesis is the work by Titi and Cao [5], which are treating the problem from a more analytical point of view.

The purpose of this thesis has NOT been to copy the results from the work mentioned above. The goal has been reviewing the problem at an undergraduate level and also give examples of possible error sources that can be studied in more detail for further research.

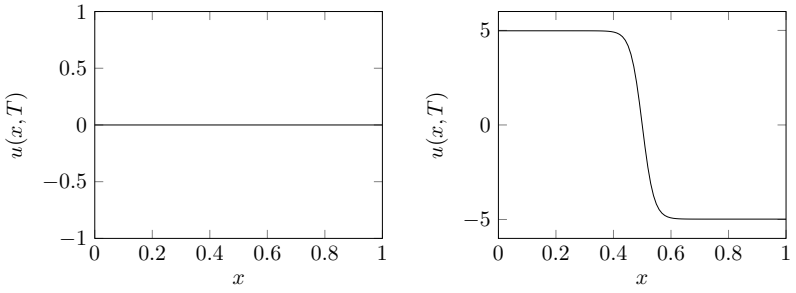


Figure 1.2: (a) Expected steady state solution to the system (1.6) at a time T, for the initial condition $u_0(x) = 5\cos(\pi x)$. (b) Numerical solution to the same problem.

Chapter 2

Analytical solutions

Why using numerical methods and finite precision instead of the analytical analogy? To answer this question, the analytical treatment of Burgers' equation are presented in this chapter. It shows that complexity of the boundary conditions restrict the possibility for an explicitly given solution to exist. The viscid Burgers' equation (1.1) is one of the few non-linear partial differential equations which can be solved exactly for a restricted set of arbitrary initial conditions. Indeed, by changing variables the non-linear equation can be linearized by the so called Cole-Hopf transformation [7]. Hopf introduced the transformation by first rewriting the space derivatives into the following form

$$u_t = \left(\nu u_x - \frac{u^2}{2}\right)_x. \tag{2.1}$$

Then by introducing the dependant variable $\phi = \phi(x,t)$, defined as

$$\phi(x,t) = \exp\left\{-\frac{1}{2\nu}\int_0^x u(s,t)\mathrm{d}s\right\}, \tag{2.2}$$

the final Cole Hopf transformation is reached

$$u(x,t) = -2\nu\frac{\phi_x}{\phi}. \tag{2.3}$$

Theorem 2.0.1. *If $\phi(x,t)$ is any solution to the heat equation*

$$\phi_t(x,t) = \nu\phi_{xx}(x,t), \tag{2.4}$$

then $u(x,t) = -2\nu\phi'(x,t)/\phi(x,t)$ is a solution to the viscid Burgers' equation (1.1).

Proof. Compute the terms in (1.1) individually

$$\begin{aligned} u_t &= 2\nu\frac{\phi_t\phi_x - \phi\phi_{xt}}{\phi^2}, \\ uu_x &= 4\nu^2\frac{\phi_x(\phi\phi_{xx} - \phi_x^2)}{\phi^3}, \\ \nu u_{xx} &= -2\nu^2\frac{2\phi_x^3 - 3\phi\phi_{xx}\phi_x + \phi^2\phi_{xxx}}{\phi^3}. \end{aligned}$$

Substitute into (1.1) yields

$$\begin{aligned} & 2\nu\frac{\phi_t\phi_x - \phi\phi_{xt}}{\phi^2} + 4\nu^2\frac{\phi_x(\phi\phi_{xx} - \phi_x^2)}{\phi^3} = -2\nu^2\frac{2\phi_x^3 - 3\phi\phi_{xx}\phi_x + \phi^2\phi_{xxx}}{\phi^3} \\ \iff & -\phi\phi_{xt} + \phi_x(\phi_t - \nu\phi_{xx}) + \nu\phi\phi_{xxx} = 0 \\ \iff & \phi_x(\phi_t - \nu\phi_{xx}) = \phi(\phi_{xt} - \nu\phi_{xxx}) = \phi(\phi_t - \nu\phi_{xx})_x = 0. \end{aligned}$$

Thus, if ϕ solves $\phi_t - \nu\phi_{xx} = 0$, then (2.3) solves (1.1). □

Any initial condition to (1.1) in the form $u_0(x) = u(x, 0)$ is also easily transformed by (2.3) into the form

$$\phi(x, 0) = \exp\left\{-\frac{1}{2\nu}\int_0^x u_0(\xi)\,\mathrm{d}\xi\right\}. \tag{2.5}$$

Boundary conditions, must also be transformed by (2.3). But, it shows that basic boundary conditions are not always trivial after the transformation is applied. Hence, the simplicity of the equation after applying the Cole-Hopf transform is somewhat limited to the form of the boundary conditions. This difficulties are shown in the next section.

By the Cole-Hopf transformation it suffices to solve the heat equation for the transformed boundary conditions and initial condition. And the heat equation can be solved explicitly by e.g. separating variables and Fourier analysis. Such a solution is the next thing to consider.

2.1 General solution using separation of variables

Consider the viscid Burgers' equation with an arbitrary chosen initial condition u_0 on the real spatial interval $[0, 1]$, where the boundary conditions are not yet decided:

$$\begin{cases} u_t + uu_x = \nu u_{xx}, & (x, t) \in (0, 1) \times (0, \infty),\ \nu > 0, \\ u \text{ satisfies some boundary conditions} & (x, t) \in \{0, 1\} \times (0, \infty), \\ u = u_0, & (x, t) \in [0, 1] \times \{0\}. \end{cases} \tag{2.6}$$

Using Cole-Hopf transform (2.3) it suffices to solve the system governed by the heat equation

$$\begin{cases} \phi_t - \nu\phi_{xx} = 0, & (x,t) \in (0,1) \times (0,\infty),\ \nu > 0, \\ \phi \text{ satisfies some boundary conditions} & (x,t) \in \{0,1\} \times (0,\infty), \\ \phi = \exp\left(-\frac{1}{2\nu}\int_0^x u_0(\xi)\,\mathrm{d}\xi\right), & (x,t) \in [0,1] \times \{0\}. \end{cases} \tag{2.7}$$

Separate, the variable $\phi(x,t)$ into the spatial variable $X(x)$ and the time variable $T(t)$: $\phi(x,t) = X(x)T(t)$. Substitute into (2.7) it follows that

$$(X(x)T(t))_t = \nu(X(x)T(t))_{xx},$$

which is rewritten to

$$X(x)T_t(t) = \nu X_{xx}(x)T(t).$$

Dividing the above equation by $\nu X(x)T(t)$ yields

$$\frac{1}{\nu}\frac{T_t(t)}{T(t)} = \frac{X_{xx}(x)}{X(x)}.$$

The left hand side depend only on t and the right hand side only depends on x; thus both are equal to the same constant if there exists a solution $\phi(x,t) = X(x)T(t)$ to (2.7) so T and X satisfy

$$\begin{aligned} \frac{T_t(t)}{\nu T(t)} &= -\lambda \\ \frac{X_{xx}(x)}{X(x)} &= -\lambda. \end{aligned}$$

where λ is an arbitrary separation constant. It is good to mention that the minus sign is not necessary, but it is useful further on. Hence, the problem turns into solving the eigenvalue problem

$$\begin{cases} -X_{xx} = \lambda X, & x \in (0,1),\ \lambda \in \mathbb{C}, \\ X \text{ satisfies some boundary conditions} & x \in \{0,1\}. \end{cases} \tag{2.8}$$

where λ is the eigenvalue of $-\frac{\partial^2}{\partial x^2}$ and X is the corresponding eigenfunction. Thus, to solve for ϕ, the first thing to do is to solve this eigenvalue problem, then solve for $T(t)$ and a multiplication of the solutions is our solution.

The solutions vary for different boundary conditions. There is no need to show the whole solution process with other boundary conditions than for Neumann conditions, which is the main issue in this thesis. But it is worthwhile considering what happens when the Cole-Hopf transformation is applied to Direclet conditions for comparison purposes.

2.1.1 Direchlet boundary conditions

Homogeneous, Direchlet boundary conditions are easily transformed by the Cole Hopf transformation. If $u(0,t) = u(1,t) = 0$, then the transformation follows

$$u(0,t) = -2\nu\frac{\phi_x(0,t)}{\phi(0,t)} = 0 \implies \phi_x(0,t) = 0, \tag{2.9}$$

$$u(1,t) = -2\nu\frac{\phi_x(1,t)}{\phi(1,t)} = 0 \implies \phi_x(1,t) = 0. \tag{2.10}$$

Thus, in this case, Direchlet conditions are transformed into Neumann conditions.

2.1.2 Neumann boundary conditions

Consider now, the Cole Hopf transformation applied to homogeneous Neumann conditions. On the domain $[0,1]$, these are written as

$$u_x(0,t) = u_x(1,t) = 0. \tag{2.11}$$

But applying the Cole Hopf transformation now yields rather complex expressions.

$$\begin{aligned} & u_x(0,t) = \left(-2\nu\frac{\phi_x(0,t)}{\phi(0,t)}\right)_x = -2\nu\frac{\phi_{xx}(0,t)\phi(0,t) - \phi_x^2(0,t)}{\phi^2(0,t)} = 0, \\ \iff \quad & \phi_{xx}(0,t)\phi(0,t) - \phi_x^2(0,t) = 0 \qquad (2.12) \\ & u_x(1,t) = \left(-2\nu\frac{\phi_x(1,t)}{\phi(1,t)}\right)_x = -2\nu\frac{\phi_{xx}(1,t)\phi(0,t) - \phi_x^2(1,t)}{\phi^2(1,t)} = 0 \\ \iff \quad & \phi_{xx}(1,t)\phi(1,t) - \phi_x^2(1,t) = 0. \qquad (2.13) \end{aligned}$$

Thus, the boundary conditions in this case are non-linear and mixed. This conditions are not easy to apply to the eigenvalue problem obtained from the separation of variables process, which means that other methods have to be developed; in general numerical methods.

Chapter 3

Steady state solutions

Physically, the equilibrium condition of a system are often the most important solutions to analyse. An equilibrium is also called a steady state, and mathematically it means that the solution of a differential equation is constant in time. As an example: a steady state solution of the heat equation with no-flux boundaries is when the heat-flow doesn't change. Another example is for the wave equation, when all waves are gone and the water is still. However, the examples just explained shows stable equilibria – as time passes the solutions converges to these states. But, there are also unstable equilibria. If the stable equilibria are interpreted as attracting points, where all trajectories attracts the point, then the behaviour of the unstable equilibria can be interpreted as repelling points, for which all trajectories leave the point.

The following result is crucial in this thesis.

Theorem 3.0.1. *$v(x)$ is the steady state solution to the viscid Burgers equation with homogeneous boundary conditions (1.6) if and only if $v(x) = C$, where C is a constant value.*

Proof. Every constant function $v(x) = C$ is a steady state solution since all terms in (1.6) are derivatives, which makes a constant vanish. Conversely, let $v(x)$ be any steady state solution to (1.6). Then $v(x)$ satisfies

$$\begin{cases} -\nu v_{xx} + (F(v))_x = 0, & x \in (0,1),\ \nu > 0, \\ v_x = 0, & x \in \{0,1\}. \end{cases} \tag{3.1}$$

where $F(v) = v^2/2$. Integrating (3.1) yields

$$v_x(x) = v_x(0)e^{\int_0^x F'(v(s))\,\mathrm{d}s} = 0. \tag{3.2}$$

And integrating both sides of (3.2) finally gives

$$v(x) = C. \tag{3.3}$$

Hence, all steady state solutions are constants. □

As a consequence of the theorem above – since every constant function is a steady state solution, the steady states of the Neumann boundary value problem cannot be uniquely defined. And since all constants are steady state solutions, we can conclude that the global attractor of the dynamical system is unbounded, since it contains the whole real axis. Hence we might expect to have different steady state solutions for different initial conditions chosen.

3.0.3 The equilibrium $u = 0$

Assume that at $t = t^*$ we have a steady state $u(x, t^*)$, then $u_t(x, t^*) = 0$. The steady state doesn't depend on t, hence define $h(x) := u(x, t^*)$. Substituting into (1.6) yields

$$\begin{cases} \frac{(h')^2}{2} = \nu h'', & x \in (0,1),\ \nu > 0, \\ h' = 0, & x \in \{0,1\}. \end{cases} \tag{3.4}$$

Plugging in a constant value into (3.4) makes every term cancels out, which confirms that all constants are steady states.

Other solutions can be found by integrating both sides of the first equation in (3.4), which yields

$$-\nu h' + \frac{h^2}{2} = C. \tag{3.5}$$

This equation can indeed be solved explicitly in the following way

$$\begin{aligned} & -\nu h' + \frac{h^2}{2} = C. \\ \iff \quad & \nu h' = \frac{h^2}{2} - C \\ \iff \quad & 2\nu h' = h^2 - 2C \\ \iff \quad & 1 = \frac{h'}{\frac{h^2 - 2C}{2\nu}} \\ \iff \quad & x = \int_0^{h(x)} \frac{\mathrm{d}z}{\frac{z^2 - 2C}{2\nu}} + D \end{aligned}$$

Use the substitution $z = \sqrt{2C}\tanh\theta$ (see [6]). Then $\mathrm{d}z = \sqrt{2C}(1 - \tanh^2\theta)\,\mathrm{d}\theta$, and hence

$$\begin{aligned} x &= \int_0^{h(x)} \frac{\sqrt{2C}(\tanh^2\theta - 1)}{\frac{2C}{2\nu}(\tanh^2\theta - 1)}\mathrm{d}\theta + D \\ \iff \theta &= \frac{\sqrt{2C}}{2\nu}(D - x). \end{aligned}$$

Then combining the results gives a final steady state solution on the form

$$h(x) = \sqrt{2C}\tanh\left(\frac{\sqrt{2C}}{2\nu}(D-x)\right) \tag{3.6}$$

Thus, there are two constants that have to be solved out. The boundary conditions can be used to this end. The derivative of (3.6) is

$$h'(x) = -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{2\nu}(D-x)\right) \tag{3.7}$$

And at the boundaries, the derivative of the steady state solution is therefore given as:

$$0 = h'(0) = -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{2\nu}D\right), \tag{3.8}$$

$$0 = h'(1) = -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{2\nu}(D-1)\right). \tag{3.9}$$

Since sech cannot be zero, the only way for (3.8) and (3.9) to be satisfied is to choose the constant C equal to zero. And if $C = 0$, then (3.6) must be equal to zero. Hence, the only constant steady state that deals with this kind of solution is the steady state $h(x) = u(x, t^*) = 0$. There is no possibility at all for (3.6) to be a constant value not equal to zero for all C, ν and D. This makes the zero steady state solution interesting. There are indeed other possibilities for (3.7) to be satisfied. If the argument of sech tends to infinity then sech itself tends to zero. Thus,

$$\lim_{\nu\to 0} -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{2\nu}(D-x)\right) = 0, \tag{3.10}$$

and

$$\lim_{C\to\infty} -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{2\nu}(D-x)\right) = 0. \tag{3.11}$$

So, if either $\nu \to 0$ or $C \to \infty$ the boundary conditions are also satisfied. But, (3.7) is more influential if the boundary conditions are non-zero Neumann conditions, which will be the case if the zero value is approximated to a non-zero. This is the main subject of this thesis, hence we will come back to this later on.

