On Entropy Stable Finite Volume Methods for Scalar Conservation Laws

Loke Kristoffersson

Bachelor's thesis
2023:K21

Lund University

Faculty of Science
Centre for Mathematical Sciences
Numerical Analysis

Abstract

In this thesis, entropy conservative and entropy stable finite volume methods for one-dimensional scalar conservation laws are studied. The need for weak solutions is shown and entropy is explored to obtain uniqueness. Using discrete analogues of properties of entropy solutions, entropy stable finite volume methods are constructed and implemented to solve a number of scalar conservation laws, in particular the transport equation, Burgers' equation, a traffic flow problem and a model for enhanced oil recovery. The solutions obtained are accurate leading up to shocks, at which point the solutions break down. The viscous approximation to the scalar conservation law is solved to further understand the behaviour of solutions after shocks. It is theoretically shown and experimentally validated that the finite volume schemes are stable. Second order convergence is observed for all test problems with several different choices of numerical fluxes. The experiments indicate that numerical fluxes conserving or dissipating the squared entropy have the smallest errors.

Acknowledgements

I would like to express my deepest gratitude to my supervisor Viktor Linders. I thank you for your time, patience and understanding. The valuable advice and clear guidance you provided me has been invaluable and has left me truly proud of this project.

Table of contents

1 Introduction

One-dimensional *scalar conservation laws* are time-dependent partial differential equations (PDEs) that arise in a wide range of interesting models in physics and engineering, such as traffic flow models and enhanced oil recovery models presented in [1]. The homogeneous scalar conservation laws that we will be focusing on have the form

$$u_t + f(u)_x = 0, \qquad u(a,t) = u(b,t), \qquad u(x,0) = u_0(x), \tag{1}$$

on periodic domains $(x,t) \in [a,b] \times \mathbb{R}_+$, where $u(x,t)$ is the unknown function or *conserved quantity*, f is the flux function and u_0 is the initial function. Conservation laws are so called because they conserve a measurable quantity in a system as time develops. If we take the quantity $\int_a^b u\,dx$ and look at the time derivative of this integral we get the identity

$$\frac{d}{dt}\int_a^b u\,dx = \int_a^b u_t\,dx = \int_a^b -f(u)_x\,dx = f(u(a,t)) - f(u(b,t)). \tag{2}$$

The identity (2) states that the quantity $\int_a^b u\,dx$ can only change due to the flow of u across the boundaries. For non-periodic domains, this means the quantity $\int_a^b u\,dx$ changes when the flow is not equal at the boundaries. In our case however, the domain is periodic and the right hand side vanishes, leaving us with the quantity $\int_a^b u\,dx$ conserved for all times.

For nonlinear fluxes f, we find by following [2] and [1] that smooth solutions to scalar conservation laws break down after a certain amount of time. We see this by looking at characteristics of solutions and show that they intersect even for smooth initial functions. To remedy this, we introduce weak solutions to allow for discontinuities in our solutions. We find, however, that weak solutions lack uniqueness. We introduce the concept of entropy to impose entropy conditions on the weak solutions that will guarantee uniqueness in the weak solutions. Additionally, these entropy conditions lead to nonlinear bounds on the solutions. These bounds can be mimicked in the discrete setting, which implies stability in the the numerical method [3].

Scalar conservation laws are often solved using the finite volume method studied in LeVeque's book on numerical methods for conservation laws [4]. The finite volume method approximates cell averages instead of point-wise approximations to handle potential discontinuities. We are able to prove stability for this numerical method using discrete analogues of results from the analysis of entropy in the continuous case. With this method we solve a number of different scalar conservation laws, including a simple traffic flow model and an enhanced oil recovery model. We present solutions to these conservation laws and discuss their accuracy, and we study what happens when shocks are encountered.

The thesis is structured as follows. In Section 2 we present theory for the continuous problem. We use characteristics in Section 2.1 to show that smooth solutions break down. We introduce the concept of *weak solutions* in Section 2.2. We soon encounter uniqueness problems that we overcome by looking at different entropy conditions on the weak solutions in Section 2.3. In Section 3 we use our knowledge of entropy solutions to find numerical methods for solving (1). We introduce the finite volume method in Section 3.1. In Section 3.2 we define the entropy stable numerical flux that is the key to stability in the numerical method. We explore entropy solutions and show their behavior and properties in Section 4. We will solve some simple examples of conservation laws to highlight the strengths of the approach, but also show areas of uncertainty where research is still ongoing. In Section 5 we summarize our findings and draw conclusions. Possible extensions of this work are discussed in Section 6.

2 Theory of scalar conservation laws

In this section we will see that smooth solutions cannot always be found for scalar one-dimensional conservation laws. We will remedy this by introducing weak solutions that pose non-uniqueness problems. The concept of entropy will be introduced to obtain uniqueness in our solutions and impose conditions that we can mimic in the discrete case.

2.1 Characteristics

For all one-dimensional nonlinear scalar conservation laws it is the case that we cannot find a smooth solution for all time. To show this, we make use of *characteristics.* These are lines along which the solution $u(x(t), t)$ is constant in time, meaning $u(x(t), t) = u_0(x_0)$ where $x(0) = x_0$. The idea is to look at $\frac{\mathrm{d}u}{\mathrm{d}t}$ and match terms with the conservation law to create these lines. Using that $x = x(t)$ is a function of time we can derive $\frac{\mathrm{d}u}{\mathrm{d}t}$ using the chain rule as

$$\frac{\mathrm{d}u}{\mathrm{d}t} = u_t + \frac{\mathrm{d}x}{\mathrm{d}t} u_x.$$

Since we desire that u is constant in time we can set this to 0 and use the conservation law (1) to find $\frac{\mathrm{d}x}{\mathrm{d}t}$.

Assuming that f is differentiable, we get the following set of equations:

$$\begin{cases} \frac{\mathrm{d}u}{\mathrm{d}t} = u_t + \frac{\mathrm{d}x}{\mathrm{d}t} u_x = 0, \\ u_t + f(u)_x = u_t + f'(u) u_x = 0. \end{cases}$$

Here we can set $x'(t) = f'(u(x(t), t))$ and since u is constant in time by definition, we have $x'(t) = f'(u_0(x_0))$. Integrating in time gives us the characteristic

$$x(t) = f'(u_0(x_0))t + x_0. \tag{3}$$

Perhaps one of the most exemplified conservation laws is *Burgers' equation* with flux $f(u) = u^2/2$. This simple example is enough to show that solutions break down. We observe that $f'(u) = u$ which, by (3) gives characteristics of the form

$$x(t) = u_0(x_0)t + x_0. \tag{4}$$

We now consider a *Riemann problem* which is the initial value problem (1) with initial data with a jump discontinuity of the form

$$u_0(x) = \begin{cases} u_l & \text{if } x \leq 0 \\ u_r & \text{if } x > 0. \end{cases} \tag{5}$$

Riemann problems help us look at what happens around jump discontinuities in various solutions. Let us consider the case where $u_l = 1$ and $u_r = 0$. This leads to characteristics with slope 1 when $x_0 < 0$ and slope 0 when $x_0 > 0$. Pictured in Figure 1 are the intersecting characteristics.

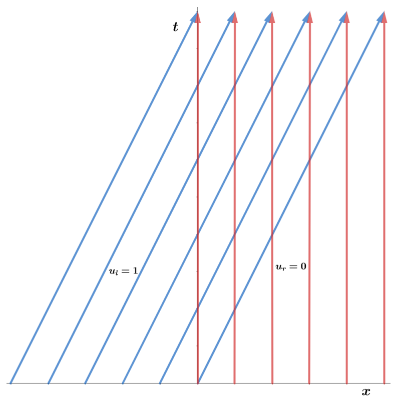
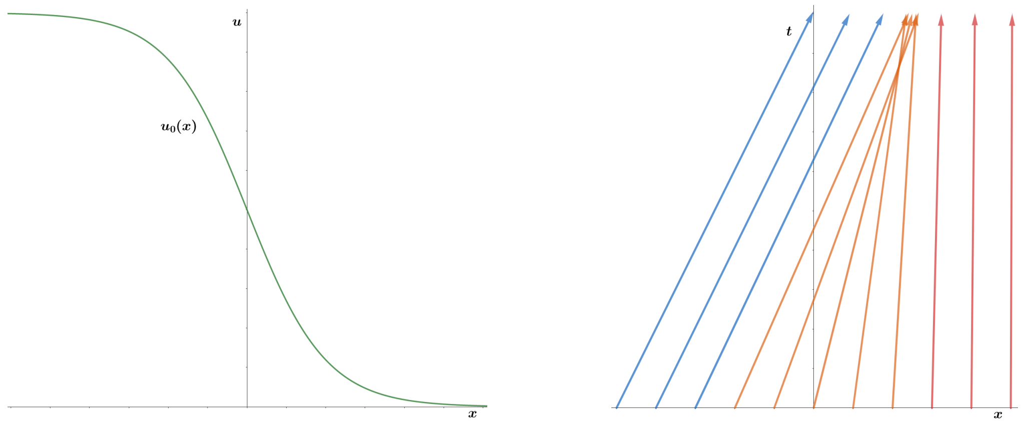


Figure 1: Intersecting characteristics for the Riemann problem (3) with $u_l = 1$, $u_r = 0$.

We observe a strange behaviour when the characteristics intersect, where the solution takes more than one different values at the same point x. This means the solution will break down since it is no longer a function. This strange behaviour persists even for smooth initial data, pictured in Figure 2 where we have used $u_0(x) = 1/(1+e^{5x})$. This behaviour implies that we cannot guarantee smooth solutions for our conservation law, even if we have smooth initial data. We have now shown using characteristics that solutions to (1) break down, and that we need an alternative type of solution.



(a) Initial data

(b) Intersecting characteristics

Figure 2: Intersecting characteristics with smooth initial data $u_0(x) = 1/(1+e^{5x})$.

2.2 Weak solutions and non-uniqueness

Since we observed that solutions to one-dimensional scalar conservation laws break down, we introduce the concept of a *weak solution*. Weak solutions are so-called because they allow for discontinuities in the solution. In this section, we will find that they pose a problem of non-uniqueness.