3.1 Linearization

A deeper analysis of a steady state can be performed locally. The linearization principle states that designs based on linearizations work locally for the original system [12]. Hence

by linearizing at the zero state, the equilibrium becomes simpler to study. A linearization is most often performed by the Taylor expansion, where all non-linear terms are dropped. For the Burgers' equation let $u(x,t) = 0 + \delta w(x,t)$, where δ is the magnitude of the increment around the zero equilibrium. Substituting into the general form (1.6) yields:

$$\begin{cases} w_t = \nu w_{xx}, & (x,t) \in (0,1) \times (0,\infty),\ \nu > 0 \\ w_x = 0, & (x,t) \in \{0,1\} \times (0,\infty). \\ w = u_0 & (x,t) \in [0,1] \times \{0\}. \end{cases} \tag{3.12}$$

Thus, the linearized problem at zero is indeed the linear heat equation, which can be solved in the way described in previous section, namely by separating variables: $w(x,t) = X(x)T(t)$. The first part of this process is therefore already done, and it remains to solve for the Neumann boundary conditions.

Recall that the problem that has to be solved is the eigenvalue problem in the form

$$\begin{cases} -X_{xx} = \lambda X, & x \in (0,1),\ \lambda \in \mathbb{C} \\ X_x = 0. & x \in \{0,1\} \end{cases} \tag{3.13}$$

Consider the three cases for λ: $\lambda > 0$, $\lambda < 0$ and $\lambda = 0$. Starting with $\lambda > 0$; the general solution in this case is

$$X(x) = C\cos(\sqrt{\lambda}x) + D\sin(\sqrt{\lambda}x). \tag{3.14}$$

Now, applying the boundary conditions gives

$$0 = X_x(0) = \sqrt{\lambda}D \implies D = 0, \tag{3.15}$$

$$0 = X_x(1) = -\sqrt{\lambda}C\sin(\sqrt{\lambda}) \implies \sqrt{\lambda} = n\pi, \quad n = 1,2,\dots \tag{3.16}$$

Hence, the eigenvalues $\lambda > 0$ with corresponding eigenfunctions of (3.13) are given by

$$\lambda_n = (n\pi)^2, \quad X_n(x) = \cos(n\pi x), \quad n = 1,2,3,\dots \tag{3.17}$$

The case $\lambda = 0$ gives the general solution

$$\varphi(x) = C + Dx. \tag{3.18}$$

And applying the boundary conditions to this equation yields

$$0 = X_x(0) = D \implies X(x) = C. \tag{3.19}$$

Hence, $\lambda = 0$ is an eigenvalue of the boundary value problem and the eigenfunction

corresponding to this eigenvalue is

$$X(x) = 1. \tag{3.20}$$

Finally for $\lambda < 0$, the general solution is

$$X(x) = C\cosh(\sqrt{-\lambda}x) + D\sinh(\sqrt{-\lambda}x). \tag{3.21}$$

And applying the boundary conditions gives

$$0 = X_x(0) = \sqrt{-\lambda}D \implies D = 0 \tag{3.22}$$

$$0 = X_x(1) = \sqrt{-\lambda}C\sinh(\sqrt{-\lambda}). \tag{3.23}$$

But, $\sqrt{-\lambda} \neq 0$, which implies that $\sinh(\sqrt{-\lambda}) \neq 0$. Hence, $C = 0$; so both C and D are equal to zero, which means that there exist no eigenvalues in this interval.

Summarize all three cases – the general solution to the eigenvalue problem is

$$\lambda_n = (n\pi)^2, \quad X_n(x) = \cos(n\pi), \quad n = 0, 1, 2, \ldots \tag{3.24}$$

Now, solving for t – the time dependent problem has the solution

$$T(t) = e^{-kn^2\pi^2 t}. \tag{3.25}$$

Hence, the product-solution becomes

$$w_n = A_n\cos(n\pi x)e^{-kn^2\pi^2 t}, \quad n = 0, 1, 2, \ldots \tag{3.26}$$

This gives the general solution to the linearization of the viscid Burgers' equation:

$$w(x,t) = A_0 + \sum_{n=1}^{\infty} A_n\cos(n\pi x)e^{-kn^2\pi^2 t}. \tag{3.27}$$

It is indeed a cosine-series. For the initial condition it is given that

$$u_0(x) = w(x,0) = A_0 + \sum_{n=1}^{\infty} A_n\cos(n\pi x). \tag{3.28}$$

Multiplying both sides by $\cos(m\pi x)$, $m \in \mathbb{N}$, and integrating over $[0,1]$ then gives

$$\int_0^1 u_0(x)\cos(m\pi x)\,\mathrm{d}x = \int_0^1 A_0\,\mathrm{d}x + \sum_{n=1}^{\infty}\int_0^1 A_n\cos(n\pi x)\cos(m\pi x)\,\mathrm{d}x. \tag{3.29}$$

Since the cosine terms are orthogonal to each other in all cases where $n \neq m$, it follows that the Fourier-series converge to the initial condition u_0, with the smoothness property

$u_0 \in L^2(0,1)$

$$A_n = \begin{cases} \int_0^1 u_0(x)\,\mathrm{d}x, & n = 0, \\ \int_0^1 u_0(x)\cos(n\pi x)\,\mathrm{d}x, & n \neq 0, \end{cases} \tag{3.30}$$

where $\cos(n\pi x)$ are the eigenfunctions corresponding to the eigenvalues $\lambda_n = n^2\pi^2$. Letting $t \to \infty$ makes $e^{-kn^2\pi^2 t} \to 0$ in (3.27). And hence,

$$\lim_{t\to\infty} w(x,t) = A_0. \tag{3.31}$$

This means that the solution converges to a constant steady state for each specifically chosen initial condition. However, since zero is a steady state, and also where the linearization is performed, the only true equilibrium is the zero solution itself. Hence, by (3.31), the zero state is reached from initial conditions with mean value zero.

3.2 The class of odd functions around $x = 1/2$

By the Fourier series solution (3.30) we restrict the initial conditions being square integrable. And by the result (3.31) we can conclude that all initial conditions which has a mean value of zero, shall converge to the zero function. A class of such functions in $[0,1]$ is the class of odd functions around $x = 1/2$. Define this class of functions in $L^2(0,1)$ as

$$L^2_{odd} := \{u \in L^2(0,1) : u(x,t) = -u(1-x,t)\}. \tag{3.32}$$

Also, this class is independent of the viscosity, and solutions to (1.6), which can be checked by plugging in $u(x,t) = -u(1-x)$ into (1.6):

$$\begin{aligned} &- u_t(1-x,t) - u(1-x,t)u_x(1-x,t) = -\nu u_{xx}(1-x,t) \\ \iff &u_t(x,t) + u(x,t)u_x(1-x,t) = \nu u_{xx}(x,t). \end{aligned}$$

For the boundary conditions it follows that

$$u_x(0,t) = u_x(1,t) = 0.$$

And for the initial condition

$$-u(1-x,0) = -u_0(1-x) = u_0(x) = u(x,0).$$

Hence $u(x,t) = -u(1-x,t)$. According to the steady state expression (3.6) – since it is an odd function (well known fact for the tanh function) the D constant can be solved out

using the fact that $h(1/2) = 0$. Thus,

$$\sqrt{2C}\tanh\left(\frac{\sqrt{2C}}{2\nu}\left(D-\frac{1}{2}\right)\right) = 0 \implies D = \frac{1}{2}. \tag{3.33}$$

This form is used in the sequel in this thesis. It is only of interest to consider odd initial conditions, since they shall converge to the constant zero solution, which is the only solution that actually corresponds to the linearized results.

3.3 Steady state solutions in finite precision

Recall that the steady state solutions in $L^2_{odd}(0,1)$ are in the form

$$h(x) = \sqrt{2C}\tanh\left(\frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2}-x\right)\right) \tag{3.34}$$

with the derivative

$$h'(x) = -\frac{C}{\nu}\operatorname{sech}^2\left(\frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2}-x\right)\right) \tag{3.35}$$

where the boundary conditions are

$$h'(0) = h'(1) = -\frac{C}{\nu}\operatorname{sech}^2\left(\frac{\sqrt{2C}}{4\nu}\right) = 0. \tag{3.36}$$

But, in finite precision, one cannot be sure about computing something which generates an exact zero solution. Even though the approximation will be something very close to zero it will still be treated as a constant instead of a zero value, and as we know, performing an arithmetic operation will always end up with errors up to machine epsilon, which are numbers that still are saved in the memory of a computer system. Denote approximate discrete numerical solution as $U \approx u$ and the approximation of zero as the constant γ. Then the analogue system to (1.6) in finite precision is written as

$$\begin{cases} U_t + UU_x = \nu U_{xx}, & (x,t) \in (0,1)\times[0,T),\ \nu > 0, \\ U_x = -\gamma, & (x,t) \in \{0,1\}\times[0,T),\ \gamma > 0. \\ U = U_0, & (x,t) \in [0,1]\times\{0\},\ U_0 \in L^2_{odd}(0,1), \end{cases} \tag{3.37}$$

were T is the final time of the computation, $T < \infty$. According to this system, the same steady state solution as for (1.6) with $u_0 \in L^2_{odd}(0,1)$ is obtained, namely (3.34) – but now the constant C inside (3.34) must be non-zero to satisfy the boundary conditions. I.e. the

constant has to be solved out from:

$$h'(0) = h'(1) = -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{4\nu}\right) = -\gamma, \tag{3.38}$$

Since $C \neq 0$, the steady state solutions must be a non-zero function in the form of (3.34). Note that the non-zero derivative must be negative for $-\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{4\nu}\right)$ to exist. In the case $\gamma < 0$, the solutions are truly non-existent. Hence, if this case appear, the round off errors may not give rise to any solutions on the form (3.34) at all. And for sure, there cannot be any constant solutions either since they all must have zero derivatives at the boundaries. However, the small positive values are very small, and their existence does not influence the existence of any other types of solutions, which makes them most likely converge to an approximate value close to the true one. Solutions for monotonically increasing and monotonically decreasing initial conditions has been tested, and will be considered in section 5.5.

The shape of the steady states and the derivatives for two different C are depicted in Figure 3.1. Note that as C is getting smaller, then $h(x)$ is getting closer to the zero solution.

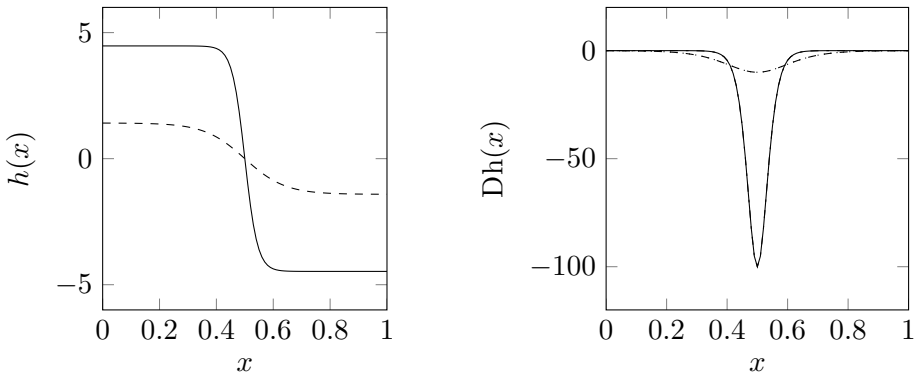


Figure 3.1: Stationary solutions with $\nu = 0.1$, where $C = 1$ [- -] and $C = 10$ [–].

3.3.1 Existence of the zero steady state in finite precision

It was shown in section 3.0.3 that (3.38) tends to zero if either $\nu \to 0$ or $C \to \infty$. Hence, for an iterative process the derivatives on the boundaries may be non-zero at one iteration, but zero at the next, due to round off errors that occur when ν is small enough or C is large enough. Let us consider what values of ν and C that may give $\gamma = 0$. Both single and double precision are considered:

First consider the case where ν is fixed to 0.1 and C is varying. Then the opposite case, where $C = 1$ and ν is varying. Recall that a subnormal value is a number in a

floating point system on the form $M \times 2^e$, $M \in [1, 2)$, which are numbers smallest than $2^{e_{min}}$, where $e_{min} = -1022$ in double precision and $e_{min} = -126$ in single precision (see [8]). The smallest subnormal value is the smallest number existing in the actual floating point format.

Single precision

Let $\nu = 0.1$, then for the derivative expression (3.38) to be smaller than the smallest subnormal value, the following inequality has to be fulfilled

$$10C \operatorname{sech}^2\left(\frac{\sqrt{2C}}{0.4}\right) < 1.4 \times 10^{-45}. \tag{3.39}$$

Solving for C gives that the result must satisfy $C > 253.137$ or $C < 1.4 \times 10^{-45}$. But $C < 1.4 \times 10^{-45}$ is already smaller than the smallest value by itself and must be approximated as zero. Hence in this setting, the only way for (3.38) to be rounded down to zero is if $C > 253.137$.

For a fixed $C = 1$, (3.38) is rounded down to zero if

$$\frac{1}{\nu} \operatorname{sech}^2\left(\frac{\sqrt{2}}{4\nu}\right) < 1.4 \times 10^{-45}. \tag{3.40}$$

And solving for ν gives the result $\nu < 0.0064452$ or $\nu > 10^{45}$. Hence, large ν that satisfying the inequality must be larger than the largest possible number, which is not possible due to memory restrictions. But $\nu < 0.0064452$ are valid numbers in single precision.

Double precision

Similarly for double precision:

Let $\nu = 0.1$, then for (3.38) to be smaller than the smallest subnormal value, the following inequality has to be fulfilled

$$10C \operatorname{sech}^2\left(\frac{\sqrt{2C}}{0.4}\right) < 4.94 \times 10^{-324}. \tag{3.41}$$

Solving for C gives that the result must satisfy $C > 11475$ or $C < 4.94 \times 10^{-324}$. Thus, the same problem occur here as for single precision: small C cannot force (3.38) to zero, but $C > 11475$ are valid numbers in double precision.

Lastly, for a fixed $C = 1$, the derivative expression (3.38) is rounded down to zero if

$$\frac{1}{\nu} \operatorname{sech}^2\left(\frac{\sqrt{2}}{4\nu}\right) < 4.94 \times 10^{-324}. \tag{3.42}$$

And solving for ν gives $\nu < 0.000939$ or $\nu > 2.02 \times 10^{323}$. Similar to single precision, the large number that force (3.38) to zero is so big it may give memory overflow. But the small numbers $\nu < 0.000939$ are valid numbers in double precision.

These results tell that cases for which the derivative is rounded down to zero in the two most common finite precision formats, may occur if ν is very small or if C is very large. But these numbers are much smaller/larger than machine epsilon for respective format. Hence, it is impossible to get consistent results with this sizes of numbers. Also, the numerical schemes will have stability problems with numbers like this. This is not proved, but stability problems have been noticed even for unconditionally stable schemes when $\nu < 0.01$ and $C > 50$.