To define the weak solution, we need a smooth test function $\phi \in C_c^1(\mathbb{R} \times \mathbb{R}_+)$ where C_c^1 is the space of all continuously differentiable functions with compact support. This test function is multiplied to both sides of (1) and using integration by parts we can define the weak solution as the solution of the resulting integral. We get that

$$\begin{aligned}
0 &= \int_{\mathbb{R}\times\mathbb{R}_+} \phi(u_t + f(u)_x)\,dxdt \\
&= \int_{\mathbb{R}\times\mathbb{R}_+} u_t\phi\,dxdt + \int_{\mathbb{R}\times\mathbb{R}_+} f(u)_x\phi\,dxdt \\
&= \int_{\mathbb{R}} \left(\underbrace{[u\phi]_0^\infty}_{=-u(x,0)\phi(x,0)} - \int_{\mathbb{R}_+} u\phi_t\,dt \right) dx + \int_{\mathbb{R}_+} \left(\underbrace{[f(u)\phi]_{-\infty}^{\infty}}_{=0} - \int_{\mathbb{R}} f(u)\phi_x\,dx \right) dt \\
&= -\left(\int_{\mathbb{R}\times\mathbb{R}_+} u\phi_t + f(u)\phi_x dxdt + \int_{\mathbb{R}} u(x,0)\phi(x,0)dx \right).
\end{aligned}$$

Definition 1. (Weak solution) *A function $u \in L^1(\mathbb{R}\times\mathbb{R}_+)$ is a weak solution to the conservation law (1) if for all test functions $\phi \in C_c^1(\mathbb{R}\times\mathbb{R}_+)$*

$$\int_{\mathbb{R}\times\mathbb{R}_+} u\phi_t + f(u)\phi_x dxdt + \int_{\mathbb{R}} u(x,0)\phi(x,0)dx = 0. \tag{6}$$

Note that since ϕ has compact support in $\mathbb{R}\times\mathbb{R}_+$, the only term we are left with from the boundaries is the one for $t = 0$, the initial condition. Definition 1 only imposes the condition for u to be in L^1, meaning we no longer require our solution to be differentiable or even continuous. This implies that the solution can contain discontinuities which appear in physics as *shock waves*.

It turns out that shock waves cannot be arbitrary curves in the $x - t-$plane; they must satisfy a certain condition. The *Rankine-Hugoniot condition* can be derived using Leibnitz' rule for differentiation under the integral sign on the time derivative of the conserved quantity $\frac{\mathrm{d}}{\mathrm{d}t}\int_a^b u\,dx$ (see Appendix 7.1).

Theorem 1. *(Rankine-Hugoniot jump condition) If the function u is a weak solution to the conservation law (1) with trace values u^- and u^+ on either side of a shock, it holds that*

$$s(t) = \frac{f(u^-(t)) - f(u^+(t))}{u^-(t) - u^+(t)} \tag{7}$$

where $s(t)$ is the speed of the shock.

The Rankine-Hugoniot condition (7) states that the speed of the shock must be equal to the jump in flux across the shock divided by the jump in the solution. With this condition we can determine if a shock is stationary or if it is moving, and if it is, in which direction. It is also a powerful tool with which we can find solutions to Riemann problems.

Example 1. *(Solution to Riemann problem using Rankine-Hugoniot) Earlier we used the Riemann problem* (5)*with $u_l = 1$ and $u_r = 0$ to show that we cannot find smooth solutions for all initial conditions. Now we instead consider a weak solution to this problem. At $x = 0$ we encounter a discontinuity, where we can use the Rankine-Hugoniot condition to derive the speed of the shock. For this example we will once again use Burgers' flux $f(u) = u^2/2$ to find that the shock speed is given by*

$$s(t) = \frac{f(u_l) - f(u_r)}{u_l - u_r} = \frac{1^2/2 - 0^2/2}{1-0} = \frac{1}{2}.$$

With this we can construct a weak solution to (1) by noting that the shock is located along the line $x(t) = \frac{1}{2}t$. We obtain that

$$u(x,t) = \begin{cases} 1 & \text{if } x < \frac{1}{2}t \\ 0 & \text{if } x > \frac{1}{2}t. \end{cases}$$

Earlier we saw that the characteristics intersected. Now we instead have a shock along the previous intersection as pictured in red in Figure 3. Note that shocks are not a problem here since we do not require continuity. □

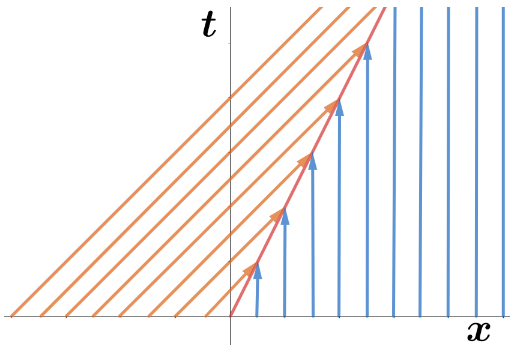


Figure 3: Characteristics for the weak solution to the Riemann problem with $u_l = 1$, $u_r = 0$.

Example 2. *Let us now consider an example where $u_l < u_r$, say $u_l = 0$ and $u_r = 1$. The weak solution for this problem together with the inviscid Burgers' flux can once again be reached using the Rankine-Hugoniot condition*

$$s(t) = \frac{f(u_l) - f(u_r)}{u_l - u_r} = \frac{0^2/2 - 1^2/2}{0 - 1} = \frac{1}{2}.$$

One possible weak solution is then

$$u(x,t) = \begin{cases} 0 & \text{if } x < \frac{1}{2}t \\ 1 & \text{if } x > \frac{1}{2}t. \end{cases}$$

But this is in fact not the only weak solution for this problem, we can for instance construct a solution for every $0 < \alpha < 1$ of the form

$$u(x,t) = \begin{cases} 0 & \text{if } x < \frac{\alpha}{2}t \\ \alpha & \text{if } \frac{\alpha}{2}t < x < \frac{\alpha+1}{2}t \\ 1 & \text{if } \frac{\alpha+1}{2}t < x. \end{cases}$$

This is done by using the Rankine-Hugoniot condition on every jump in the solution. With this, we have an infinite amount of solutions for our conservation law implying a lack of uniqueness. □

2.3 Entropy conditions and solutions

To obtain uniqueness for our weak solutions we need to impose additional conditions. These conditions are called *entropy conditions*. We look at some of the first entropy conditions that were introduced to tackle the problem of non-uniqueness.

The first condition, presented by Peter D. Lax, is based on characteristics. Characteristics can be thought of as representing the flow of information in our solution, so it is important that they flow out of the initial condition on the x-axis. If we look at the examples above, Figure 3 shows the characteristics flowing from the initial data into the shock, which is admissible. Figure 4 shows the inadmissible case where the characteristics flow from the initial condition but also away from the shock.

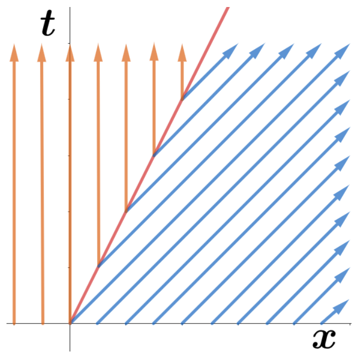


Figure 4: Characteristics for the weak solution to the Riemann problem with $u_l = 0$, $u_r = 1$.

If the characteristics flow out from the shock, we are in a sense creating information at the shock which is very unnatural and undesired. In the case of the inviscid Burgers' equation, this leads to the requirement that

$$u^-(t) > s(t) > u^+(t).$$

This can however be generalized to any strictly convex functions f in the conservation law. To generalize we simply realize that if $u^- > u^+$ we can use the convexity of f to get $f'(u^-) > f'(u^+)$, so we can generalize the requirement to the *Lax entropy condition*,

Definition 2. (Lax entropy condition) Let u be a weak solution of (1) and u^-, u^+ be trace values on either side of a shock σ with speed $s(t)$. The shock σ satisfies the *Lax entropy condition* if for a strictly convex f,

$$f'(u^-(t)) > s(t) > f'(u^+(t)). \tag{8}$$

Condition 8 was first presented by Peter D. Lax in [2]. It tells us that whenever the flux is strictly convex, we can use the Lax entropy condition to justify that our solution will be unique in the case when $u^- > u^+$. If we on the other hand have that $u^- < u^+$, the Lax entropy condition no longer holds and we cannot guarantee uniqueness in the solution. We need a way to uniquely fill the gap in our solution when $u^- < u^+$. For this we introduce *rarefaction waves.*

Rarefaction waves use self similarity in the solutions and convexity of f to produce a unique solution of the form $u(x,t) = (f')^{-1}(x/t)$ in the gap in the solution. We can now create weak solutions to conservation laws for strictly convex f where $u_l < u_r$. For example the Riemann problem (5) with $u_l < u_r$ has solution

$$u(x,t) = \begin{cases} u_l & \text{if } x \leq f'(u_l)t, \\ (f')^{-1}(x/t) & \text{if } f'(u_l)t < x \leq f'(u_r)t, \\ u_r & \text{if } x > f'(u_r)t, \end{cases} \tag{9}$$

which satisfies the Lax entropy condition.

We now have formulas for solutions for both cases $u_l < u_r$ and $u_l > u_r$ whenever f is strictly convex. This result leads to the second entropy condition introduced by Olga Oleinik in effort to generalize the Lax entropy condition to more general fluxes.

Definition 3 (Oleinik entropy condition)**.** Let u be a weak solution of (1) and let u_l, u_r be the values on either side of a shock with speed s. (For notational convenience we drop the time dependence in all the quantities.) The solution satisfies the Oleinik entropy condition if

$$s \leq \frac{f(u^*) - f(u_l)}{u^* - u_l} \tag{10}$$

for all u^* between u_l and u_r.

Condition 10 has once again been shown by Peter D. Lax in [2]. As a consequence of this condition, we can construct solutions to Riemann problems when f has a finite amount of inflection points by combining shocks and rarefaction waves. We will however focus on yet another entropy condition that is an equivalent generalization of the point-wise Oleinik entropy condition at a shock. This generalization is based on the *viscous* approximation to the scalar conservation law,

$$u_t^\epsilon + f(u^\epsilon)_x = \epsilon u_{xx}^\epsilon, \tag{11}$$

where $\epsilon > 0$ is a small parameter. Here the addition of the second derivative u_{xx} is what makes the equation *viscous.* Because of the viscous term, we now have a convection-diffusion equation to which solutions are continuous as shown by Nash in [5].