Another thing according to these results is; if the derivatives are rounded down to zero in this way, the solution (3.34) must be a non constant value with very large magnitude on the boundaries. Because, if the argument of tanh is large, the magnitude of the term $\sqrt{2C}$ will be demanding. In general: $h(0) \to \sqrt{2C}$ as $\sqrt{2C}/(4\nu) \to \infty$.

Chapter 4

A bifurcation analysis of the finite precision steady states

In this chapter, we analyse the steady state $u = 0$ further. Assuming there exist both the zero solution and a non-constant solution, it is of interest to know when those solutions occur, and if the equilibria are stable or unstable.

4.1 Consequences of non-zero Neumann conditions

It was observed in the previous chapter that because of round off errors in finite precision the Neumann conditions in (1.6) are more likely written equal to a small negative constant $-\gamma$ instead of zero for a discretized system. Assuming this fact, the expression for the derivative of the steady state solution on the left boundary is therefore given by

$$h'(0) = h'(1) = -\frac{C}{\nu}\text{sech}^2\left(\frac{\sqrt{2C}}{4\nu}\right) = -\gamma. \tag{4.1}$$

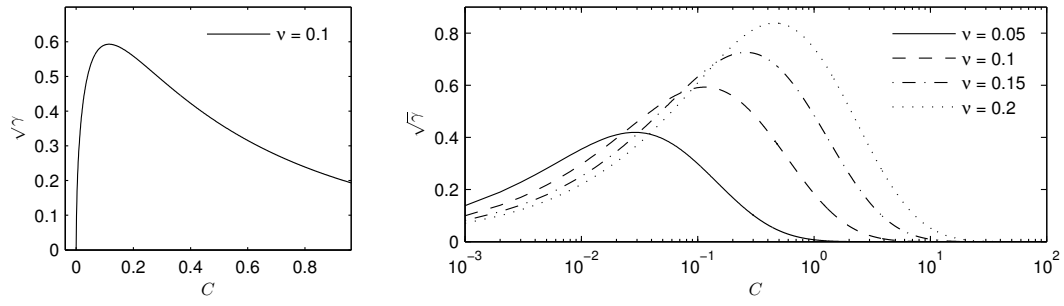
Now, we derive a bifurcation analysis based on this expression. This means that the different solutions which can be obtained by varying C and ν are considered.

The equation (4.1) depends on both C and ν. It is already proven that $\nu \to 0$ gives a small γ. But for C; small γ can occur both if C is small and large. We want to study the possible C values and their corresponding solutions. First, rewrite the expressions a bit. Take the square root of (4.1), then

$$\sqrt{\frac{C}{\nu}}\text{sech}\left(\frac{\sqrt{2C}}{4\nu}\right) = \sqrt{\gamma}. \tag{4.2}$$

Since the sech function in (4.2) is smooth and even, it is possible for the constant C to have two solutions for each $\sqrt{\gamma}$. Hence, if γ is small, then either C is very small or very big. Consider the graphs of (4.2) in Figure 4.1. Define C^* as the C which gives the

extreme value of (4.2). As can be observed – when ν is getting smaller, the C^* is also getting smaller. Hence if γ is fixed, both solutions; call them $C^<$ (for $C < C^*$) and $C^>$ (for $C > C^*$), are getting smaller. In particular, for every γ, there are either zero solutions (above maxima), one solution (at C^*) and two solutions (below maxima). Let us compute



(a) (b)

Figure 4.1: (a) The shape of (4.2) for $\nu = 0.1$ in a linear plot. (b) The equation (4.2) plotted in a semilog plot for varying ν.

the maxima of this second order expression and observe for what γ, there are solutions. Multiply (4.2) on both sides with $\sqrt{2}/(4\sqrt{\nu})$ and obtain

$$\frac{\sqrt{2C}}{4\nu}\operatorname{sech}\left(\frac{\sqrt{2C}}{4\nu}\right) = \sqrt{\frac{\gamma}{8\nu}}. \tag{4.3}$$

Now, use the variable substitution $R := \sqrt{2C}/(4\nu)$. Then,

$$F(R) := \sqrt{8\nu}R\,\operatorname{sech}(R) = \sqrt{\gamma}. \tag{4.4}$$

The extrema of $F(R)$ is computed by differentiating $F(R)$ with right to R and finding the root. Thus,

$$\frac{\mathrm{d}F}{\mathrm{d}R} = \sqrt{8\nu}\,\operatorname{sech}(R)(1 - R\tanh(R)). \tag{4.5}$$

Then letting $\mathrm{d}F/\mathrm{d}R = 0$ gives an expression only depending on R

$$0 = \operatorname{sech}(R)(1 - R\tanh(R)). \tag{4.6}$$

The roots of this expression are $R^* = \pm 1.1997$. But only the positive root is defined for (4.5). It is easy to see that (4.5) has a negative second derivative at 1.1997, which implies that the extreme is a maximum (which can also be seen in Figure 4.1). Substituting back to C gives,

$$C^* = 8\nu^2 1.1997^2. \tag{4.7}$$

Furthermore, the maximum value that $\sqrt{\gamma}$ can possess becomes

$$\max_R F(R) = F(R^*)\sqrt{8\nu} \approx 0.663\sqrt{8\nu} \geq \sqrt{\gamma}. \tag{4.8}$$

Which means that

$$0 < \sqrt{\gamma} \leq 0.663\sqrt{8\nu}. \tag{4.9}$$

Thus, for small γ, there are two different steady states, one corresponding to all $C^<$ and one corresponding to all $C^>$:

$$h^<(x) = \sqrt{2C^<}\tanh\left(\frac{\sqrt{2C^<}}{2\nu}\left(\frac{1}{2} - x\right)\right), \tag{4.10}$$

$$h^>(x) = \sqrt{2C^>}\tanh\left(\frac{\sqrt{2C^>}}{2\nu}\left(\frac{1}{2} - x\right)\right). \tag{4.11}$$

This is the two cases that we may expect the numerical methods converge to. The existence of solutions for γ is restricted by the upper bound computed in equation (4.8). For $\nu = 0.1$ this bound is computed to ≈ 0.593 and for $\nu = 0.01$ we get ≈ 0.188. These are big values in comparison to the round off errors that we expect generates the γ. We expect γ to be close to machine epsilon; but to get equality in (4.9) we can conclude that $\nu < \gamma$, which means that ν must be smaller than machine epsilon in both double precision and single precision for a solution to be non-existing.

4.2 Stability analysis

Studying the eigenvalues of the linearized system give rise to more valuable information about the stability and instability of possible equilibrium. We now know that the equilibrium is either a constant solution or in the form $h^<$ or $h^>$. An eigenvalue analysis therefore focuses on the latter solutions, corresponding to the linearizations around these points.

4.2.1 Linearization

Similarly to the work in section 3.1, the linearization idea is to make use of a first order Taylor approximation. Hence, substitute $u(x,t) := h(x) + \delta\xi(x,t)$ into (1.6) and keep the first order terms. To get control over this procedure one may evaluate the derivatives of (1.6) separately:

$$u_t \approx \xi_t \tag{4.12}$$

$$\left(\frac{u^2}{2}\right)_x = \left(\frac{h^2 + 2h\delta\xi + \delta^2\xi^2}{2}\right)_x \approx (h\xi)_x \tag{4.13}$$

$$\nu u_{xx} \approx \nu\delta\xi_{xx} \tag{4.14}$$

Substituting the boundary conditions yields

$$u_x(0) = h_x(0) + (\delta\xi(0,t))_x = 0 \implies \xi_x(0,t) = 0 \tag{4.15}$$

$$u_x(1) = h_x(1) + (\delta\xi(1,t))_x = 0 \implies \xi_x(1,t) = 0. \tag{4.16}$$

All together with only first order terms kept gives the eigenvalue problem

$$\begin{cases} L\xi := \xi_t = \nu\xi_{xx} - (h\xi)_x = \lambda\xi, & (x,t) \in (0,1) \times [0,\infty),\ \nu > 0,\ \lambda \in \mathbb{C}, \\ \xi_x = 0, & (x,t) \in \{0,1\} \times [0,\infty). \end{cases} \tag{4.17}$$

For simplicity reasons we add a Direchlet condition $\xi(1/2,t) = 0$ and change the domain from $[0,1]$ to $[0,1/2]$. This is possible since we only use odd initial conditions around $x = 1/2$, and the reason we are doing that is that it makes the computations a bit easier further on. Hence, the eigenvalue problem to be considered is

$$\begin{cases} L\xi := \xi_t = \nu\xi_{xx} - (h\xi)_x = \lambda\xi, & (x,t) \in (0,1/2) \times [0,\infty),\ \nu > 0,\ \lambda \in \mathbb{C}, \\ \xi_x = 0, & (x,t) \in \{0,1/2\} \times [0,\infty). \end{cases} \tag{4.18}$$

4.2.2 Transformation into Sturm Liouville form

A problem with the eigenvalue problem on the form (4.18) is that the operator $L\xi = \nu\xi_{xx} - (h\xi)_x$ is not self adjoint. I.e. letting $u, v \in [0,1]$, the inner product $\langle Lu, v\rangle$ is not satisfying $\langle Lu, v\rangle = \langle u, Lv\rangle$. But, transforming into Sturm Liouville form helps to get rid of that problem, since it is a well known fact that the Sturm Liouville operator is self adjoint.

A consequence of a self adjoint operator is that all eigenvalues are real and distinct, which means that it suffices to study the smallest eigenvalue and check if this is positive or negative to get insight in the stability properties.

The transformation process is as follows: Introduce a Liouville transformation

$$\begin{aligned} \eta = \exp\left(\frac{1}{2\nu}\int_{1/2}^{x} h(s)\,\mathrm{d}s\right) &= \exp\left(\frac{1}{2\nu}\int_{1/2}^{x} \sqrt{2C}\tanh\left(\frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2}-s\right)\right)\,\mathrm{d}s\right), \\ &\left\{u = \frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2}-x\right), \quad \mathrm{d}s = -\frac{2\nu}{\sqrt{2C}}\,\mathrm{d}u\right\}, \\ &= \exp\left(\int_0^u \tanh(w)\,\mathrm{d}w\right), \\ &= \exp\left(\ln\left(\mathrm{sech}(u)\right)\right), \\ &= \mathrm{sech}\left(\frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2}-x\right)\right), \end{aligned} \tag{4.19}$$

and do the variable substitution $\xi = \eta\psi$. The derivatives in (4.18) are then computed as

$$\xi_x = \eta_x\psi + \eta\psi_x, \tag{4.20}$$

$$\xi_{xx} = \eta_{xx}\psi + 2\eta_x\psi_x + \eta\psi_{xx}. \tag{4.21}$$

And the derivatives of η are computed as

$$\eta_x = \frac{h\eta}{2\nu}, \tag{4.22}$$

$$\eta_{xx} = \frac{h_x\eta + h\eta_x}{2\nu} = \frac{h_x\eta}{2\nu} + \frac{h^2\eta}{4\nu^2}. \tag{4.23}$$

Substituting (4.22) and (4.23) into (4.20)-(4.21), as such are inserted into (4.18) yields

$$\begin{aligned}
& \lambda\eta\psi = \nu\left(\left(\frac{h_x\eta}{2\nu} + \frac{h^2}{4\nu}\right)\psi + 2\frac{1}{2\nu}h\eta\psi_x + \eta\psi_{xx}\right) - h_x\eta\psi - h^2\frac{1}{2\nu}\eta\psi - h\eta\psi_x \\
\iff \quad & \lambda\psi = -\frac{h_x\psi}{2} - \frac{h^2\psi}{4\nu} + \nu\psi_{xx} \\
\iff \quad & \frac{\lambda}{\nu}\psi = \psi_{xx} - \left(\frac{h_x}{2\nu} + \frac{h^2}{4\nu^2}\right)\psi
\end{aligned}$$

Hence, the eigenvalue problem on Sturm Liouville form is written as

$$\psi_{xx} - q(x)\psi = \frac{\lambda}{\nu}\psi, \tag{4.24}$$

where $q(x)$ is

$$\begin{aligned}
q(x) &:= \frac{h^2}{4\nu^2} + \frac{h_x}{2\nu} \\
&\quad \left\{-\nu h_x + \frac{h^2}{2} = C\right\} \\
&= \frac{C}{2\nu^2}\left(1 - 2\,\mathrm{sech}^2\left(\frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2} - x\right)\right)\right).
\end{aligned} \tag{4.25}$$

The boundary conditions of (4.18) are also transformed in the similar way:

$$\begin{aligned}
& 0 = (\eta(0)\psi(0))_x = \eta_x(0)\psi(0) + \eta(0)\psi_x(0) \\
& \quad = \frac{h(0)}{2\nu}\eta(0)\psi(0) + \eta(0)\psi_x(0) \\
\iff \quad & 0 = \frac{h(0)}{2\nu}\psi(0) + \psi_x(0) \\
& \quad = \psi'(0) + \frac{\sqrt{2C}}{2\nu}\tanh\left(\frac{\sqrt{2C}}{4\nu}\right)\psi(0).
\end{aligned} \tag{4.26}$$

$$\begin{aligned} & 0 = \eta\left(\frac{1}{2}\right)\psi\left(\frac{1}{2}\right) \\ \iff \quad & 0 = \psi\left(\frac{1}{2}\right). \end{aligned} \tag{4.27}$$

Finally, all together, the Sturm-Liouville problem can be stated in the form

$$\begin{cases} \psi_{xx} - q(x)\psi = \frac{\lambda}{\nu}\psi, & x \in (0,1),\ \nu > 0,\ \lambda \in \mathbb{R}, \\ \psi_x(0) + \frac{\sqrt{2C}}{2\nu}\tanh\left(\frac{\sqrt{2C}}{4\nu}\right)\psi(0) = 0, & C \in \mathbb{R}^+, \\ \psi\left(\frac{1}{2}\right) = 0. \end{cases} \tag{4.28}$$

4.2.3 Eigenvalue approximation

The eigenvalue problem can be solved in many ways. We used a second order finite difference approximation, with a N_x point space discretization: $x = i\Delta x$, where $i = 1, 2, \ldots, \mathbb{N}_x$, $\Delta x = 1/N_x$. The discretization in time is written as: For the inner points

$$\frac{\psi_{i-1} - 2\psi_i + \psi_{i+1}}{(\Delta x)^2} - q(x_i)\psi_i = \frac{\lambda}{\nu}\psi_i, \quad i = 2, 3, \ldots, N_x - 1. \tag{4.29}$$