Of great importance in the notion of entropy solutions are the functions $\eta(u)$, the entropy function, which is *any* strictly convex function, and $q(u)$, the entropy flux, defined as

$$q(u) = \int_0^u f'(s)\eta'(s)\, ds.$$

Together, the pair (η, q) is called an *entropy pair.* From this definition of q we get the relation

$$q'(u) = \eta'(u)f'(u).$$

We can multiply (11) by $\eta'(u)$ and use this relation alongside the chain rule to get,

$$\begin{aligned}
&\eta'(u^\epsilon)u_t^\epsilon + \eta'(u^\epsilon)f(u^\epsilon)_x = \epsilon\eta'(u^\epsilon)u_{xx}^\epsilon \\
\implies\ &\eta'(u^\epsilon)u_t^\epsilon + \eta'(u^\epsilon)f'(u^\epsilon)u_x^\epsilon = \epsilon\eta'(u^\epsilon)u_{xx}^\epsilon \\
\implies\ &\eta'(u^\epsilon)u_t^\epsilon + q'(u^\epsilon)u_x^\epsilon = \epsilon\eta'(u^\epsilon)u_{xx}^\epsilon \\
\implies\ &\eta(u^\epsilon)_t + q(u^\epsilon)_x = \epsilon\eta(u^\epsilon)_{xx} - \epsilon\eta''(u^\epsilon)(u_x^\epsilon)^2.
\end{aligned}$$

By the convexity of η, we have that $\eta''(u^\epsilon) \geq 0$ and we obtain the inequality

$$\eta(u^\epsilon)_t + q(u^\epsilon)_x \leq \epsilon\eta(u^\epsilon)_{xx}. \tag{12}$$

It can be shown that the limit $u = \lim_{\epsilon\to 0} u^\epsilon$ exists and is a weak solution of the conservation law (1). Using this limit, the inequality (12) becomes the *entropy inequality.*

Theorem 2. *(Entropy inequality) For a weak solution u to a scalar conservation law* (1)*, it holds for every entropy pair* (η, q) *that*

$$\eta(u)_t + q(u)_x \leq 0, \tag{13}$$

where equality holds until a shock is encountered.

It has been shown in [6] that a weak solution u satisfies the entropy equality (13) if and only if it satisfies the Oleinik entropy condition (10), thus making them equivalent. A weak solution u to (1) that satisfies the entropy inequality (13) in the weak sense is called an *entropy solution.* There are several interesting properties that follow from the concept of entropy, but we will make use of only one. It turns out that we can bound the integral of $\eta(u)$ over space, essentially putting a bound on u. We have that an entropy solution satisfies the inequality

$$\frac{\mathrm{d}}{\mathrm{d}t}\int \eta(u)\, dx \leq 0, \tag{14}$$

where equality holds until a shock is encountered. This inequality can be reached by integrating (13) in space and using the periodic boundary conditions. This property will be mimicked in the discrete case because of its direct implication on the stability of the numerical method.
Other interesting properties can be derived from the entropy inequality (13). However, since they are not of immediate importance for the goals of this thesis, they will only be briefly mentioned. Further details are found in the lecture notes by Siddhartha Mishra [1].

- Using (14) alongside the entropy functions $\eta(u) = |u|^p$, $p \geq 1$, we can get bounds on the solution in L^p spaces, since

$$\begin{aligned}\int \eta(u(x,t))\,dx \leq \int \eta(u_0(x))\,dx \qquad &\Longrightarrow \int |(u(x,t))|^p\,dx \leq \int |(u_0(x))|^p\,dx \\ &\Longrightarrow \quad ||u(\cdot,t)||_{L^p} \leq ||u_0||_{L^p}.\end{aligned} \tag{15}$$

- Entropy solutions are *total variation diminishing,* giving a bound on the spatial derivative,

$$||u(\cdot,t)||_{TV} \leq ||u_0||_{TV}, \qquad ||u||_{TV} := \int_a^b |u_x|\,dx. \tag{16}$$

- Two entropy solutions u and v with initial data u_0 and v_0 satisfy

$$\int |u(x,t) - v(x,t)|\,dx \leq \int |u_0(x) - v_0(x)|\,dx, \qquad \text{for all } t > 0. \tag{17}$$

  If u_0 and v_0 are the same initial function, the inequality states that u and v will be equal for all time, implying uniqueness.

3 Entropy-stable finite volume method

In this section we construct entropy stable finite volume methods that will be used to create snapshots of solutions in time. We impose conditions on the numerical flux to ensure the method is entropy conservative and provide a proof for this property.

3.1 Introduction to finite volume methods

Finite volume methods (FVMs) are often used for conservation laws because of the appearance of discontinuities. The FVM approximates cell averages of the function u, treating the boundaries of each cell as Riemann problems, using theory from previous sections. To begin the discretization, we split the system, in this case the interval $[a, b]$, into N intervals $[x_{j-1/2}, x_{j+1/2}]$ of size Δx where $x_{j\pm1/2} = (j \pm 1/2 - 1/2)\Delta x$, $j = 1, 2, ..., N$. We view $\hat{u}_j^n$ as an approximation of the cell average

$$\bar{u}_j^n := \frac{1}{\Delta x}\int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t_n)\, dx, \tag{18}$$

rather than an approximation to the point-wise value u_j^n. This interpretation makes sense since we will discretize the integral form of the conservation law over a single cell where these terms appear,

$$\int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t_{n+1})\, dx - \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t_n)\, dx = -\left[\int_{t_n}^{t_{n+1}} f(u(x_{j+1/2}, t))\, dt - \int_{t_n}^{t_{n+1}} f(u(x_{j-1/2}, t))\, dt\right]. \tag{19}$$

If we divide (19) by Δx and use the definition of cell averages, we get

$$\bar{u}_j^{n+1} - \bar{u}_j^n = -\frac{1}{\Delta x}\left[\int_{t_n}^{t_{n+1}} f(u(x_{j+1/2}, t))\, dt - \int_{t_n}^{t_{n+1}} f(u(x_{j-1/2}, t))\, dt\right]. \tag{20}$$

Since the finite volume method approximates cell averages, we work with constant solution values in each cell, creating discontinuities at the boundaries of each cell. These discontinuities create Riemann problems at the boundaries of each cell. Because of this, one usually constructs a numerical flux function f_S that approximates the average flux through the cell boundary $x_{j+1/2}$ over the time interval $[t_n, t_{n+1}]$,

$$f_S(u_j, u_{j+1}) \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} f(u(x_{j+1/2}, t))\, dt. \tag{21}$$

The numerical flux is defined in section 3.2 as it is a major component in the entropy-stable numerical method. If we apply (21) to (20), we would get the method in *conservative form*,

$$\hat{u}_j^{n+1} - \hat{u}_j^n = -\frac{\Delta t}{\Delta x}\left[f_S(\hat{u}_j^n, \hat{u}_{j+1}^n) - f_S(\hat{u}_{j-1}^n, \hat{u}_j^n)\right]. \tag{22}$$

The conservative form is one of the requirements of the celebrated Lax-Wendroff theorem [4], which gives conditions for a numerical scheme to converge to a weak solution in the limit of vanishing Δx and Δt.

In this thesis we will leave time continuous and treat the semi-discretization

$$\frac{\mathrm{d}\hat{u}_j}{\mathrm{d}t} = -\frac{f_S(\hat{u}_j, \hat{u}_{j+1}) - f_S(\hat{u}_{j-1}, \hat{u}_j)}{\Delta x}. \tag{23}$$

To account for our periodic boundary conditions, we define the discretizations at the boundaries separately, yielding the system

$$\begin{aligned}
\frac{\mathrm{d}\hat{u}_1}{\mathrm{d}t} &= -\frac{f_S(\hat{u}_1, \hat{u}_2) - f_S(\hat{u}_N, \hat{u}_1)}{\Delta x},\\
\frac{\mathrm{d}\hat{u}_i}{\mathrm{d}t} &= -\frac{f_S(\hat{u}_j, \hat{u}_{j+1}) - f_S(\hat{u}_{j-1}, \hat{u}_j)}{\Delta x},\\
\frac{\mathrm{d}\hat{u}_N}{\mathrm{d}t} &= -\frac{f_S(\hat{u}_N, \hat{u}_1) - f_S(\hat{u}_{N-1}, \hat{u}_N)}{\Delta x}.
\end{aligned} \tag{24}$$

This system of equations can be written as an equation of matrices using the skew symmetric differentiation matrix $\mathbf{Q}$ and flux matrix $\mathbf{F}$ defined as

$$\mathbf{Q} = \frac{1}{2\Delta x}\begin{bmatrix} 0 & 1 & & \dots & -1 \\ -1 & 0 & 1 & & \\ & -1 & 0 & 1 & \\ & & & \ddots & \\ 1 & & \dots & -1 & 0 \end{bmatrix}, \qquad \mathbf{F}_{ij} = f_S(\hat{\mathbf{u}}_i, \hat{\mathbf{u}}_j), \qquad i,j = 1,2,...,N.$$

With these matrices, the system (24) can be written in matrix form as

$$\frac{\mathrm{d}\hat{\mathbf{u}}}{\mathrm{d}t} = -2(\mathbf{Q} \circ \mathbf{F})\mathbf{1}, \tag{25}$$

where $\circ$ is the *Hadamard*, or point-wise, product.

In the results section we will also look at the effect of adding a small viscosity term to the conservation law. This means we are instead solving the problem $u_t + f(u)_x = \epsilon u_{xx}$ to which solutions, as we saw in Section 2.3, are continuous. To discretize this new second derivative term we use the periodic Laplacian matrix $\mathbf{K}$, defined as

$$\mathbf{K} = \frac{1}{\Delta x^2}\begin{bmatrix} 2 & -1 & 0 & \dots & 0 & -1 \\ -1 & 2 & -1 & 0 & \dots & 0 \\ 0 & -1 & 2 & -1 & 0 & \dots \\ & & & \ddots & & \\ 0 & \dots & 0 & -1 & 2 & -1 \\ -1 & 0 & \dots & 0 & -1 & 2 \end{bmatrix}.$$

Equation (25) now becomes

$$\frac{\mathrm{d}\hat{\mathbf{u}}}{\mathrm{d}t} = -2(\mathbf{Q} \circ \mathbf{F})\mathbf{1} - \epsilon\mathbf{K}\hat{\mathbf{u}}. \tag{26}$$

3.2 Entropy stability

In section 2.3 we saw that meaningful solutions must satisfy the entropy inequality (14). More importantly for us is that we have equality when the solution has not yet encountered a shock. To mimic this conservation of entropy in the discrete case, we impose three conditions on the numerical flux f_S presented by Jesse Chan in [3]:

$$\begin{aligned} f_S(u,u) &= f(u), && \text{(consistency)} \\ f_S(u_l,u_r) &= f_S(u_r,u_l), && \text{(symmetry)} \\ (\eta'(u_l) - \eta'(u_r))f_S(u_l,u_r) &= \psi(u_l) - \psi(u_r), && \text{(entropy conservation)} \end{aligned} \tag{27}$$

where ψ is the *entropy potential* defined as $\psi(u) = \eta'(u)f(u) - q(u)$. A flux that satisfies these three conditions is called *entropy conservative.*

Proposition 1. *A numerical flux defined as*

$$f_S(u_l,u_r) = \begin{cases} f(u_l) & \text{if } u_l = u_r, \\ \frac{\psi(u_l)-\psi(u_r)}{\eta'(u_l)-\eta'(u_r)} & \text{if } u_l \neq u_r, \end{cases} \tag{28}$$

satisfies all three conditions in (27).

Proof. If we use the entropy conservation condition in (27) to define the numerical flux, we get

$$f_S(u_l,u_r) = \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)}, \tag{29}$$

which guarantees conservation of entropy. From this definition it is clear that the numerical flux is symmetric, fulfilling the second condition. To treat the final condition, consistency, we note that if $u_l = u_r$, the flux in this case is undefined and thus we have a discontinuity when $u_l = u_r$. We separately define the function as $f(u_l)$ to fulfill consistency, and note that this is a continuous extension. To see this, we consider the limit as $u_l \to u_r$ in (29) and get

$$\lim_{u_l \to u_r} f_S(u_l, u_r) = \lim_{u_l \to u_r} \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)} = \frac{0}{0}.$$

Here we can use L'Hôpital's rule and the definition of ψ to determine the limit as $u_l \to u_r$

$$\begin{aligned}
\lim_{u_l \to u_r} \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)} &\overset{\text{L'H}}{=} \lim_{u_l \to u_r} \frac{\psi'(u_l)}{\eta''(u_l)} \\
&= \lim_{u_l \to u_r} \frac{(\eta'(u_l) f(u_l) - \int_0^{u_l} \eta'(s) f'(s)\, ds)'}{\eta''(u_l)} \\
&= \lim_{u_l \to u_r} \frac{\eta''(u_l) f(u_l) + \eta'(u_l) f'(u_l) - \eta'(u_l) f'(u_l)}{\eta''(u_l)} \\
&= \lim_{u_l \to u_r} f(u_l) = f(u_r).
\end{aligned}$$

Note that this only holds when $\eta''(u)$ exists and is non-zero. For the limit as $u_r \to u_l$, the computation is analogous. □

We note that in some cases it it possible to simplify the fraction in the definition of f_S to avoid cases where we get $0/0$, in this case we do not need to define the function separately for when $u_l = u_r$.

With definition (28) of the entropy conservative numerical flux, the system (25) satisfies a discrete analogue of the conservation of entropy (14).

Theorem 3. *Let $\hat{\mathbf{u}} = (\hat{u}_1, \hat{u}_2, \dots, \hat{u}_N)$ be a solution of (25) for which the entropy $\eta(\hat{\mathbf{u}})$ is convex, and let the flux $f_S(u_l, u_r)$ be entropy conservative as defined by (27). Then, $\hat{\mathbf{u}}$ satisfies the semi-discrete conservation of entropy*

$$\Delta x \mathbf{1}^T \frac{\mathrm{d}\eta(\hat{\mathbf{u}})}{\mathrm{d}t} = 0. \tag{30}$$

Proof. We start by multiplying (25) by $\Delta x \eta'(\hat{\mathbf{u}})^T$ to get

$$\Delta x \eta'(\hat{\mathbf{u}})^T \frac{\mathrm{d}\hat{\mathbf{u}}}{\mathrm{d}t} + \eta'(\hat{\mathbf{u}})^T 2(\mathbf{Q} \circ \mathbf{F})\mathbf{1} = 0.$$

If we now assume continuity in time we can simplify $\eta'(\hat{\mathbf{u}})^T \frac{\mathrm{d}\hat{\mathbf{u}}}{\mathrm{d}t}$:

$$\frac{\mathrm{d}\eta(\hat{\mathbf{u}})^T}{\mathrm{d}\hat{\mathbf{u}}} \frac{\mathrm{d}\hat{\mathbf{u}}}{\mathrm{d}t} = \sum_{j=1}^{N} \left(\frac{\mathrm{d}\eta(\hat{\mathbf{u}})}{\mathrm{d}\hat{\mathbf{u}}} \right)_j^T \frac{\mathrm{d}\hat{u}_j}{\mathrm{d}t} = \sum_{j=1}^{N} \frac{\mathrm{d}\eta(\hat{\mathbf{u}})_j}{\mathrm{d}t} = \mathbf{1}^T \frac{\mathrm{d}\eta(\hat{\mathbf{u}})}{\mathrm{d}t}.$$

We can now use the skew-symmetry of $\mathbf{Q}$ to rewrite the term $\eta'(\hat{\mathbf{u}})^T 2(\mathbf{Q} \circ \mathbf{F})\mathbf{1}$:

$$\sum_{i,j=1}^{N} \eta'(\hat{\mathbf{u}})_i^T 2\mathbf{Q}_{ij} f_S(\hat{u}_i, \hat{u}_j) = \sum_{i,j=1}^{N} (\mathbf{Q}_{ij} - \mathbf{Q}_{ji}) \eta'(\hat{\mathbf{u}})_i^T f_S(\hat{u}_i, \hat{u}_j).$$

From here we can rearrange the indices in the sum and use the symmetry of f_S to obtain

$$\begin{aligned}
\sum_{i,j=1}^{N} (\mathbf{Q}_{ij} - \mathbf{Q}_{ji}) \eta'(\hat{\mathbf{u}})_i^T f_S(\hat{u}_i, \hat{u}_j) &= \sum_{i,j=1}^{N} \mathbf{Q}_{ij} (\eta'(\hat{\mathbf{u}})_i - \eta'(\hat{\mathbf{u}})_j)^T f_S(\hat{u}_i, \hat{u}_j) \\
&= \sum_{i,j=1}^{N} \mathbf{Q}_{ij} (\psi(\hat{\mathbf{u}})_i - \psi(\hat{\mathbf{u}})_j) = \boldsymbol{\psi}^T \mathbf{Q} \mathbf{1} - \mathbf{1}^T \mathbf{Q} \boldsymbol{\psi} = 0,
\end{aligned}$$

where the entropy conservation property of f_S has been used in the second equality. □

The identity (30) is a discrete version of the continuous entropy relation (14). This relationship is a bound on the entropy, which in turn places a bound on the discrete solution $\hat{\mathbf{u}}$. Consequently, Theorem 3 is a statement of nonlinear stability. This is a surprisingly strong stability result that holds for all one-dimensional scalar conservation laws.

4 Numerical experiments

With the implementation of the entropy stable finite volume method, we will look at some applications that benefit from such a method. We are interested in the solution before a shock wave occurs, and we show that the solutions are accurate. For details about how numerical errors are computed, see Appendix 7.3. We start by looking at the transport equation. Since we can solve it analytically, we can make sure the code is functioning properly. The second equation is Burgers' equation. For the two more applicative models we look at a traffic flow model and an enhanced oil recovery model.

The code for this experiment was written in Python and can be found in Appendix 7.2. We integrate in time using SciPy's odeint function for solving systems of differential equations. For each problem, we will check that the solution is accurate by looking at the convergence of the error and then we verify the conservation of entropy. Since the entropy can be any convex function, we look at different entropies for each equation and compare error convergence rate and runtime to compare entropies. For the nonlinear equations that encounter shocks, we show that the conservation of entropy leads to faulty solutions. We remedy this by solving the viscous approximation to the conservation law (11) for a small epsilon to mimic entropy dissipation at a shock.

4.1 Transport equation

The most simple conservation law we can look at is the transport equation since it can be solved analytically. This means that we can make comparisons to our numerical solutions and look for abnormalities or differences. The flux for the transport equation is $f(u) = au$ which moves the solution with rate $f'(u) = a$. Here we see that the rate at which the solution moves is constant, and does not depend on the solution value. This means the solution should move constantly along the x-axis over time in a direction and speed determined by the constant a.

For the analytical solution we use the method of characteristics. The characteristics for this equation have to satisfy $x'(t) = a$ for u to be constant in time, integrating this yields $x(t) = at + x_0$. Since u is by definition constant in time and $x_0 = x - at$, we have that for any $t \geq 0$

$$u(x,t) = u(x_0, 0) = u(x - at, 0) = u_0(x - at).$$

The solution being $u_0(x - at)$ once again tells us that we should expect the initial function for u to move in time with rate a.

The numerical solutions require us to find the entropy conservative flux. Using $\eta(u) = u^2/2$ and Proposition 1, we get

$$\begin{aligned} f_S(u_l, u_r) = \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)} &= \frac{(\eta'(u_l)f(u_l) - \int_0^{u_l} \eta'(s)f'(s)\,ds) - (\eta'(u_r)f(u_r) - \int_0^{u_r} \eta'(s)f'(s)\,ds)}{\eta'(u_l) - \eta'(u_r)} \\ &= \frac{(u_l \cdot au_l - \int_0^{u_l} s \cdot a\,ds) - (u_r \cdot au_r - \int_0^{u_r} s \cdot a\,ds)}{u_l - u_r} \\ &= \frac{a(u_l^2 - u_r^2)}{2(u_l - u_r)} = \frac{a(u_l + u_r)}{2}. \end{aligned}$$

We can use this flux with the implementation from Section 3 to solve for $\hat{\mathbf{u}}$.

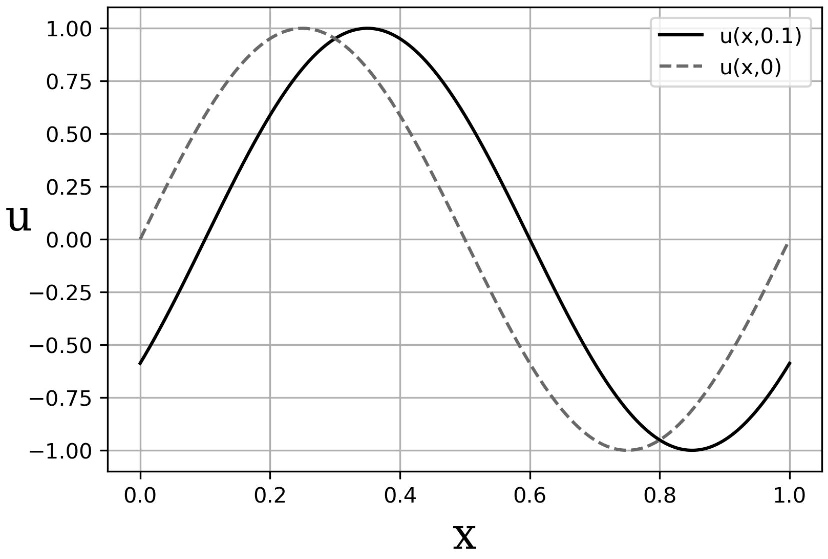


Figure 5: Computed solution for the transport equation at time $t = 0.1$ for the initial condition $u_0(x) = \sin(2\pi x)$ with $a = 1$ and $\Delta x = 1/640$.

The numerical solution pictured in Figure 5 aligns with the expectations that the initial data would be shifted in time.

In order to verify that we have convergence to the analytic solution we look at the convergence of the error. For consistency with later experiments, we compute an estimated order of convergence (EOC) by comparing numerical solutions at successively finer grids. We compare local averages for the different resolutions and then compute the global error using the L^1 norm. By looking at quotients of the errors for succesively finer grids, and taking the logarithm, we can estimate the order of the method. For details, see Appendix 7.3. Figure 6 shows how the error behaves as we increase the value of N. In this case we see clear quadratic convergence, which is further verified by the error convergence in Table 1.

Table 1: Estimated error (EE) and estimated of order of convergence (EOC) for the transport equation using the entropy $|u|^2/2$.