And for the boundary points we used a second order discretization of the derivative at $\psi(1)$: $(\psi(2) - \psi(0))/2\Delta x$, which was inserted into (4.26). For $\psi(1/2)$, (4.27) was directly used. Hence,

$$\begin{aligned} & \frac{\psi_2 - \psi_0}{2\Delta x} + \frac{\sqrt{2C}}{2\nu}\tanh\left(\frac{\sqrt{2C}}{4\nu}\right)\psi_1 = 0 \\ \implies & \psi_0 = \frac{\Delta x\sqrt{2C}}{\nu}\tanh\left(\frac{\sqrt{2C}}{4\nu}\right)\psi_1 + \psi_2 \end{aligned} \tag{4.30}$$

$$\psi_{N_x+1} = 0 \tag{4.31}$$

Which gives

$$\frac{\frac{\Delta x\sqrt{2C}}{\nu}\tanh\left(\frac{\sqrt{2C}}{4\nu}\right)\psi_1 + 2\psi_2}{(\Delta x)^2} - q(x_1)\psi_1 = \frac{\lambda}{\nu}\psi_1, \tag{4.32}$$

$$\frac{-2\psi_{N_x} + \psi_{N_x-1}}{(\Delta x)^2} - q(x_{N_x})\psi_{N_x} = \frac{\lambda}{\nu}\psi_{N_x}. \tag{4.33}$$

Written in matrix form the finite difference approximation yields

$$A\psi = \frac{\lambda}{\nu}\psi, \tag{4.34}$$

where

$$A = \frac{1}{(\Delta x)^2}\begin{bmatrix} \frac{\Delta x\sqrt{2C}}{\nu}\tanh\left(\frac{\sqrt{2C}}{4\nu}\right) & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} - \operatorname{diag}\left(q(x_1), q(x_2), \ldots, q(x_{N_x})\right). \tag{4.35}$$

The smallest eigenvalue of A can be computed by e.g. inverse power iteration. In Matlab there is the inbuilt `eig` function, to get the eigenvalues too. We do not care that the eigenvalues indeed are divided by the ν constant. The result is scaled, which doesn't effect the signs of the eigenvalues. Solving for different C, we can see that the eigenvalues are all smaller than zero if $C < C^*$, one eigenvalue equal to zero and all other negative for $C = C^*$, and one eigenvalue positive and the rest negative for $C > C^*$. In Figure 4.2, consider how the first scaled eigenvalue ($\hat{\lambda} = \lambda/\nu$) is changing for different sizes of $R = \sqrt{2C}/(4\nu)$. This variable does not care about ν, as ν is fixed. Therefore it gives a general solution for the C constant. In the plot we can see that the general value for the extrema $R^* = 1.1997$, occur when the first eigenvalue changes from being negative to being positive, for a varying R with fixed ν.

We can conclude that by the numerical approximation of the eigenvalues, the equilibria corresponding to $R < R^*$ is stable, R^* belongs to the centre manifold (eigenvalues equal to zero) and for $R > R^*$ there are unstable equilibras.

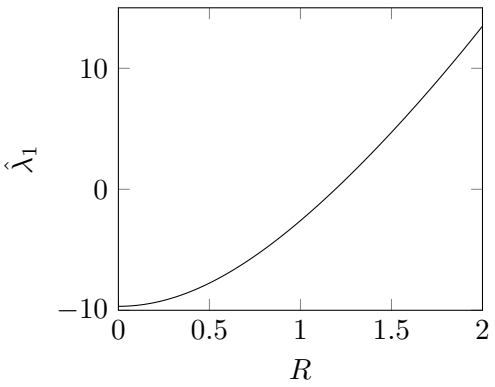


Figure 4.2: The first eigenvalue of the eigenvalue problem for varying R and a fixed $\nu = 0.1$.

Chapter 5

Numerical results

Analytically we have deduced a lot of facts according to the steady state solutions in finite precision arithmetic. Now it is up to showing that these results actually occur. We give in this chapter some examples where the finite precision approximations perform poorly. Aside from stability questions, which obviously differ, different methods based on different discretization approximations give rise to different problems. We have implemented three different numerical schemes, all of different characters: One explicit finite differences method, the standard first order explicit Euler method; One implicit second order finite differences method, the Crank Nicolson method; and a piecewise finite element method solved in time with the inbuilt adaptive Matlab function `ode15s`. The implementation of the three methods are given in the appendix. The basic idea of using just these three methods was the difference in the implementation of the boundary conditions. The explicit Euler method uses a first order discretization of the boundary conditions, the Crank Nicolson uses a second order discretization and for the finite element implementation the Neumann conditions are incorporated implicitly. In all comparing experiments antisymmetric initial conditions are used because of the knowledge that they shall converge to the zero steady state.

Before actually showing results, there is a need of proving existence of possible solutions to the discretized problems. It follows that the results are not the same for all three methods studied.

5.1 Existence of numerical solutions

We have proved that the numerical methods may converge to wrong, non-existing steady states of (1.6), and the hypothesis is that these occur because of finite precision approximations. There is of interest before doing the numerical test to know if those wrong solutions exist for the numerical schemes implemented. From previous results, we expect the steady state solutions to the viscid Burgers' equation with homogeneous Neumann

conditions and odd initial conditions be in the form

$$h(x) = \sqrt{2C}\tanh\left(\frac{\sqrt{2C}}{2\nu}\left(\frac{1}{2} - x\right)\right), \tag{5.1}$$

where the viscosity parameter ν is fixed, and the constant C is zero for the only correct solution and non-zero for wrong ones.

5.1.1 Explicit Euler method

The explicit Euler method is a finite differences approximation that is discretized forward in both time and space. The implementation for the viscid Burgers' equation is written as (the whole discretization process can be found in appendix.)

$$u_1^{n+1} = u_1^n + \nu\Delta t\left(\frac{-u_1^n + u_2^n}{(\Delta x)^2}\right), \tag{5.2}$$

$$u_i^{n+1} = u_i^n + \Delta t\left(\frac{f(u_i^n) - f(u_{i+1}^n)}{\Delta x} + \nu\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{(\Delta x)^2}\right), \tag{5.3}$$

$$u_{N_x}^{n+1} = u_{N_x}^n + \nu\Delta t\left(\frac{-u_{N_x}^n + u_{N_x-1}^n}{(\Delta x)^2}\right). \tag{5.4}$$

where $f(u) = u^2/2$ and $i = 1, 2, \ldots, N_x$, $\Delta x = 1/(N_x - 1)$.

Theorem 5.1.1. *For the explicit Euler implementation to the viscid Burgers' equation with homogeneous Neumann conditions (stated in (5.2)-(5.4)) with an arbitrary initial condition $u_0 \in L^2(0,1)$, all steady state solutions must be constants.*

Proof. At a steady state we have $u^{n+1} = u^n$. If we start from the left boundary, we prove the theorem by induction. Let $i = 1$, then at a steady state we get $u_1 = u_2$ from (5.2). Substituting into (5.3) yields

$$\begin{aligned} u_1 &= u_1 + \Delta t\left(\frac{u_1^2 - u_1^2}{2\Delta x} + \nu\frac{u_2 - 2u_2 + u_3}{(\Delta x)^2}\right) \\ \iff 0 &= \nu\Delta t\left(\frac{u_3 - u_2}{(\Delta x)^2}\right). \end{aligned}$$

This means that $u_3 = u_2$. Assuming $i - 1 = i$ gives similarly

$$\begin{aligned} u_{i-1} &= u_{i-1} + \Delta t\left(\frac{u_{i-1}^2 - u_{i-1}^2}{2\Delta x} + \nu\frac{u_i - 2u_i + u_{i+1}}{(\Delta x)^2}\right) \\ \iff 0 &= \nu\Delta t\left(\frac{u_{i+1} - u_i}{(\Delta x)^2}\right). \end{aligned}$$

Hence, $u_{i+1} = u_i$, which proves that all points must be equal. If all points are equal, also the right boundary condition must be satisfied, since letting $i = n - 1$ implies that (5.4) is fulfilled as $u^{n+1} = u^n$. I.e all steady state solutions computed by the eplicit Euler

implementation must be constants.

□

5.1.2 Crank Nicolson method

The Crank Nicolson method is a second order implicit method, which is discretized forward in time and centred in space. For the viscid Burgers' equation the implementation yields (see appendix for details)

$$\frac{u_1^{(n+1)}-u_1^{(n)}}{\Delta t}=\nu\left(\frac{-u_1^{(n)}+u_2^{(n)}}{(\Delta x)^2}+\frac{-u_1^{(n+1)}+u_2^{(n+1)}}{(\Delta x)^2}\right), \tag{5.5}$$

$$\begin{aligned}\frac{u_i^{(n+1)}-u_i^{(n)}}{\Delta t}+\frac{f(u_{i+1}^{(n)})-f(u_{i-1}^{(n)})}{2\Delta x}+\frac{f(u_{i+1}^{(n+1)})-(fu_{i-1}^{(n+1)})}{2\Delta x}\\ =\nu\left(\frac{u_{i+1}^{(n)}-2u_i^{(n)}+u_{i-1}^{(n)}}{2(\Delta x)^2}+\frac{u_{i+1}^{(n+1)}-2u_i^{(n+1)}+u_{i-1}^{(n+1)}}{2(\Delta x)^2}\right),\end{aligned} \tag{5.6}$$

$$\frac{u_{N_x}^{(n+1)}-u_{N_x}^{(n)}}{\Delta t}=\nu\left(\frac{-u_{N_x}^{(n)}+u_{N_x-1}^{(n)}}{(\Delta x)^2}+\frac{-u_{N_x}^{(n+1)}+u_{N_x-1}^{(n+1)}}{(\Delta x)^2}\right). \tag{5.7}$$

where $f(u)=u^2/2$ and $i=1,2,\ldots,N_x$, $\Delta x=1/(N_x-1)$.

Theorem 5.1.2. *For the Crank Nicolson implementation to the viscid Burgers' equation (stated in (5.5)-(5.7)) with an initial condition $u_0\in L^2_{odd}(0,1)$, there exist a zero steady state solution and a steady state solution on the form*

$$u=\begin{cases}\frac{2\nu}{\Delta x} & 0\le i\le\frac{N_x-1}{2}\\ 0 & i=\frac{N_x-1}{2}+1\\ -\frac{2\nu}{\Delta x} & \frac{N_x-1}{2}+2\le i\le N_x\end{cases}, \tag{5.8}$$

where N_x is an odd integer.

Proof. The zero solution is directly fulfilled for (5.5)-(5.7), since all terms are non-constants. For a steady state, $n=*$, we can assume that $u^n=u^{n+1}$. Hence the inner point expression (5.6) is reformed to

$$\left(u_{i+1}^*\right)^2-\left(u_{i-1}^*\right)^2-\frac{2\nu}{\Delta x}\left(u_{i+1}^*-2u_i^*+u_{i-1}^*\right)=0. \tag{5.9}$$

There are actually four cases to study:

Case 1: Let $u_{i-1}=u_i=u_{i+1}=C$ for $i<(N_x-1)/2$ and $u_{i-1}=u_i=u_{i+1}=D$ for $i>(N_x-1)/2+2$, then everything on both left hand side and right hand side cancels out. Thus, if all three points in the stencil are equal, we have a solution. This case also covers the boundary conditions, which follows from the assumption.

Case 2: Let $u_{(N_x-1)/2-1} = u_{(N_x-1)/2} = C$ and $u_{(N_x-1)/2+1} = 0$, then

$$\begin{aligned} \left(-C^2\right) - \frac{2\nu}{\Delta x}(-C) &= 0 \\ \iff C &= \frac{2\nu}{\Delta x}. \end{aligned}$$

Case 3: Let $u_{(N_x-1)/2+1} = 0$, and $u_{(N_x-1)/2+2} = u_{(N_x-1)/2+3} = D$, then

$$\begin{aligned} \left(D^2\right) - \frac{2\nu}{\Delta x}(-D) &= 0 \\ \iff D &= -\frac{2\nu}{\Delta x}. \end{aligned}$$

Case 4: Let $u_{(N_x-1)/2} = C$, $u_{(N_x-1)/2+1} = 0$ and $u_{(N_x-1)/2+2} = D$, then

$$\begin{aligned} \left(D^2 - C^2\right) - \frac{2\nu}{\Delta x}(D + C) &= 0 \\ \iff \left(D^2 - C^2\right) &= \frac{2\nu}{\Delta x}(D + C). \end{aligned}$$

By plugging in the results from case 2 and case 3 into case 4, both left hand side and right hand side of the last expression cancels out. Hence, we have another solution. Summarize all cases gives the final result:

$$u = \begin{cases} \frac{2\nu}{\Delta x} & 0 \leq i \leq \frac{(N_x-1)}{2}, \\ 0 & i = \frac{(N_x-1)}{2} + 1, \\ -\frac{2\nu}{\Delta x} & \frac{(N_x-1)}{2} + 2 \leq i \leq N_x, \end{cases} \tag{5.10}$$

where N_x is an odd integer. □

5.1.3 Finite element method

In matrix form, the finite element implementation is written as (see appendix for the implementation)

$$\dot{\xi} = M^{-1}(\nu K\xi - B(\xi \circ \xi)). \tag{5.11}$$

where M, K and B are all $(n+1) \times (n+1)$ stiffness-matrices, ξ is the approximate $(n+1) \times 1$ Galerkin solution vector and $\xi \circ \xi := \left(\xi_0^2, \xi_1^2, \ldots, \xi_{n+1}^2\right)^\top$. For a steady state, the time derivative is zero. Hence, at the steady state we have,

$$0 = M^{-1}(\nu K\xi - B(\xi \circ \xi)), \tag{5.12}$$

which can be simplified to

$$\nu K\xi - B(\xi \circ \xi) = 0. \tag{5.13}$$

where the form of the matrices can be found in the appendix, stated in equations (8.51)-(8.53). It follows from this system that the inner points of (5.13) are formulated as

$$(\xi_{i+1})^2 - (\xi_{i-1})^2 - \frac{4\nu}{\Delta x}(\xi_{i-1} - 2\xi_i + \xi_{i+1}) = 0, \tag{5.14}$$

where $i = 1, 2, \ldots, n$. And the expressions for the boundary points are

$$(\xi_1)^2 - (\xi_0)^2 - \frac{4\nu}{\Delta x}(\xi_1 - \xi_0) = 0 \tag{5.15}$$

$$(\xi_{n+1})^2 - (\xi_n)^2 - \frac{4\nu}{\Delta x}(\xi_{n+1} - \xi_n) = 0 \tag{5.16}$$

This gives the following result:

Theorem 5.1.3. *For the finite element implementation to the viscid Burgers' equation (stated in (5.11)) with an initial condition $u_0 \in L^2_{odd}(0,1)$, there exist a zero steady state solution and a steady state solution on the form*

$$u = \begin{cases} \frac{4\nu}{\Delta x} & 0 \leq i \leq \frac{n}{2}, \\ 0 & i = \frac{n}{2} + 1, \\ -\frac{4\nu}{\Delta x} & \frac{n}{2} + 2 \leq i \leq n + 1, \end{cases} \tag{5.17}$$

where n is an odd integer.

Proof. The main proof follows exactly the same steps as for the Crank Nicolson implementation. Note the similarity of the expression (5.9) against (5.14). For the boundary points it is easy to see that letting $\xi_1 = \xi_0$ and $\xi_{n+1} = \xi_n$ solves (5.15) and (5.16), which is sufficient to prove the result. □

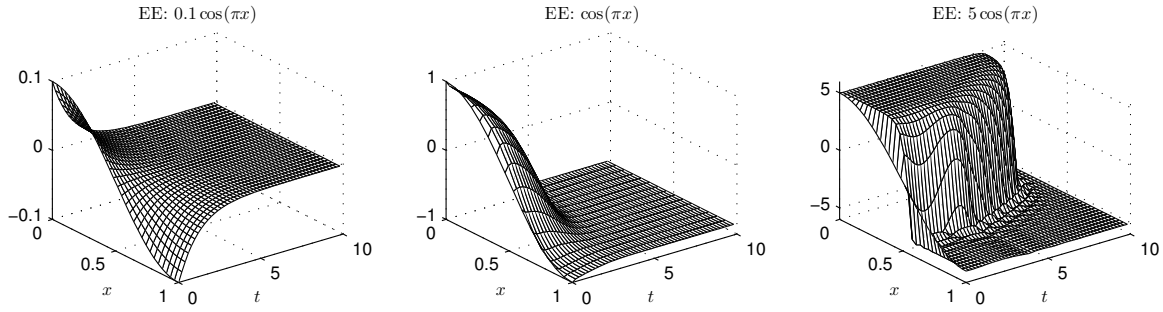
5.2 Results

In this section, the numerical results in form of plots and tables are presented. The section is divided into treating different interesting cases for the different methods implemented. Note that it is hard to trust numerical proofs. The interpretation of the numerical results must be taken with care. Therefore the main goal of this section is to review the main problems that can occur and show counter-examples. More detailed studies as exact proofs of why the numerical results occur are left for further studies.

5.2.1 ν fixed, varying magnitude of the initial condition

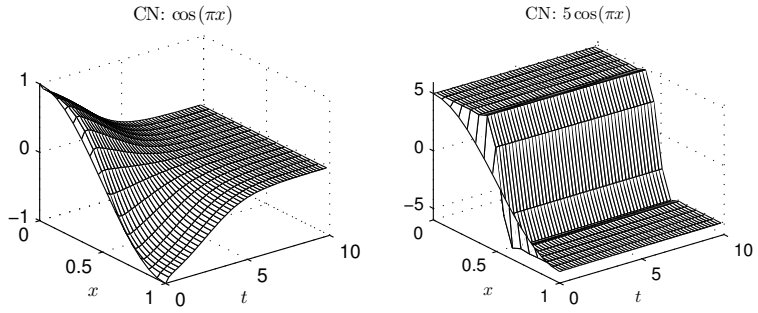
First, fix the viscosity parameter to $\nu = 0.1$ and consider the impact of different initial conditions with varying magnitudes.

Consider the results we get for the $D\cos(\pi x)$ condition for varying D settings. See Figure 5.1 for the result computed by the explicit Euler method, 5.2 for the result computed with Crank Nicolson method and 5.3 for the result computed with the finite elements in space and Matlab `ode15s` in time. The results for the Crank Nicolson and the finite element implementations gives more or less identical results as can be seen in the plots.



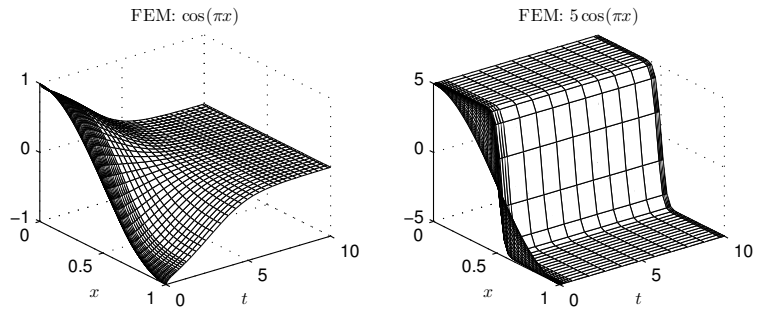
(a) (b) (c)

Figure 5.1: Explicit Euler results.



(a) (b)

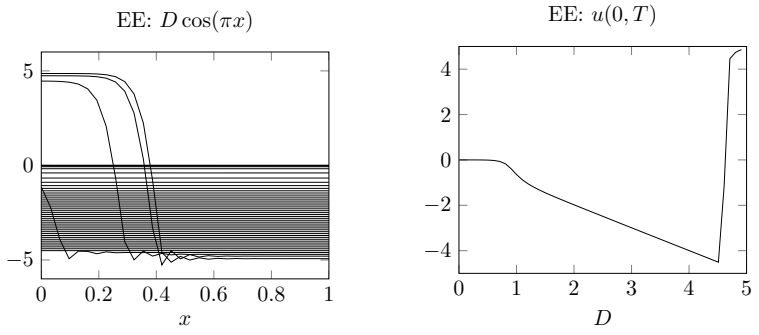
Figure 5.2: Crank Nicolson results.



(a) (b)

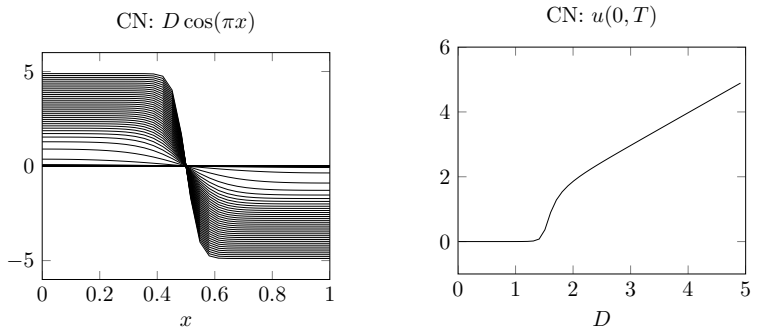
Figure 5.3: Finite element results.

From the plots we can see that the solutions for the initial condition $D\cos(\pi x)$, where D is small are all converging to solutions close to zero, and for the case when $D = 5$ the steady state is non-constant for the Crank Nicolson and the FEM implementations. Even for the explicit Euler the solution for large D looks as something close to the tanh solution even though this solution does not exist, which was proved earlier in this chapter. If computing the solution for all $D \in [0.01, 5]$ one can see approximately where the solution starts switching from being a non-constant solution to being the zero solution. In plots 5.4-5.6, it is noticeable that the explicit Euler is more sensitive than the other two methods (it is just a first order method, so we cannot make any conclusions about this more than state it at a notice. The other methods should converge faster because the order of the discretization is higher). The explicit Euler method converges to zero for all D approximately smaller than 0.75 and the Crank Nicolson and FEM needs D smaller than approximately 1.5 to converge to a solution close to the zero function.



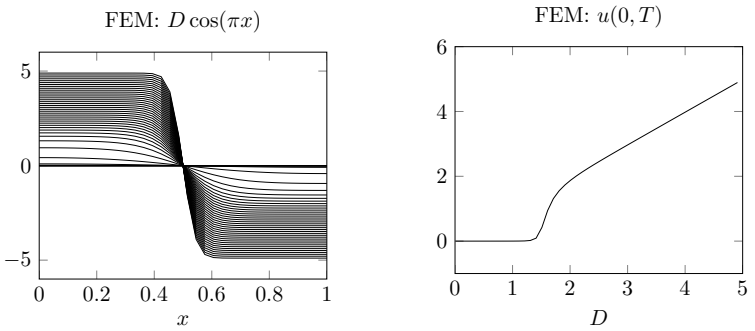
(a) (b)

Figure 5.4: Explicit Euler solutions for $D \in [0.01, 5]$ at $T = 10$ with $\nu = 0.1$.



(a) (b)

Figure 5.5: Crank Nicolson solutions for $D \in [0.01, 5]$ at $T = 10$ with $\nu = 0.1$.

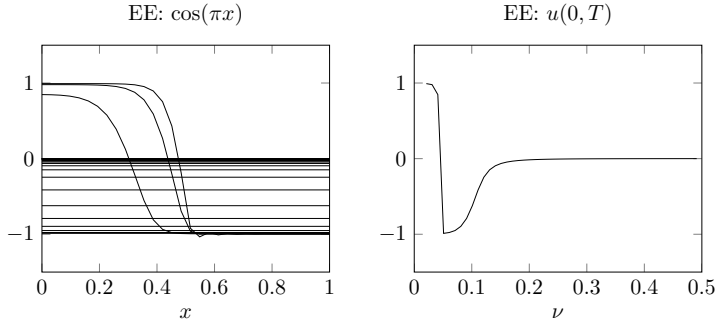


(a) (b)

Figure 5.6: Finite elements solutions for $D \in [0.01, 5]$ at $T = 10$ with $\nu = 0.1$.

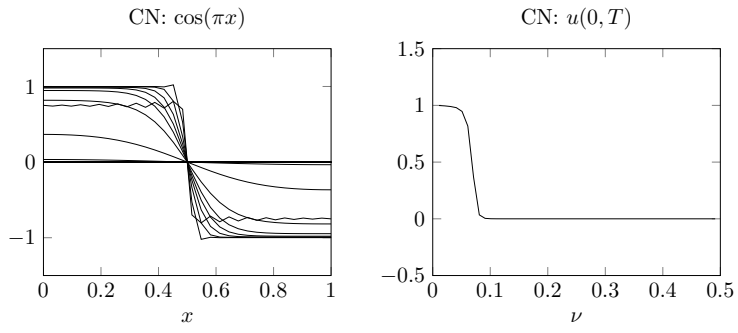
5.2.2 Magnitude of the initial condition fixed, varying ν

As ν is getting small, the steady state solutions are influenced in the same way as if the C constant is getting large. By plotting the final state for a varying ν, we can measure the sensitivity for which the solution tends to zero or not. Now, the D constant is fixed to being one all time. As one can see in Figure 5.7 the explicit Euler needs ν larger than approximately 0.15 to converge to zero for the initial condition $\cos(\pi x)$. But similarly as the case where ν was fixed and D was changing, the zero solution is not as sensible as for the Crank Nicolson and FEM implementations (compare to Figures 5.8 and 5.9). The oscillations as can be seen in the plot of the Crank Nicolson (Figure 5.8) are spurious oscillations that occur for the Crank Nicolson method instead of a blown-up solutions, when the method has stability problems. The method is unconditionally numerically stable and this is other issues that indeed can occur. For the finite element implementation, we used the inbuilt Matlab function `ode15s`, which is adaptive and shows more numerically stable results.



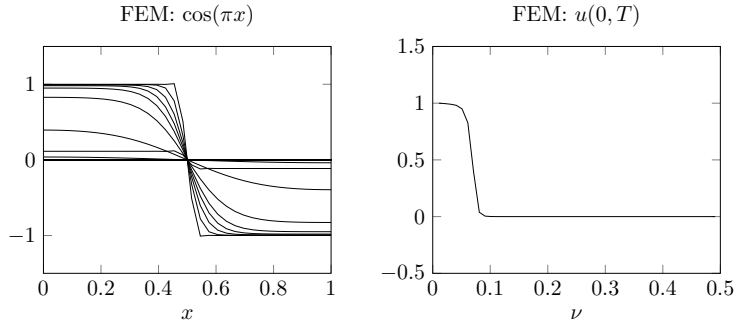
(a) (b)

Figure 5.7: Explicit Euler solutions at $T = 10$ with $D = 1$ and $\nu \in [0.01, 0.5]$.



(a) (b)

Figure 5.8: Crank Nicolson solutions at $T = 10$ with $D = 1$ and $\nu \in [0.01, 0.5]$.



(a) (b)

Figure 5.9: Finite elements solutions at $T = 10$ with $D = 1$ and $\nu \in [0.01, 0.5]$.

5.3 Approximation of roots and derivatives

For the class of odd initial conditions defined in section 3.2, there is always one root, which lies in the middle of the spatial interval $[0, 1]$. Since all solutions are in the class of odd solutions for all iterations, the root shall be invariant in time. We have talked about the zero approximation of the Neumann condition before and that the C constant is computed by

$$C = -\nu h_x(0) + \frac{h(0)^2}{2}. \tag{5.18}$$

Thus, there is actually two things that controls the value of C: the $\nu h_x(0)$ term and the $h(0)^2/2$ term. For an iterative solver, every current state can be seen as an initial condition. And since the solution is the constant mean value of the initial condition, the mean value must be zero for all current states. Hence, if the convergent state is the zero function, then the approximation of the root at the middle of the spatial domain is crucial for every time step. If this approximation is computed wrongly in one time step the error cannot be cured through the rest of the process, since there is no global attractor (which was proven in theorem (3.0.1).

Consider in Table 5.1 how the $x = 0.5$ value is changing as time grows for the case where the solution is close to the zero solution, and consider Table 5.2 for the case where the solution is a non-constant solution. The viscosity term is fixed as $\nu = 0.1$ in both cases. The results are computed with the finite element method, but similar results are observed also for the other two methods. Notice that, if we assume that the finite element

Time	$u((N_x - 1)/2 + 1)$
0	6.1232e-17
1.3722e-03	9.4046e-17
2.7444e-03	-8.3506e-17
4.1166e-03	-3.8248e-16
1.2952e-02	-8.7108e-16
2.1788e-02	-3.8444e-14
3.0623e-02	-1.1726e-13
⋮	⋮
2.6849e+04	-9.1408e-08
3.6849e+04	-9.1408e-08

Table 5.1: Solutions at $x = 1/2$ for different time states, computed from the initial condition $\cos(\pi x)$, by the FEM implementation. The space discretization was chosen as $N_x = 401$. for a fixed $\nu = 0.1$.

Time	$u((N_x - 1)/2 + 1)$
0	3.0616e-16
6.2314e-04	5.8441e-16
1.2463e-03	1.5376e-13
1.8694e-03	3.8122e-13
4.3869e-03	6.5038e-13
6.9044e-03	1.4049e-12
⋮	⋮
1.1604e+00	-4.6776e-08
1.1605e+00	-4.6776e-08

Table 5.2: Solutions at $x = 1/2$ for different time states, computed from the initial condition $5\cos(\pi x)$, by the FEM implementation. The space discretization was chosen as $N_x = 401$. for a fixed $\nu = 0.1$.

approximation approximates the derivatives almost correctly, there is still other errors that influences the choice of the C constant, and therefore makes the iterative process converge to wrong type of solutions.

Consider in Tables 5.3 and 5.4 the approximative derivative at $x = 0$ for the two test cases $\cos(\pi x)$ and $5\cos(\pi x)$. The number of inner points used was the same as for the $x = 1/2$ analysis, namely 401. A second order approximation of the derivative was preformed and the method used was the finite element method. For the case where the initial condition was chosen as $\cos(\pi x)$, the derivative seems stabilized on a zero value at

time 1.87×10^6, as can be seen in Table 5.3 – but then, two time steps later it changes to a non-zero value, and after that going back to zero again. This explains the difficulty of the zero result. On the other hand, the derivatives corresponding to the other initial condition for which the solution has been converging to a non-constant function is stabilized at a small non-zero value, not even close to machine epsilon ($\approx 10^{-16}$), see Table 5.4.