Δx	EE	EOC
0.1	0.1845901149078717	1.981584085700765
0.05	0.04674037489214805	1.9955453104638867
0.025	0.011721230196944266	1.9988959083575761
0.0125	0.0029325509661010662	1.9997204243777793
0.00625	0.0007332798278953945	1.999918508719413
0.003125	0.0001833303121768215	

We have now checked that we have convergence to a solution with an error of size $O(\Delta x^2)$. Next, we look at the entropy for the computed solution to verify that it is conserved. Table 2 shows the difference in entropy as time develops. We notice that the entropy is not quite conserved, it is in fact slightly growing. We suspect that these errors come from the fact that we are not setting the tolerances rtol and atol that put a bound the estimated error in SciPy's odeint function when integrating in time. In Table 3 we see that decreasing the values of rtol and atol (i.e. the relative and the absolute tolerances) decreases the entropy error with the same magnitude. We however note that decreasing the tolerances greatly increases runtime. The runtime was calculated by importing the "time" module and using "t1=time.time()" and "t2=time.time()" before and after solving with odeint, respectively, then evaluating their difference "t2-t1". With the default

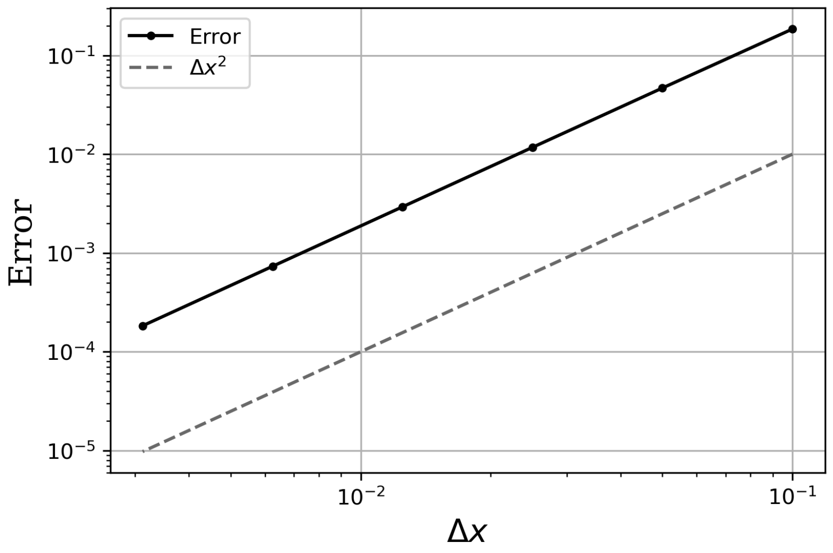


Figure 6: Error of numerical solution to the transport equation with $a = 1$ at time $t = 0.1$ for the initial condition $u_0(x) = \sin(2\pi x)$.

tolerance, a runtime of 14 seconds was achieved, but using 10^{-12} led to a runtime of 2 minutes and 48 seconds. For later experiments that already have a longer runtime, we solve with the default tolerances and take values close to 10^{-6} to be entropy conservative. This is assuming that using lower tolerances will give more accurate entropy errors for all experiments.

Table 2: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function.

Time	$t = 0.02$	$t = 0.04$	$t = 0.06$	$t = 0.08$	$t = 0.1$
Entropy error	$+9.2 \cdot 10^{-7}$	$+3.9 \cdot 10^{-7}$	$+1.2 \cdot 10^{-6}$	$+2.1 \cdot 10^{-6}$	$+2.9 \cdot 10^{-6}$

Table 3: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function with different tolerances in odeint.

Time	$t = 0.02$	$t = 0.04$	$t = 0.06$	$t = 0.08$	$t = 0.1$
Entropy error with rtol=atol=$1.49012 \cdot 10^{-8}$ (Default)	$+9.2 \cdot 10^{-7}$	$+3.9 \cdot 10^{-7}$	$+1.2 \cdot 10^{-6}$	$+2.1 \cdot 10^{-6}$	$+2.9 \cdot 10^{-6}$
Entropy error with rtol=atol=10^{-10}	$+2.5 \cdot 10^{-10}$	$+5.7 \cdot 10^{-9}$	$+4.6 \cdot 10^{-8}$	$+6.6 \cdot 10^{-8}$	$+8.9 \cdot 10^{-8}$
Entropy error with rtol=atol=10^{-12}	$-9.3 \cdot 10^{-11}$	$+3.9 \cdot 10^{-10}$	$-1.2 \cdot 10^{-10}$	$-1.3 \cdot 10^{-10}$	$-1.8 \cdot 10^{-10}$

We would like to compare the solutions when solving with different entropies. To compare with the quadratic entropy, we can use the entropies u^p/p, for even p. We calculate the entropy

Table 4: Estimation of order of the method for the transport equation using the entropy $|u|^4/4$.

Δx	EE	EOC
0.1	0.6170462962807343	1.4722381880496886
0.05	0.22239749668977332	2.4406162938867624
0.025	0.04096671892302706	1.2269610753380995
0.0125	0.017501654047413884	1.6274869483093999
0.00625	0.005664436413557017	1.3956899702072576
0.003125	0.0021528420136449067	

conservative flux for these entropies using Proposition 1 and simplifying the answer. We get that

$$\begin{aligned} f_S(u_l, u_r) = \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)} &= \frac{(u_l^{p-1} \cdot au_l - \int_0^{u_l} s^{p-1} \cdot a\,ds) - (u_r^{p-1} \cdot au_r - \int_0^{u_r} s^{p-1} \cdot a\,ds)}{u_l^{p-1} - u_r^{p-1}} \\ &= \frac{a(p-1)}{p} \cdot \frac{(u_l^p - u_r^p)}{u_l^{p-1} - u_r^{p-1}} \\ &= \frac{a(p-1)}{p} \cdot \frac{\sum_{k=0}^{p-1} u_l^{p-1-k} u_r^k}{\sum_{k=0}^{p-2} u_l^{p-2-k} u_r^k}. \end{aligned}$$

This implementation works well for $p = 2$ as shown above but already in the case $p = 4$ we see numerical artifacts around solutions crossing the x-axis as pictured in Figure 7. Since the difference is that we are now dividing by function values to some power, this numerical artifact could be a result of the higher powers of values close to zero in the denominator. For the results however, this means our numerical solution is faulty and does not converge accurately to a solution. Table 4 shows that the order p does not converge, leaving the solution to be untrustworthy. This tendency remains for larger values of p, and once we reach $p = 10$, the solver breaks down.

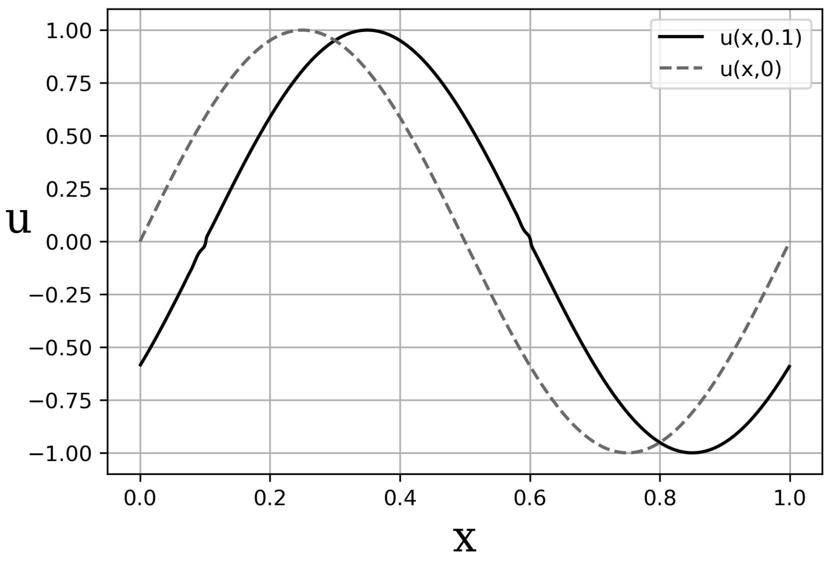


Figure 7: Faulty solution to the transport equation at time $t = 0.1$ for the initial condition $u_0(x) = \sin(2\pi x)$ with $\Delta x = 1/640$.

The numerical artifacts only appear for solutions crossing the x-axis, if we instead look at the initial data given by a Gaussian pulse $u_0(x) = \exp(-100(x - 0.5)^2)$, the numerical artifacts disappear as can be seen in Figure 8. Using this Gaussian pulse, we look for qualitative differences

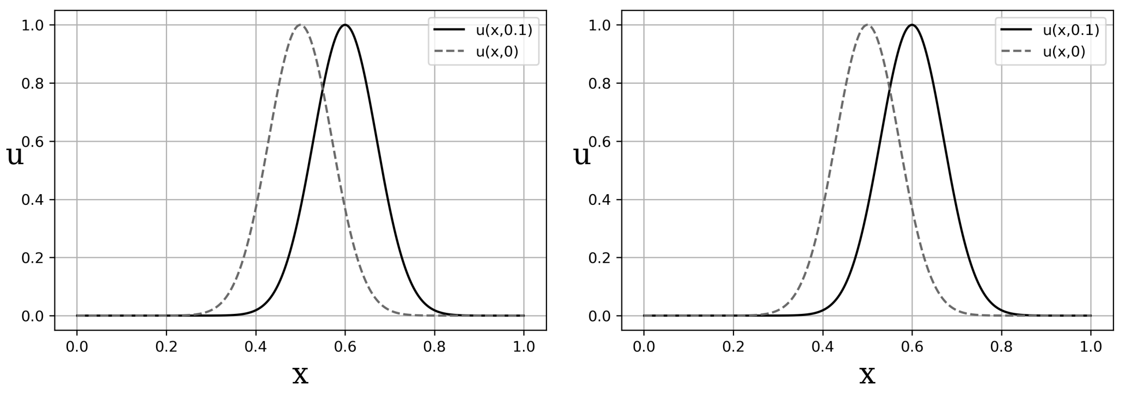


Figure 8: Solution to transport equation for Gaussian pulse with entropy $u^2/2$ (left) and entropy $u^{10}/10$ (right) using $\Delta x = 1/640$.

between the different entropies. For this we look at the EOC for each entropy, presented in Table 5. In Figure 9 we also plot the estimated errors compared for the different entropies. These two measures show rather clearly that using the entropy $u^2/2$ gives the best qualitative result. This, alongside the fact that we get cancellation in the flux construction, making the computations faster and avoiding artifacts, means that the preferred entropy for this problem is $u^2/2$.

Table 5: EOC using the Gaussian pulse initial condition with entropies u^p/p for even p.

Entropy	$u^2/2$	$u^4/4$	$u^6/6$	$u^8/8$	$u^{10}/10$
EOC	1.731314742	2.17766349	2.19345481	2.13603658	2.11207388
	1.61539162	1.37138543	1.33333014	1.30292607	1.23011048
	1.94314112	1.92231257	1.89022958	1.85403756	1.82072800
	1.99020608	1.98481020	1.97174113	1.95150939	1.92527609
	1.99770453	1.99643475	1.99252520	1.98612189	1.97739490

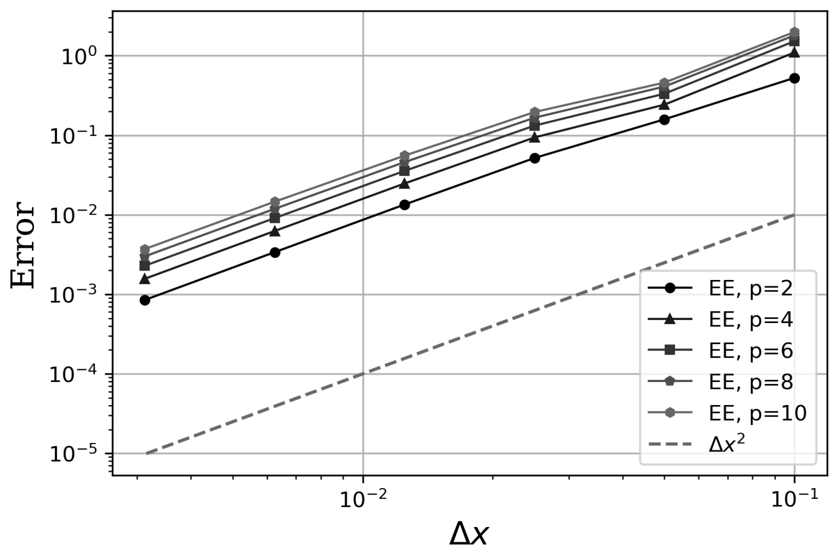


Figure 9: Comparison of the estimated errors using the entropies u^p/p for even p.

4.2 Burgers' equation

Burgers' equation has flux $f(u) = u^2/2$ which moves the solution with rate $f'(u) = u$, so the rate at which the solution moves is exactly equal to its value. To see an example of how this function behaves we calculate the entropy conservative flux for the problem. Using the entropy $\eta(u) = u^2/2$ and Proposition 1, we obtain

$$\begin{aligned} f_S(u_l, u_r) = \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)} &= \frac{(u_l \cdot u_l^2/2 - \int_0^{u_l} s \cdot s\,) - (u_r \cdot u_r^2/2 - \int_0^{u_r} s \cdot s\,)}{u_l - u_r} \\ &= \frac{1}{6} \cdot \frac{(u_l^3 - u_r^3)}{u_l - u_r} \\ &= \frac{u_l^2 + u_l u_r + u_r^2}{6}. \end{aligned}$$

Figure 10 shows an example with the initial function $u_0(x) = 0.5\sin(2\pi x) + 0.5$. We notice that the function moves faster where its value is greater and around the function value 0 we see considerably less movement.

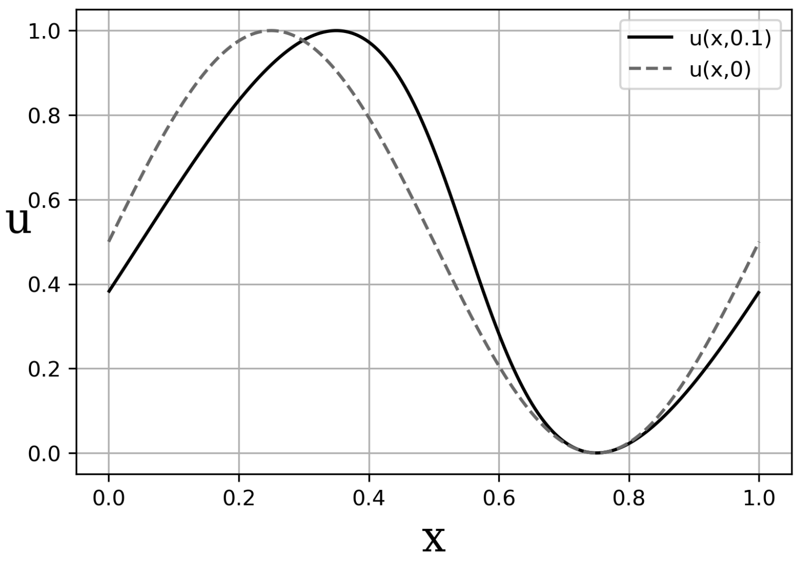


Figure 10: Solution to the inviscid Burgers' equation at time $t = 0.1$ for the initial condition $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ and $\Delta x = 1/640$.

To justify that the solution is accurate, we show the error convergence in Figure 11. We observe that the error is of second order. Additionally, we verify the conservation of entropy using values from Table 6. Since we are not solving with lower tolerances in odeint, we take these values to mean the entropy is conserved.

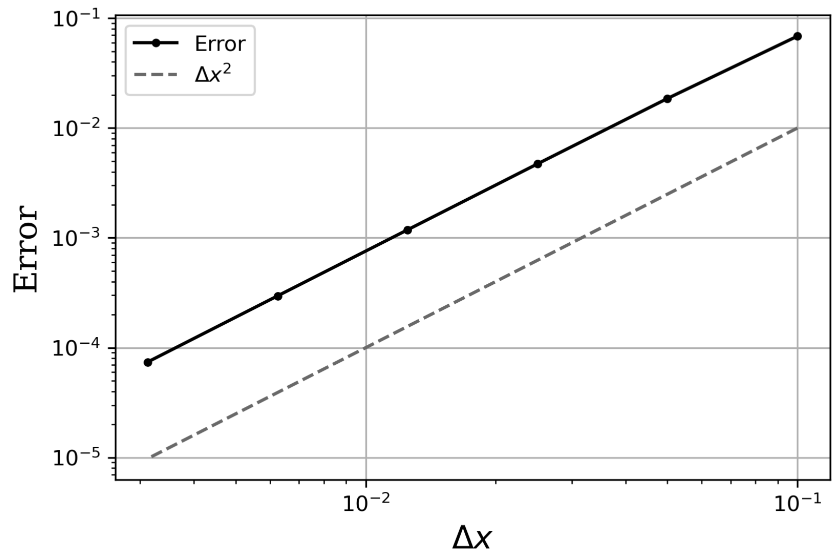


Figure 11: Errors for different step sizes in Burgers' equation for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$.

Table 6: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function.

Time	$t = 0.02$	$t = 0.04$	$t = 0.06$	$t = 0.08$	$t = 0.1$
Entropy error	$+8.4 \cdot 10^{-8}$	$+8.9 \cdot 10^{-8}$	$+1.2 \cdot 10^{-7}$	$+2.1 \cdot 10^{-7}$	$+2.5 \cdot 10^{-7}$

Since the solution does not cross the x-axis, we can compare different entropies for Burgers' equation. To compare these entropies we find the entropy conservative flux using Proposition 1 with the entropies $\eta(u) = u^p/p$ for even p. We get that

$$\begin{aligned} f_S(u_l, u_r) = \frac{\psi(u_l) - \psi(u_r)}{\eta'(u_l) - \eta'(u_r)} &= \frac{(u_l^{p-1} \cdot u_l^2/2 - \int_0^{u_l} s^{p-1} \cdot s\,ds) - (u_r^{p-1} \cdot u_r^2/2 - \int_0^{u_r} s^{p-1} \cdot s\,ds)}{u_l^{p-1} - u_r^{p-1}} \\ &= \frac{(p-1)}{2(p+1)} \cdot \frac{u_l^{p+1} - u_r^{p+1}}{u_l^{p-1} - u_r^{p-1}} \\ &= \frac{(p-1)}{2(p+1)} \cdot \frac{\sum_{k=0}^{p} u_l^{p-k} u_r^k}{\sum_{i=0}^{p-2} u_l^{p-2-i} u_r^i}. \end{aligned}$$

We solve the problem for $p = 10$, and picture the result in Figure 12. We observe that the flux is once again a quotient for values of p higher than 2, with powers depending on p. The higher powers and quotient made these numerical fluxes only slightly more computationally heavy, taking a few more seconds to produce a solution. For $p = 2$ and $p = 10$, respectively, the times were 12.1 and 14.1 seconds. We can also see in Table 7 that higher values of p in the entropy lead to convergence of the error becoming only slightly slower. In Figure 13 we see the estimated errors for each entropy, and we observe that the quadratic entropy barely yields the best result.

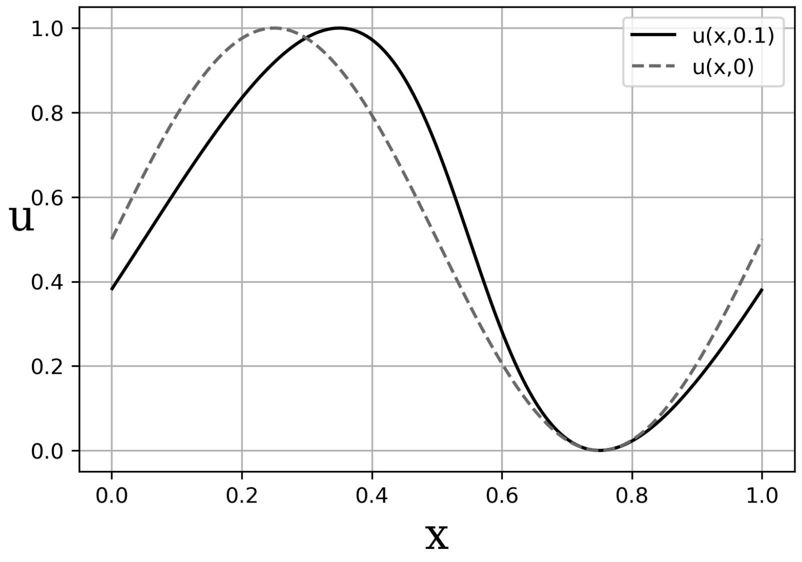


Figure 12: Solution to the inviscid Burgers' equation for the initial condition $u_0(x) = \sin(2\pi x)$ with entropy $u^{10}/10$ and $\Delta x = 1/640$.

Table 7: EOC for the sinusoidal initial function and entropies u^p/p for even p.