Time	$u(3) - u(1)$
0	-1.2337e-04
1.3722e-03	-1.1055e-04
2.7444e-03	-1.0437e-04
4.1166e-03	-9.9844e-05
1.2952e-02	-8.3906e-05
2.1788e-02	-7.3908e-05
2.1788e-02	-7.3908e-05
3.0623e-02	-6.6745e-05
4.9973e-02	-5.5301e-05
⋮	⋮
1.8760e+06	0
2.6800e+06	0
3.6800e+06	-1.3235e-23
4.6800e+06	1.3235e-23
5.6800e+06	0
6.6800e+06	0

Table 5.3: Second order approximation of the derivative at different times for the finite element approximation. For the initial condition $\cos(\pi x)$.

Time	$u(3) - u(1)$
0	-6.1684e-04
6.2314e-04	-4.3621e-04
1.2463e-03	-3.5537e-04
1.8694e-03	-3.0242e-04
4.3869e-03	-1.6863e-04
6.9044e-03	-1.1600e-04
9.4219e-03	-8.4382e-05
1.1939e-02	-6.2092e-05
⋮	⋮
1.1009e+00	-5.0768e-12
1.1010e+00	-5.0768e-12

Table 5.4: Second order approximation of the derivative at different times for the finite element approximation. For the initial condition $\cos(\pi x)$.

Comparing how the approximated values of the derivatives elapses with time for the two cases shows that the approximations seem to be closely to the same magnitude for the first iterations in both the solution computed with $\cos(\pi x)$ and the solution computed from

$5\cos(\pi x)$. Hence, the derivative seems not be the main case for the solution to converge to wrong solutions. We can note that which steady state solution the method chooses to converge to is probably mostly not depending on the magnitude at the derivatives; and the steady state solution for which the method converges to is decided more or less from the first iteration – especially for the cases where the solution converges to a non-constant solution, which is reached in short time compared to the much longer time that is needed for convergence to the constant solutions.

5.3.1 Wrong solutions because of the approximation at $x = 1/2$

We have shown that the approximation of the root at $x = 1/2$ for initial conditions in $L^2_{odd}(0,1)$ is a hard one for finite precision to handle. Indeed, according to the round off error at this point – for some cases, it shows that the solution makes a drastic change and converges to a wrong constant solution instead of the zero solution that was expected. By imposing a Direchlet condition at the root we get rid of this problem, which makes it even more clear that the round off errors in the point is a source to the occurrence of this problems. Consider in Figure 5.10 two different cases, where the $L^2_{odd}(0,1)$ initial conditions $4\cos(\pi x)$ and the discontinuous: 5 if $0 \leq x < 1/2$, 0 if $x = 1/2$ and -5 if $1/2 < x \leq 1$ were used.

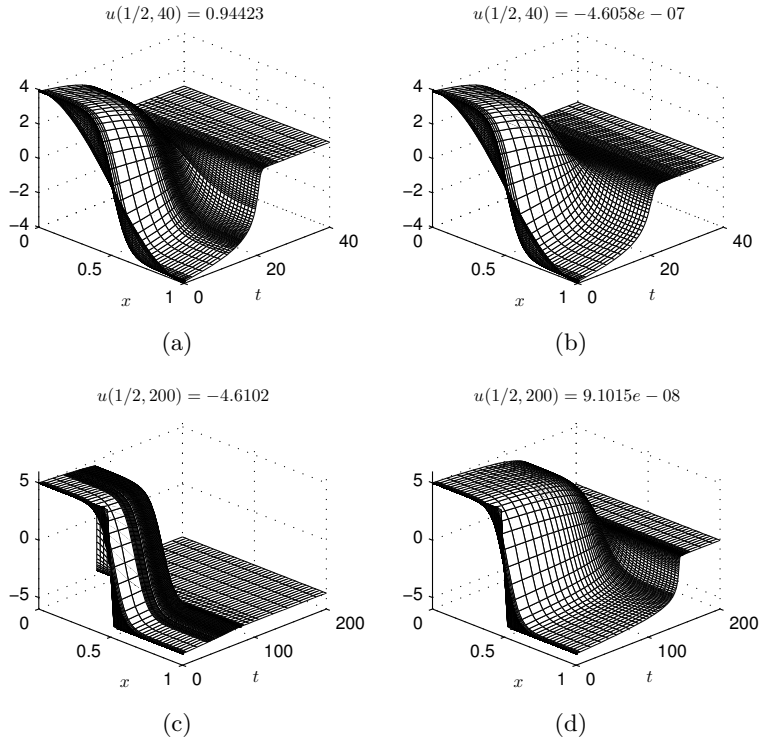


Figure 5.10: (a) and (c): Without condition at $x = 1/2$. (b) and (d): With the added Direchlet condition $u(1/2, t) = 0$.

5.4 Solutions in different decimal formats

Changing the finite precision format to something that can handle less decimals give rise to even more round off errors. Consider in Figure 5.11, a case where the solution converge to the zero function in one format (double precision) and converging to the non-zero function in another format (four significant digits format).

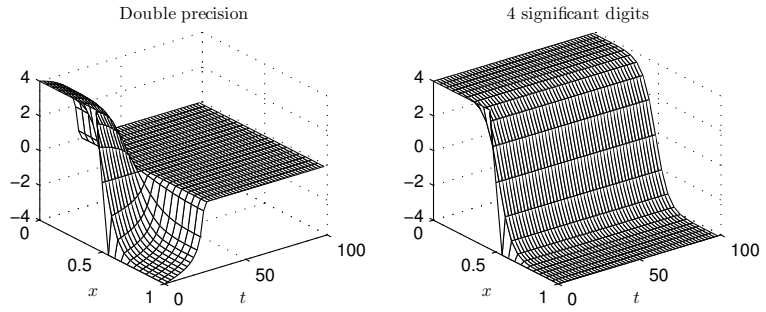
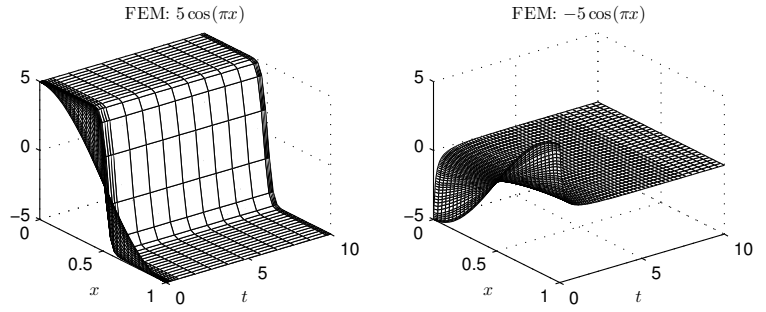


Figure 5.11: Solution in time for double precision and a four digit significant digit approximation for an initial condition with magnitude 4.

5.5 Monotonically increasing odd initial conditions

The results shows that the magnitude of the initial data is a reason by itself for the solution to converge to a non-constant solution. This is indeed not exactly the case. In section 3.3 we deduced the fact that the derivative must be negative for the non-constant solutions to exist. In Figure 5.12, there is an example of this: For the monotonically decreasing initial condition $5\cos(\pi x)$ over the interval $[0, 1]$ the solution converge to a non constant, but for the monotonically decreasing $-5\cos(\pi x)$ over the same interval it does not.



(a) (b)

Figure 5.12: Solutions generated from monotonically decreasing vs monotonically increasing initial condition.

As a second case: Even for a really large amplitude of the initial condition the solution converge to zero. And the convergence is really fast; in approximately $T = 0.001$ the solution is close to zero. Consider this case in Figure 5.13.

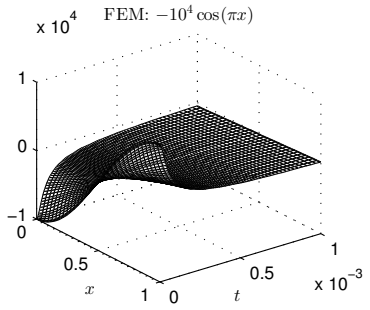


Figure 5.13: Very high magnitude monotonically increasing initial condition used.

5.6 Comparison with the non-homogeneous problem

In finite precision, recall we assume that the approximated problem is written as:

$$\begin{cases} U_t + UU_x = \nu U_{xx}, & (x,t) \in (0,1) \times (0,T],\ \nu > 0, \\ U_x = -\gamma, & (x,t) \in \{0,1\} \times (0,T],\ \gamma > 0, \\ U = U_0, & (x,t) \in [0,1] \times \{0\},\ U_0 \in L^2_{odd}(0,1), \end{cases} \tag{5.19}$$

where U is the discrete solution and T is the final computation time. Let us compare the results we get when non-constant Neumann conditions are imposed instead of the true homogeneous analogue. If our hypotheses are true, the results shall be equal to each other for sufficiently small γ.

Consider the computed C values for different initial conditions at a convergent time in Table 5.5. As we can see; for the case when the initial condition is small and the convergence is close to a zero solution, the homogeneous results are similar to the non-homogeneuos when γ is close to machine epsilon. This means in particular that the $h^<$ solutions may converge to zero by themselves. But for a case with larger initial condition and convergence to a non-constant function (Table 5.6), the C constant in the homogeneous implementation is approximated close to the case where γ is set to 10^{-10} for the non-homogeneuos one. Taking larger values than $\approx 10^{-7}$ gives not relyable solutions, where the value at $x = 1/2$ are not even close to zero.

It seems that the finite element solver has problems with the combination of the non-homogeneuos boundary conditions together with the approximation of the root at $x = 1/2$. An example of that with $\gamma = 10^{-4}$ is shown in Figure 5.14, where the root is completely wrong approximated after the time $T = 1$.

In particular, what can be concluded from the Tables 5.5-5.6 and the Figures 5.14-5.15 is that for small γ, the non-homogeneous and the homogeneous boundary conditions gives

results very close to each other, and for γ close the size of machine epsilon, the solutions are identical.

		γ			
		0	1e-12	1e-14	1e-17
T	0	5.0123e-01	5.0123e-01	5.0123e-01	5.0123e-01
	100	2.0942e-15	1.9452e-15	1.8846e-15	2.0942e-15
	1000	1.7015e-16	4.9020e-18	8.0881e-16	1.7015e-16

Table 5.5: C at different times for the initial condition $\cos(\pi x)$

		γ			
		0	1e-5	1e-7	1e-10
T	0	1.2506e+01	1.2506e+01	1.2506e+01	1.2506e+01
	0.01	1.2428e+01	1.2420e+01	1.2428e+01	1.2428e+01
	1	1.2400e+01	1.1915e+01	1.2395e+01	1.2400e+01

Table 5.6: C at different times for the initial condition $5\cos(\pi x)$

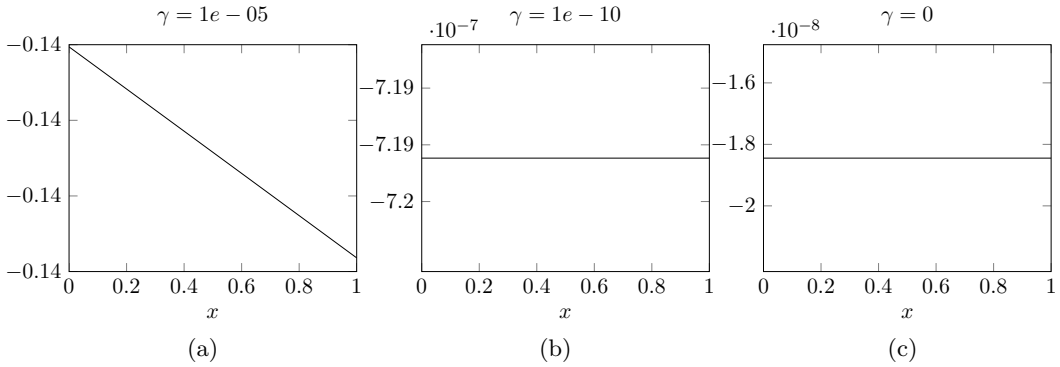


Figure 5.14: Varying γ for the initial condition $\cos(\pi x)$.

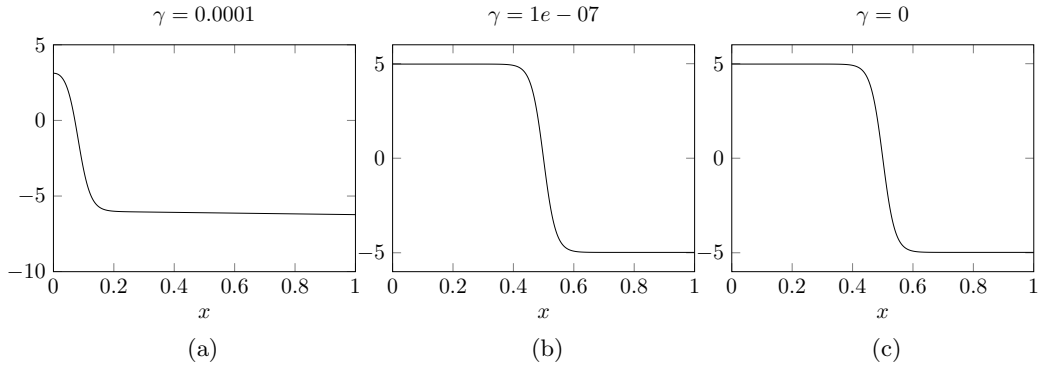


Figure 5.15: Time $T = 1$ for the initial condition $5\cos(\pi x)$.

Chapter 6

Possible treatment on boundary conditions

Besides the surprise that the wrong solutions exists, the main thoughts that appear when dealing with this kind of problems may be how to actually get rid of them, or how to at least make the errors as small as possible.

At this moment, we have fortunately some insight in how the solution should look like. This makes it possible to add feedback to the boundary conditions such that it allows the problem to behave as we wish.

If the ν is assumed to be fixed, then it is the C constant in (3.6) that actually makes the solution behave wrongly. This can be seen in detail in for instance chapter 4. If we set the constant to a value corresponding to the solutions $h^<$ in each iteration, then the solution shall also converge to such a solution. And if we let the constant be equal to zero, we are actually helping the computer choose this constant as it should be chosen to satisfy the true zero Neumann conditions.

Thus, letting the boundary conditions be based on:

$$C = \frac{h(0)^2}{2} - \nu h'(0) = \frac{h(1)^2}{2} - \nu h'(1), \tag{6.1}$$

which is in the from the solution process for a general steady state solution to (1.6). Using this as boundary conditions at $x = 0$ and $x = 1$ makes the new system looking like this

$$\begin{cases} U_t + UU_x = \nu U_{xx}, & (x,t) \in (0,1) \times (0,T],\ \nu > 0 \\ \frac{U^2}{2} - \nu U_x = C & (x,t) \in \{0,1\} \times (0,T],\ C \in \mathbb{R}^+,\ \nu > 0 \\ U = U_0 & (x,t) \in [0,1] \times \{0\},\ U_0 \in L^2_{odd}(0,1) \end{cases}, \tag{6.2}$$

where U is the discrete solution and T is the final computation time. Thus, choosing C close to zero gives rise to a solution corresponding to $h_<$ determined in the bifurcation analysis, or the zero solution. And choosing C large enough gives the $h_>$ solutions. How large and small C must be chosen depends clearly on the choice of ν. In Figure 6.2, we

show different cases, where C is chosen differently, were the initial condition is chosen large enough so it surely makes the original problem converge to a non-constant solution.