Entropy	$u^2/2$	$u^4/4$	$u^6/6$	$u^8/8$	$u^{10}/10$
EOC	1.88442208	1.81051864	1.78429666	1.73254059	1.74270032
	1.97797211	1.95346711	1.90529463	1.83661770	1.76288792
	1.99461485	1.98878286	1.97725855	1.95636119	1.92639236
	1.99860416	1.99720008	1.99479474	1.99049052	1.98364924
	2.00043595	2.00000286	1.99925699	1.99827260	1.99695727

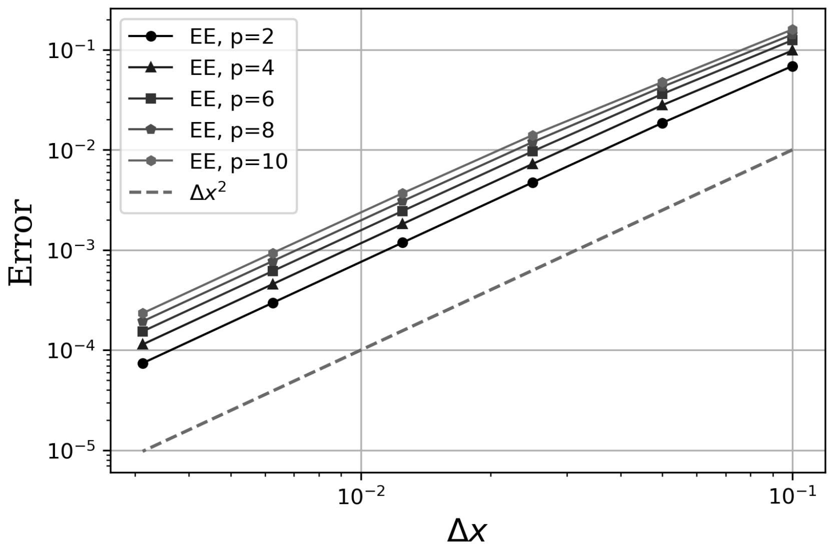


Figure 13: Comparison of the estimated errors using the entropies u^p/p for even p.

An important property of Burgers' equation is that is it nonlinear and thus encounters shocks. This shock breaks our entropy conservative solution since, as we saw in Section 2.3, entropy is dissipative at a shock. Figure 14 shows how the solution begins to oscillate when encountering a shock and Table 8 shows that the entropy is still nearly conserved. In fact, it is slightly growing. To further show that the solution we computed is not accurate, Figure 15 shows that we do not have convergence for the order of the method, showing the error at each step is not predictable.

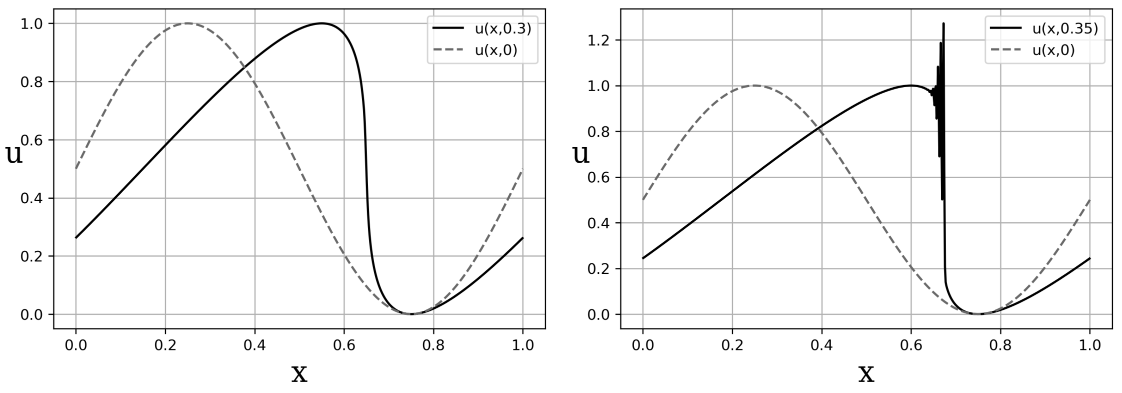


Figure 14: Solution to Burgers' equation for the initial data $u_0(x) = 0.5\sin(2\pi x)+0.5$ with entropy $u^2/2$ and $\Delta x = 1/640$ before (left) and after (right) encountering a shock.

Table 8: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ compared to the initial function.

Time	$t = 0.07$	$t = 0.14$	$t = 0.21$	$t = 0.28$	$t = 0.35$
Entropy error	$+1.9 \cdot 10^{-7}$	$+3.4 \cdot 10^{-7}$	$+3.7 \cdot 10^{-7}$	$+4.2 \cdot 10^{-7}$	$+4.8 \cdot 10^{-7}$

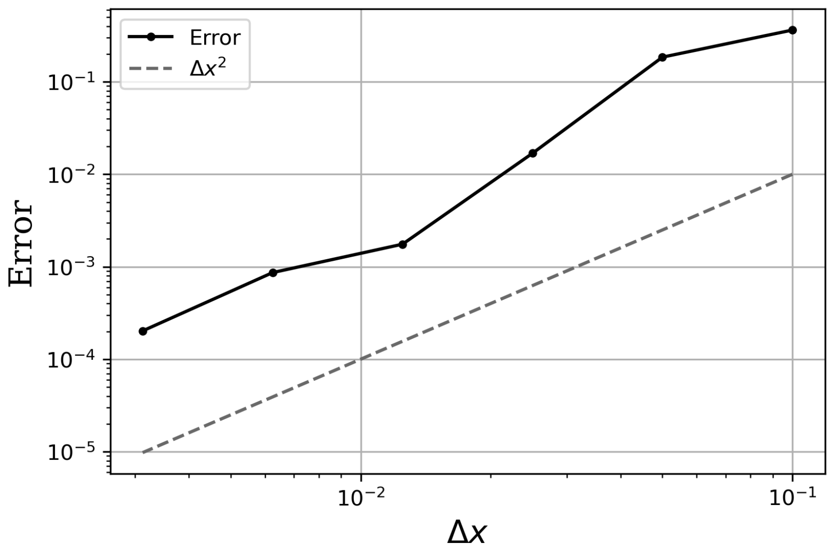


Figure 15: Errors for different step sizes for Burgers' equation using the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ when encountering a shock at time $t = 0.35$.

If we want to see how the solution would behave beyond these shocks, we could look at a numerical solution to the viscous approximation (11). Here it is important to keep in mind that we are leaving the realm of entropy *conservative* solutions, we now get entropy *dissipative* solutions where the amount of dissipation is controlled by ϵ. To obtain our solution to the viscous approximation, we solve the system (26) using odeint. Figure 16 shows the solution to the new viscous equation with two values for ϵ.

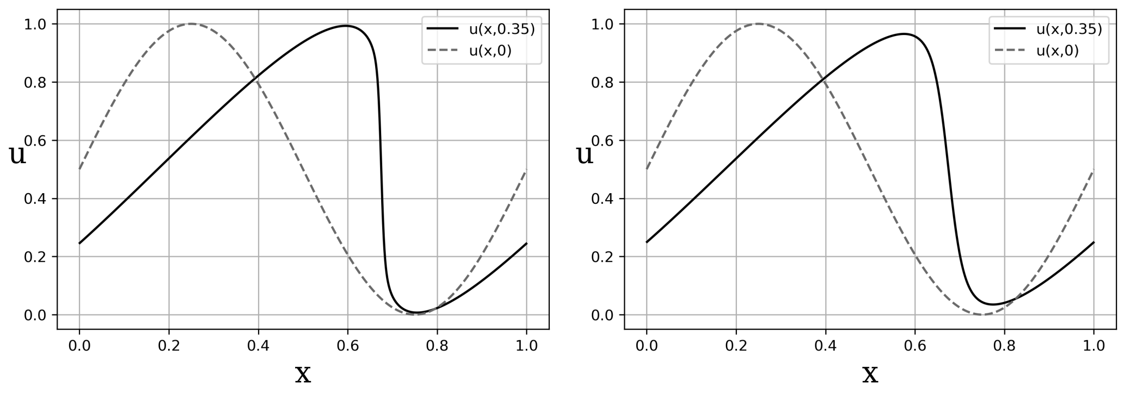


Figure 16: Solutions to the *viscous* Burgers' equation for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ at time $t = 0.35$ with $\Delta x = 1/640$ with $\epsilon = 0.001$ (left) and $\epsilon = 0.005$ (right).

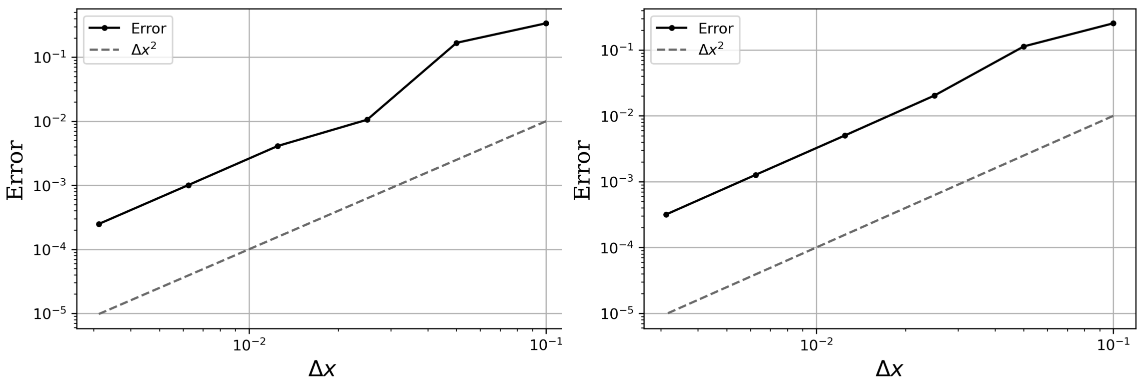


Figure 17: Errors for different step sizes for the *viscous* Burgers' equation for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ at time $t = 0.35$ with $\epsilon = 0.001$ (left) and $\epsilon = 0.005$ (right).

In the solutions to the viscous conservation law in Figure 17, we see that the error is of second order for both values of ϵ. We notice that the solution with $\epsilon = 0.001$ is more similar to the solutions of the non-viscous equation, in the sense it has less dissipation. The increased dissipation in the solution for $\epsilon = 0.005$ could be considered less physically correct, but it is unknown if this is the case.

Table 9 shows the dissipation of entropy as time develops. We know from section 2.3 that entropy is conserved until a shock is encountered for entropy solutions, so we could argue that the solution with the least amount of dissipation is more accurate. This could be further argued by remembering that entropy solutions appeared as $\epsilon \to 0$, so we may want to use the smallest ϵ possible that produces accurate solutions.

Time	$t = 0.07$	$t = 0.14$	$t = 0.21$	$t = 0.28$	$t = 0.35$
Entropy difference, $\epsilon = 0.001$	-0.2	-0.46	-0.75	-1.1	-1.9
Entropy difference, $\epsilon = 0.005$	-1.1	-2.3	-3.5	-5.1	-7.0

Table 9: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function with $\epsilon = 0.001$(top) and $\epsilon = 0.005$(bottom).