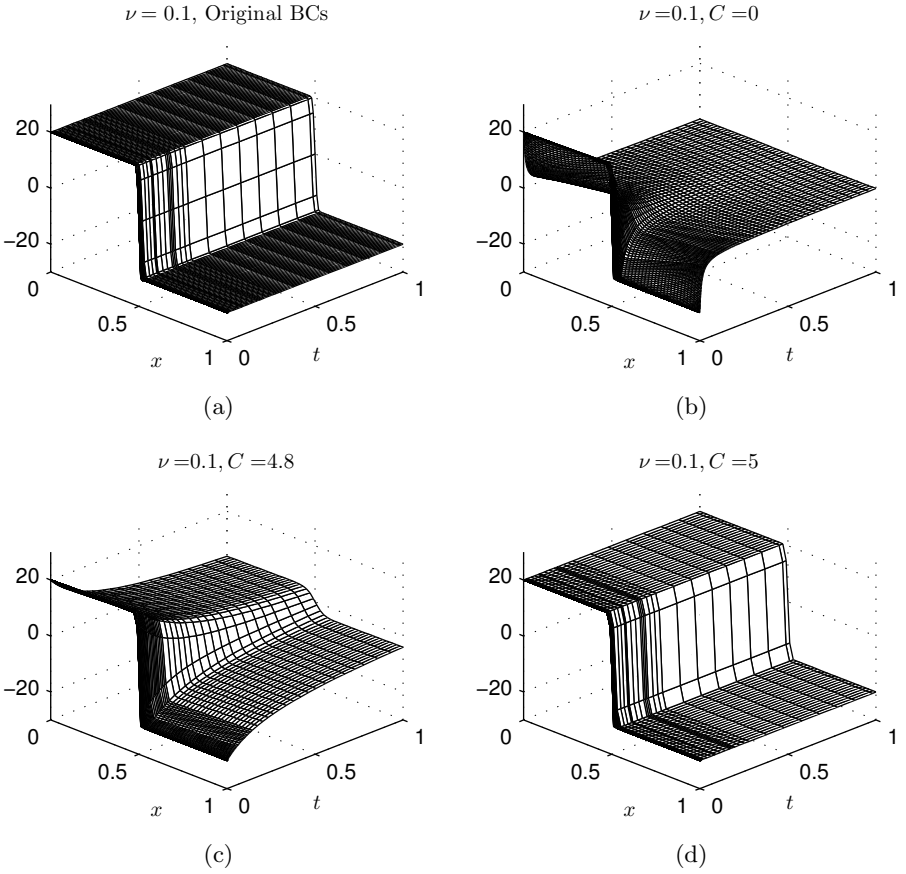


Figure 6.1: Comparison of solutions generated by different BC's.

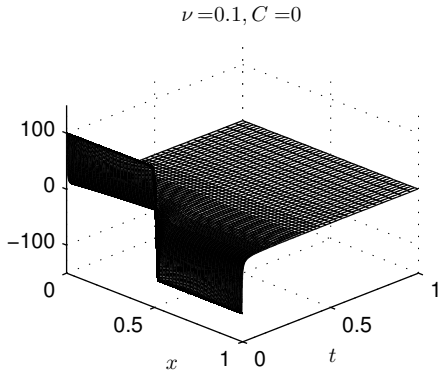


Figure 6.2: If $C = 0$, even higher magnitudes of the initial conditions still gives a zero solution. Here the magnitude at the left boundary is 100.

Chapter 7

Conclusions

In this thesis it has been shown that Burgers' equation with homogeneous Neumann conditions would have wrong solutions. This solutions exists (as the results shows) because of round off errors that occur when the boundary points are implemented as discretized derivatives. The accuracy of the numerically imposed Neumann conditions is not accurate enough to make the solution satisfy the correct boundary conditions – we have shown that the small errors was enough for non-unique solutions to exist.

The issue we have found, is that accuracy of computed solutions cannot be higher than the machine epsilon of the floating point format used. If so is the case, the problem cannot in general be solved by standard numerical methods. However, by some insight in the solution, one may include e.g. specific boundary conditions that restrict the source of the round off errors, which we have tested with success. But without pre-knowledge in the solution, other ways to approach the problem must be used, which is a possible topic for further studies.

The types of accuracy problems that we encountered have to do with the treatment of computing something exactly zero. Two different approximations were proven to generate problems: homogeneous Neumann conditions, and consistence of a root that we know should be invariant with time.

Commenting the latter problem first. Assuming only constant solutions exist; for some initial conditions there is actually a problem for the iterative methods to converge to the right steady state of the initial condition. Since there is no global attractor to the problem (there is no unique steady state for an arbitrary initial condition) a small error of the actual solution makes the problem converge to another steady state. We have seen that this makes the steady state of zero more or less non-existing in finite precision, since new errors are summarized to the old ones in each iteration. However, if there is a globally defined attractor the accuracy problems does not have such an effect. Using e.g. homogeneous Direchlet conditions only have the steady state zero for all initial conditions.

By the bifurcation analysis, we have proved that the non-constant steady states was not uniquely defined – for the discretized system there are most often two different steady

states $h^<$ and $h^>$, which depends on the choice of a constant C, which is a constant obtained from an integration in the solution process of the general steady state solution. From the numerical testing we could conclude that forcing the boundary conditions to choose $C = 0$ in every iteration makes the solution converge to the constant zero solution for the corresponding initial condition, and letting C be a sufficiently large makes a non-constant solution appear. Thus, this confirms the results of the bifurcation analysis. In the bifurcation analysis chapter we also investigated in a stability analysis, where the $h^<$ was proved to be stable and $h^>$ unstable. We have not been able to make any conclusions about this results from the numerical testing. Further studies according the effect of the stability results may be necessary for understanding.

It was also proven that non-constant solutions satisfies some numerical schemes, but for other not. For finite differences, non-constant solutions occur if the scheme is discretized with central differences in space (e.g. Crank Nicolson method), but for a forward in space discretization the non-constant solutions does not exist (explicit Euler method). Hence, to gain more insight in problems that can occur in numerical results, one may study the schemes more in detail, since most likely it is the combination of the how the solution is discretized and the accuracy of the floating point format of the computer that decide how large possible errors may be.

For further research in this area, one may consider if the same kind of problems can occur for other partial differential equations such as the Navier Stokes equation. It is also of interest to know if there are some methods to approach this kind of problems in general.

And some final words: It is important to have an insight in that numerical methods are not always to trust. Most often the impact of errors down to machine epsilon are negligible. But if the existence of them by themselves changes the whole result we have a big problem. Existence of this kind of problems must be taken into account doing scientific computations.

Chapter 8

Appendix: Implementation of numerical schemes

8.1 Explicit Euler implementation

For an explicit finite difference implementation we discretize forward in time and forward in space:

$$u_t = \frac{u_i^{(n+1)} - u^{(n)_i}}{\Delta t} \tag{8.1}$$

$$u_x = \frac{u_{i+1}^{(n)} - u_i^{(n)}}{\Delta x} \tag{8.2}$$

$$u_{xx} = \frac{u_{i+1}^{(n)} - 2u_i^{(n)} + u_{i-1}^{(n)}}{(\Delta x)^2}. \tag{8.3}$$

where $n \in [0, N_t]$ is the time-steps, $i \in [0, N_x]$ is the points of the space discretization, $\Delta x = 1/(N_x - 1)$ is the step-length in space and $\Delta t = T/N_t$ is the step-length in time, where T is the final computation time. (8.1)-(8.3) substituted into the viscid burgers equation (1.6) yields

$$\frac{u_i^{(n+1)} - u^{(n)_i}}{\Delta t} + \frac{f(u_{i+1}^{(n)}) - f(u_i^{(n)})}{\Delta x} = \nu \frac{u_{i+1}^{(n)} - 2u_i^{(n)} + u_{i-1}^{(n)}}{(\Delta x)^2}. \tag{8.4}$$

Now, the time variables need to be on different sides of the equal sign to be written in the final recursive form. Hence the left hand side in (8.4) is splitted up. The expression obtained is

$$u_i^{(n+1)} = u^{(n)_i} + \Delta t \left(\frac{f(u_i^{(n)}) - f(u_{i+1}^{(n)})}{\Delta x} + \nu \frac{u_{i+1}^{(n)} - 2u_i^{(n)} + u_{i-1}^{(n)}}{(\Delta x)^2} \right). \tag{8.5}$$

The Explicit Euler is a first order method, therefore we use a first order discretization of the Neumann boundary conditions: $u_0 = u_1$, $u_{N_x+1} = u_{N_x}$. Written in matrix form, the system of the explicit Euler discretization is

$$\begin{bmatrix} u_1^{(n+1)} \\ u_2^{(n+1)} \\ \vdots \\ u_{N_x-1}^{(n+1)} \\ u_{N_x}^{(n+1)} \end{bmatrix} = \frac{\Delta t}{\Delta x} \begin{bmatrix} 0 \\ f(u_1^{(n)}) - f(u_2^{(n)}) \\ \vdots \\ f(u_{N_x-2}^{(n)}) - f(u_{N_x-1}^{(n)}) \\ 0 \end{bmatrix} + \frac{\nu \Delta t}{(\Delta x)^2} \begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix} \begin{bmatrix} u_1^{(n)} \\ u_2^{(n)} \\ \vdots \\ u_{N_x-1}^{(n)} \\ u_{N_x}^{(n)} \end{bmatrix}. \tag{8.6}$$

8.2 Crank Nicolson implementation

The Crank Nicolson method is a second order finite difference scheme, which is numerically implicit in time. This method is more numerically stable than the explicit Euler scheme, but a bit more cumbersome to implement. The computational cost is also higher, which may be to a disadvantage for large scale problems, to which an explicit method still gives stable results for adequate large step-sizes.

The Crank Nicolson method is based on central differences in space, and the trapezoidal rule in time. Hence, the derivatives of the Burgers' equation are discretized as follows

$$u_t = \frac{u_i^{(n+1)} - u^{(n)_i}}{\Delta t} \tag{8.7}$$

$$u_x = \frac{u_{i+1}^{(n)} - u_{i-1}^{(n)}}{2\Delta x} + \frac{u_{i+1}^{(n+1)} - u_{i-1}^{(n+1)}}{2\Delta x} \tag{8.8}$$

$$u_{xx} = \frac{u_{i+1}^{(n)} - 2u_i^{(n)} + u_{i-1}^{(n)}}{2(\Delta x)^2} + \frac{u_{i+1}^{(n+1} - 2u_i^{(n+1)} + u_{i-1}^{(n+1)}}{2(\Delta x)^2}. \tag{8.9}$$

where $i = [1, N_x]$ is the index of the points in the space discretization, $\Delta x = 1/(N_x - 1)$ is the step lenght in space, $n \in [1, N_t]$ is the index of the time steps and $\Delta t = T/N_t$ is the time step legth, where T is the final computation time. Substituting into the viscid Burgers' equation (1.6) gives for the inner points that

$$\begin{aligned} &\frac{u_i^{(n+1)} - u_i^{(n)}}{\Delta t} + \frac{f(u_{i+1}^{(n)}) - f(u_{i-1}^{(n)})}{2\Delta x} + \frac{f(u_{i+1}^{(n+1)}) - (f u_{i-1}^{(n+1)})}{2\Delta x} \\ &\qquad = \nu \left(\frac{u_{i+1}^{(n)} - 2u_i^{(n)} + u_{i-1}^{(n)}}{2(\Delta x)^2} + \frac{u_{i+1}^{(n+1)} - 2u_i^{(n+1)} + u_{i-1}^{(n+1)}}{2(\Delta x)^2} \right). \end{aligned} \tag{8.10}$$

Now, the hard part is to rewrite this into a recursive formula. Since Burgers' equation has the non-linear term $f(u)$, there is no possibility to move time variable terms with index $n+1$ to the left hand side and similarly move the index n terms to the right hand side directly. The trick is to linearize the equation using Taylor expansion on the non-linear

terms. Applied to the $f(u) = (u^2/2)_x$ term of Burgers' equation the Taylor expansion reads

$$\begin{aligned} f(u_i^{(n+1)}) &= f(u_i^{(n)}) + f_t(u_i^{(n)})\Delta t + \mathcal{O}((\Delta t)^2) \\ &= u_i^{(n)} \frac{(u_i^{(n)})^2}{2} + \frac{u_i^{(n)}(u_i^{(n+1)} - u_i^{(n)})}{\Delta t}\Delta t + \mathcal{O}((\Delta t)^2) \\ &= u_i^{(n)} u_i^{(n+1)} - \frac{(u_i^{(n)})^2}{2} + \mathcal{O}((\Delta t)^2). \end{aligned} \tag{8.11}$$

Substituting (8.10) into (8.11) yields

$$\begin{aligned} &\left(\frac{\Delta t}{4\Delta x} u_{i-1}^{(n)}\right) u_{i+1}^{(n+1)} + \left(1 + \frac{\nu \Delta t}{(\Delta x)^2}\right) u_i^{(n+1)} + \left(-\frac{\Delta t}{4\Delta x} u_{i+1}^{(n)} - \nu \frac{\Delta t}{2(\Delta x)^2}\right) u_{i+1}^{(n+1)} \\ &\qquad = \nu \frac{\Delta t}{2(\Delta x)^2} u_{i-1}^{(n)} + \left(1 - \nu \frac{\Delta t}{(\Delta x)^2}\right) u_i^{(n)} + \nu \frac{\Delta t}{2(\Delta x)^2} u_{i+1}^{(n)}. \end{aligned} \tag{8.12}$$

Since Crank Nicolson is a second order method a second order central differences discretization of the Neumann boundary conditions is applied. I.e.

$$\frac{u_2 - u_0}{2\Delta x} = 0 \implies u_0 = u_2 \tag{8.13}$$

$$\frac{u_{N_x-1} - u_{N_x+1}}{2\Delta x} = 0 \implies u_{N_x+1} = u_{N_x-1}. \tag{8.14}$$

And (8.13)-(8.14) substituted into (8.12) yields the following expressions for the boundary points

$$\left(1 + \frac{\nu \Delta t}{(\Delta x)^2}\right) u_1^{(n+1)} + \left(-\nu \frac{\Delta t}{(\Delta x)^2}\right) u_2^{(n+1)} = \left(1 - \nu \frac{\Delta t}{(\Delta x)^2}\right) u_1^{(n)} + \nu \frac{\Delta t}{(\Delta x)^2} u_2^{(n)} \tag{8.15}$$

$$\left(1 + \frac{\nu \Delta t}{(\Delta x)^2}\right) u_{N_x}^{(n+1)} + \left(-\nu \frac{\Delta t}{(\Delta x)^2}\right) u_{N_x-1}^{(n+1)} = \left(1 - \nu \frac{\Delta t}{(\Delta x)^2}\right) u_{N_x}^{(n)} + \nu \frac{\Delta t}{(\Delta x)^2} u_{N_x-1}^{(n)}. \tag{8.16}$$

All together the Crank Nicolson implementation are written in matrix form as $M\left(u^{(n)}\right) u^{(n+1)} = Au^{(n)}$. Where,

$$M(u^{(n)}) = \begin{bmatrix} a_1 & -a_2 & & & & & \\ a_3 u_1^{(n)} - a_4 & a_1 & -a_3 u_3^{(n)} - a_4 & & & & \\ & a_3 u_2^{(n)} - a_4 & a_1 & -a_3 u_4^{(n)} - a_4 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & a_3 u_{N_x-2}^{(n)} - a_4 & a_1 & -a_3 u_{N_x}^{(n)} - a_4 & \\ & & & & & -a_2 & a_1 \end{bmatrix} \tag{8.17}$$

and,

$$A = \begin{bmatrix} a_5 & a_2 & & & \\ a_4 & a_5 & a_4 & & \\ & \ddots & \ddots & \ddots & \\ & & a_4 & a_5 & a_4 \\ & & & a_2 & a_5 \end{bmatrix}, \tag{8.18}$$

where the coefficients of the matrices are

$$a_1 = 1 + \frac{\Delta t}{(\Delta x)^2}, \quad a_2 = \nu\frac{\Delta t}{(\Delta x)^2}, \quad a_3 = \frac{\Delta t}{4\Delta x}, \quad a_4 = \nu\frac{\Delta t}{2(\Delta x)^2}, \quad a_5 = 1 - \frac{\Delta t}{(\Delta x)^2}. \tag{8.19}$$

The $M(u^{(n)})$ matrix is a banded square matrix, with non-zero entries. Thus, as long as it is non-singular, the system yields numerically stable results. The final recursive formula used for the implementation of the Crank Nicolson method therefore becomes

$$u^{(n+1)} = M(u^{(n)})^{-1} A u^{(n)}. \tag{8.20}$$

8.3 A piecewise linear finite element implementation

Consider the one dimensional viscid Burgers' equation on the space interval $[0, 1]$ with an arbitrary initial condition u_0 and homogeneous Neumann Neumann boundary conditions and with the viscosity parameter ν

$$\begin{cases} u_t + \left(\frac{u^2}{2}\right)_x = \nu u_{xx}, & (x,t) \in (0,1) \times (0,\infty),\ \nu > 0 \\ u_x = 0, & (x,t) \in \{0,1\} \times (0,\infty), \\ u = u_0, & (x,t) \in [0,1] \times \{0\}. \end{cases} \tag{8.21}$$

As usual when doing a finite element implementation, the first step is to rewire the differential equation into variational formulation, which is obtained by multiplying left and right hand side of the PDE with a test-function v and then integrate over the space interval $[0, 1]$. The test function is assumed to vanish at the end points, i.e. $v(0) = v(1) = 0$. Multiplying both left- and right- hand side of the PDE with the test function and integrate over the space interval yields

$$\int_0^1 \left[u_t + \left(\frac{u^2}{2}\right)_x\right] v \,\mathrm{d}x = \int_0^1 \nu u_{xx} v \,\mathrm{d}x. \tag{8.22}$$

Integration by parts of the second integral implies

$$\int_0^1 \left[u_t + \left(\frac{u^2}{2}\right)_x\right] v \,\mathrm{d}x + \int_0^1 \nu u_x v' \,\mathrm{d}x - u_x v|_0^1 = 0. \tag{8.23}$$

The last therm vanishes since $v(0) = v(1) = 0$ by assumption. Thus,

$$\int_0^1 \left[u_t + \left(\frac{u^2}{2}\right)_x\right] v \,\mathrm{d}x + \int_0^1 \nu u_x v' \,\mathrm{d}x = 0. \tag{8.24}$$

We require that the test function v and its derivative v' are square integrable on $[0, 1]$, hence their function space need to be defined as:

$$V_0 = \{v : \|v\|_{L^2[0,1]} < \infty, \|v'\|_{L^2[0,1]} < \infty, v(0) = v(1) = 0\}. \tag{8.25}$$

Thus, the final variational formulation of the differential equation is written as

$$\int_0^1 \left[u_t + \left(\frac{u^2}{2}\right)_x\right] v \,\mathrm{d}x = -\nu \int_0^1 u_x v' \,\mathrm{d}x, \quad \forall v \in V_0. \tag{8.26}$$

The numerical implementation we use is based on the Galerkin method, i.e. finding an approximate solution of u, in the space of continuous piecewise linear functions V_h and letting the test functions be chosen as tent functions φ_i, which are one at the index i corresponding to the grid point x_i and zero elsewhere. The approximated discrete solution denoted by $u^h \in V_h$ is due to the Galerkin method defined as the linear combination of tent functions

$$u^h = \sum_{j=0}^{n+1} \xi_j \varphi_j, \tag{8.27}$$

where ξ_j, $j = 0, 1, \ldots, n+1$, are the $n+2$ coefficients to be determined. The finite element method obtained are: find $u^h \in V_h$ such that

$$\int_0^1 \left[u_t^h + \left(\frac{u_h^2}{2}\right)_x\right] v \,\mathrm{d}x = -\nu \int_0^1 u_x^h v' \,\mathrm{d}x, \quad \forall v \in V_0. \tag{8.28}$$

Substituting (8.27) into (8.28) and letting $v = \varphi_i$, $i = 1, 2 \ldots, n$ yields

$$\int_0^1 \left[\sum_{j=0}^{n+1} \dot{\xi}_j \varphi_j + \frac{1}{2} \sum_{j=0}^{n+1} \xi_j^2 \varphi_j'\right] \varphi_i \,\mathrm{d}x = -\nu \int_0^1 \sum_{j=0}^{n+1} \xi_j \varphi_j' \varphi_i' \,\mathrm{d}x, \quad i = 0, 1, \ldots n. \tag{8.29}$$

This is rearranged to

$$\sum_{j=0}^{n+1}\left[\int_0^1 \varphi_i\varphi_j\,\mathrm{d}x\right]\dot{\xi}_j + \frac{1}{2}\sum_{j=0}^{n+1}\left[\int_0^1 \varphi_j'\varphi_i\,\mathrm{d}x\right]\xi_j^2 = -\nu\sum_{j=0}^{n+1}\left[\int_0^1 \varphi_i'\varphi_j'\,\mathrm{d}x\right]\xi_j, \quad i = 1, 2, \ldots n. \tag{8.30}$$

Define $m_{ij} = \int_0^1 \varphi_i\varphi_j\,\mathrm{d}x$, $b_{ij} = \frac{1}{2}\int_0^1 \varphi_j'\varphi_i\,\mathrm{d}x$ and $k_{ij} = -\int_0^1 \varphi_i'\varphi_j'\,\mathrm{d}x$, where $i = 0, 1, \ldots, n+1$, and $j = 0, 1, \ldots n+1$. Then (8.30) can be written in matrix form as

$$M\dot{\xi} + B(\xi \circ \xi) = \nu K\xi, \tag{8.31}$$

where M, B and K are $(n+2)\times(n+2)$ square matrices containing the elements m_{ij}, b_{ij} and k_{ij}, and $\xi\circ\xi := \left(\xi_0^2, \xi_2^2, \ldots, \xi_{n+1}^2\right)^\top$ is called the Hadamard product of ξ with itself. It is the simple nature of the B matrix that makes the conservation form more handy to use. Using the nonlinear expression uu_x from the beginning does force out a more complicated non-linear vector instead of $B(\xi\circ\xi)$.

The elements of the matrices are not yet evaluated. But due to the nature of the tent functions, the integrals can be solved out easily. No numerical integration is needed. Instead note that the tent functions are defined as

$$\varphi_0(x) = \begin{cases} -(n+1)(x - x_1), & x_0 \le x \le x_1 \\ 0, & \text{otherwise.} \end{cases} \tag{8.32}$$

$$\varphi_i(x) = \begin{cases} (n+1)(x - x_{i-1}), & x_{i-1} \le x \le x_i \\ -(n+1)(x_{i+1} - x), & x_i \le x \le x_{i+1} \\ 0, & \text{otherwise.} \end{cases} \tag{8.33}$$

$$\varphi_{n+1}(x) = \begin{cases} (n+1)(x - x_n), & x_n \le x \le x_{n+1} \\ 0, & \text{otherwise.} \end{cases} \tag{8.34}$$

Their space derivatives are easy to compute as

$$\varphi_0'(x) = \begin{cases} -(n+1), & x_0 \le x \le x_1 \\ 0, & \text{otherwise.} \end{cases} \tag{8.35}$$

$$\varphi_i'(x) = \begin{cases} (n+1), & x_{i-1} \le x \le x_i \\ -(n+1), & x_i \le x \le x_{i+1} \\ 0, & \text{otherwise.} \end{cases} \tag{8.36}$$

$$\varphi'_{n+1}(x) = \begin{cases} (n+1), & x_n \le x \le x_{n+1} \\ 0, & \text{otherwise.} \end{cases} \tag{8.37}$$

Note that the slope of the hat function is $(n+1)$ where $1/(n+1)$ is the step length. By using the statements above, the elements of the matrices are computed by standard integration in one variable technique

$$m_{0,0} = \int_{x_0}^{x_1} \varphi_0^2 \,\mathrm{d}x = \frac{1}{3(n+1)} \tag{8.38}$$

$$m_{n+1,n+1} = \int_{x_n}^{x_{n+1}} \varphi_{n+1}^2 \,\mathrm{d}x = \frac{1}{3(n+1)} \tag{8.39}$$

$$m_{i,i} = \int_{x_{i-1}}^{x_i} \varphi_i^2 \,\mathrm{d}x + \int_{x_i}^{x_{i+1}} \varphi_i^2 \,\mathrm{d}x = \frac{2}{3(n+1)} \tag{8.40}$$

$$m_{i,i+1} = m_{i-1,1} = \int_{x_i}^{x_{i+1}} \varphi_i \varphi_{i+1} \,\mathrm{d}x = \frac{1}{6(n+1)} \tag{8.41}$$

$$b_{0,0} = \frac{1}{2} \int_{x_0}^{x_1} \varphi'_0 \varphi_0 \,\mathrm{d}x = -\frac{1}{4} \tag{8.42}$$

$$b_{n+1,n+1} = \frac{1}{2} \int_{x_n}^{x_{n+1}} \varphi_i^2 \,\mathrm{d}x = -\frac{1}{4} \tag{8.43}$$

$$b_{i,i} = \frac{1}{2} \int_{x_{i-1}}^{x_i} \varphi_i^2 \,\mathrm{d}x + \int_{x_i}^{x_{i+1}} \varphi_i^2 \,\mathrm{d}x = 0 \tag{8.44}$$

$$b_{i,i+1} = \frac{1}{2} \int_{i}^{i+1} \varphi_i \varphi_j \,\mathrm{d}x = -\frac{1}{4} \tag{8.45}$$

$$b_{i-1,i} = \frac{1}{2} \int_{i}^{i+1} \varphi_i \varphi_j \,\mathrm{d}x = \frac{1}{4} \tag{8.46}$$

$$k_{0,0} = \int_{x_0}^{x_1} (\varphi'_0)^2 \,\mathrm{d}x = -(n+1) \tag{8.47}$$

$$k_{n+1,n+1} = \int_{x_n}^{x_{n+1}} (\varphi'_{n+1})^2 \,\mathrm{d}x = -(n+1) \tag{8.48}$$

$$k_{i,i} = \int_{x_{i-1}}^{x_i} (\varphi'_i)^2 \,\mathrm{d}x + \int_{x_i}^{x_{i+1}} (\varphi'_i)^2 \,\mathrm{d}x = -2(n+1) \tag{8.49}$$

$$k_{i,j} = k_{j,i} = \int_{i}^{i+1} \varphi'_i \varphi'_j \,\mathrm{d}x = (n+1) \tag{8.50}$$

Hence the system matrices are

$$M = \frac{1}{6(n+1)} \begin{bmatrix} 2 & 1 & & & & \\ 1 & 4 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & 1 & 4 & 1 & \\ & & & 1 & 2 & \end{bmatrix}_{(n+2)\times(n+2)} \tag{8.51}$$

$$B = \frac{1}{4} \begin{bmatrix} -1 & 1 & & & & \\ -1 & 0 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -1 & 0 & 1 & \\ & & & -1 & 1 & \end{bmatrix}_{(n+2)\times(n+2)} \tag{8.52}$$

$$K = (n+1) \begin{bmatrix} -1 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & 1 & -2 & 1 & \\ & & & 1 & -1 & \end{bmatrix}_{(n+2)\times(n+2)} \tag{8.53}$$

Obviously M is invertible since it has linearly independent columns. Thus, (8.31) can be reformed into

$$\dot{\xi} = M^{-1}(\nu K\xi - B(\xi \circ \xi)), \tag{8.54}$$

which is a system of first order ordinary differential equations, that can be solved by e.g. some numerical time stepping scheme.

Bibliography

[1] E. Allen, J. Burns, and D. Gilliam. Numerical Approximations of the Dynamical System Generated by Burgers' Equation with Neumann-Direchlet Boundary Conditions. *ESAIM – Mathematical Modelling and Numerical Analysis*, 47:1465–1492, 2013.

[2] E. Allen, J. Burns, D. Gilliam, J. Hill, and V. Shubow. The Impact of Finite Precision Arithmetic and Sensitivity on the Numerical Solution of Partial Differential Equations. *Mathematical and Computer Modelling*, 35:1165–1196, 2002.

[3] J. Burns, A. Balogh, D. Gilliam, J. Hill, and V. Shubow. Numerical Stationary Solutions for a Viscous Burgers' Equation. *Journal of Mathematical Systems, Estimation, and Control*, 8(2):1–16, 1998.

[4] CH. I. Byrnes, D. Gilliam, and V. Shubow. On the Global Dynamics of a Controlled Viscous Burgers' Equation. *Journal of Dynamical and Control Systems*, 4(4):457 – 519, 1998.

[5] C. Cao and E. Titi. Asymptotic Behaviour of Viscous 1-D Scalar Conservation Laws With Neumann Boundary Conditions. *Third Palestinian Mathematics Conference, World Scientific*, 2001.

[6] L. C. Evans. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, 1998.

[7] E. Hopf. The Partial Differential Equation $u_t + uu_x = \nu\mu_{xx}$. *Communications on Pure and Applied Mathematics*, 3:201 – 230, 1950.

[8] W. Kahan. IEEE Standard 754 for Binary Floating-Point Arithmetic. *Lecture Notes University of California*, 1997.

[9] R. J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.

[10] V. Q. Nguyen. A Numerical Study of Burgers' Equation With Robin Boundary Conditions. Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 2001.

[11] S. M. Pugh. Finite Element Approximations of Burgers' Equation. Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1995.

[12] E. D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, volume 6 of *Textbooks in Applied Mathematics*. Springer, New York, 1998.