4.3 Traffic equation

For some potential real world examples, we can look at the periodic road on which we have some density of cars $u \in [0, 1]$. It would be natural to assume that the speed at which cars travel on this road would be relative to the density, so we attain a simple equation for the speed of the cars $v(u) = v_{max}(1 - u)$. We use the conservation property (2) and that the amount of cars leaving or entering the road at some point is $v(u(x, t)) \cdot u(x, t)$ to get

$$\begin{aligned}\int_a^b -f(u)_x \, dx &= v(u(a, t)) \cdot u(a, t)) - v(u(b, t)) \cdot u(b, t) \\ &= \int_a^b -(v(u)u)_x \, dx.\end{aligned}$$

This gives the flux for this problem, $f(u) = v(u)u = v_{max}(1 - u)u$. This flux is a combination of both the transport equation and Burgers' equation and the solution will move with rate $f'(u) = v_{max}(1 - 2u)$. For simplicity we can set the maximal speed to $v_{max} = 1$ since it is a constant.

We calculate the entropy conservative flux for the entropies $\eta(u) = u^p/p$ for even p using Proposition 1, after simplifications we get that

$$\begin{aligned}f_S(u_l, u_r) &= \frac{(u_l^{p-1} \cdot (u_l - u_l^2) - \int_0^{u_l} s^{p-1} \cdot (1 - 2s)\, ds) - (u_r^{p-1} \cdot (u_r - u_r^2) - \int_0^{u_r} s^{p-1} \cdot (1 - 2s)\, ds)}{u_l^{p-1} - u_r^{p-1}} \\ &= \frac{p-1}{p} \cdot \frac{u_l^p - u_r^p}{u_l^{p-1} - u_r^{p-1}} - \frac{p-1}{p+1} \cdot \frac{u_l^{p+1} - u_r^{p+1}}{u_l^{p-1} - u_r^{p-1}} \\ &= \frac{p-1}{p} \cdot \frac{\sum_{k=0}^{p-1} u_l^{p-1-k} u_r^k}{\sum_{k=0}^{p-2} u_l^{p-2-k} u_r^k} - \frac{p-1}{p+1} \cdot \frac{\sum_{k=0}^{p} u_l^{p-k} u_r^k}{\sum_{k=0}^{p-2} u_l^{p-2-k} u_r^k}.\end{aligned}$$

We notice that this a combination of the two numerical fluxes from the previous equations, similar to how the non-numerical flux was a combination of both previous non-numerical fluxes. We use this numerical flux with our finite volume method to solve the traffic flow equation, Figure 18 shows a solution with $p = 2$. We observe that the density of cars impacts the speed. The high density areas move to the left as the cars behind catch up to the the slower cars in front, and the opposite is observed for the low density areas.

To verify that the solution is accurate we look at the error in Figure 19 that shows second order convergence. Additionally, we see in Table 10 that the entropy is conserved, with an entropy difference of $2.4 \cdot 10^{-8}$ at $t = 0.1$. We consider the entropy conserved since we are solving with higher tolerances, expecting higher entropy errors.

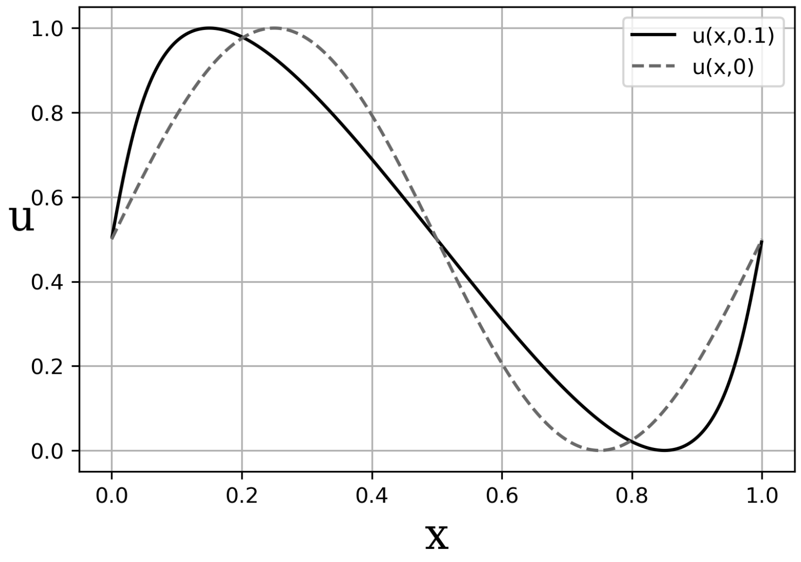


Figure 18: Solution to the traffic flow equation at time $t = 0.1$ for the initial condition $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ and $\Delta x = 1/640$.

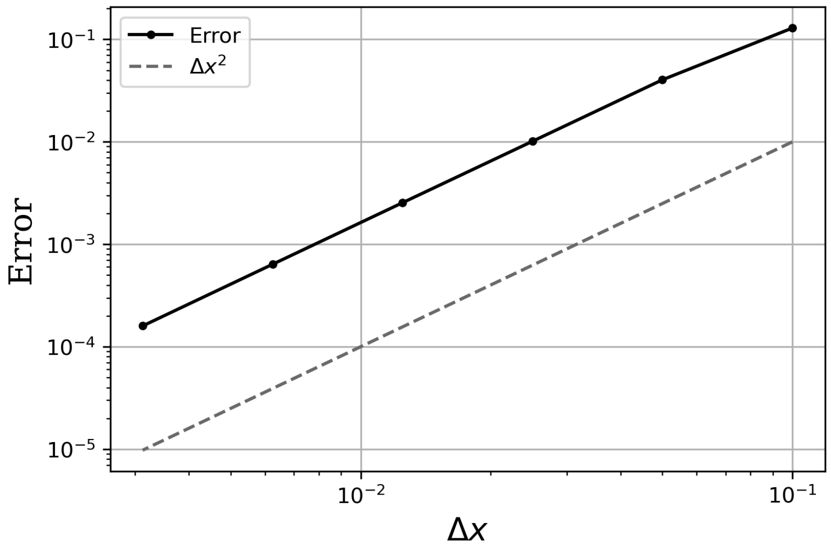


Figure 19: Errors for different step sizes for the traffic flow equation at time $t = 0.1$ for the initial condition $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$.

Table 10: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function.

Time	$t = 0.07$	$t = 0.14$	$t = 0.21$	$t = 0.28$	$t = 0.35$
Entropy error	$2.2 \cdot 10^{-8}$	$2.3 \cdot 10^{-8}$	$2.6 \cdot 10^{-8}$	$2.9 \cdot 10^{-8}$	$2.4 \cdot 10^{-8}$

Using $p = 10$ in the numerical flux, we get the solution pictured in Figure 20. The solution seems similar to the case $p = 2$, but the values in Table 11 show that the estimation of the order becomes substantially worse. When considering runtime, the computations for values of p higher than 2 took longer than the quadratic entropy, the runtimes for $p = 2$ and $p = 4$ were 13.7 and 155 seconds, respectively. In Figure 21 we observe that the error using the quadratic entropy is overall lower.

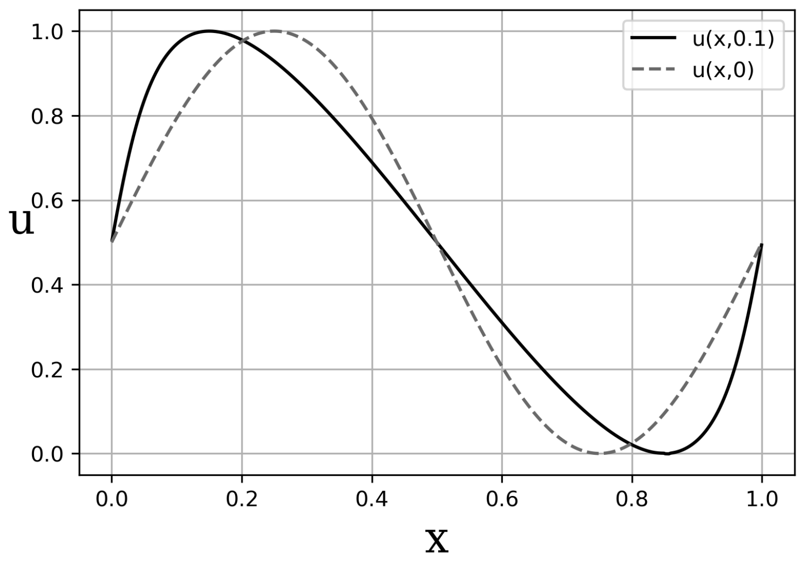


Figure 20: Solution to the traffic flow equation for the initial condition $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^{10}/10$ and $\Delta x = 1/640$.

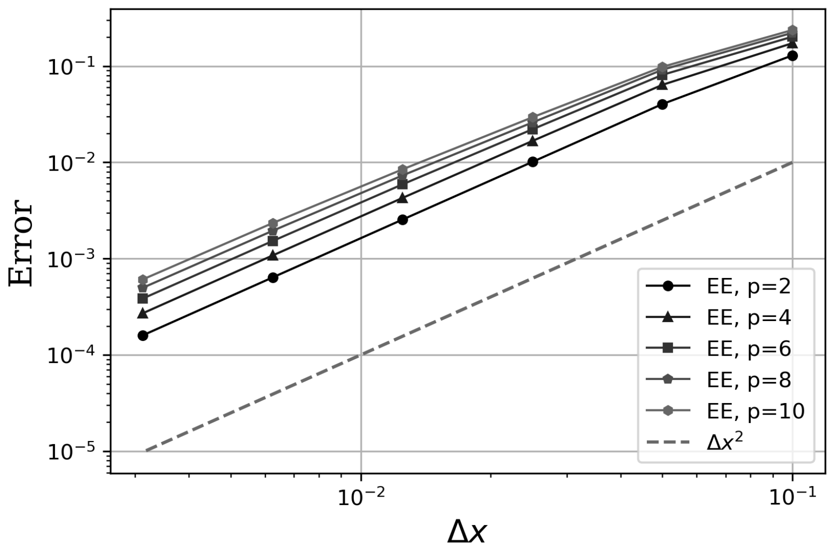


Figure 21: Comparison of the estimated errors using the entropies u^p/p for even p.

Table 11: Estimated value for the order of the method for the sinusoidal initial data and entropies u^p/p for even p.

Entropy	$u^2/2$	$u^4/4$	$u^6/6$	$u^8/8$	$u^{10}/10$
EOC	1.67838417	1.43693215	1.31628776	1.27462376	1.27033781
	1.994357552	1.93642787	1.87737953	1.81909998	1.74038288
	1.99216510	1.96631331	1.90033189	1.82773395	1.79670902
	1.99662164	1.98592642	1.95608631	1.91063596	1.85458798
	1.99936659	1.99589358	1.98517160	1.96785239	1.94473881

In Figure 22 we see the solution encounter a shock, where oscillations appear since the solver conserves entropy in the shock while trying to solve the equation. The solution is clearly faulty, if we would like to know how the solution would behave past this time point, we would need to solve the viscous approximation (26) once again. For this problem it could be interesting to consider what a shock represents. It would make sense if this shock represents a crash. As the faster cars behind catch up to the more densely packed cars in front, they have to make a sudden stop, or crash.

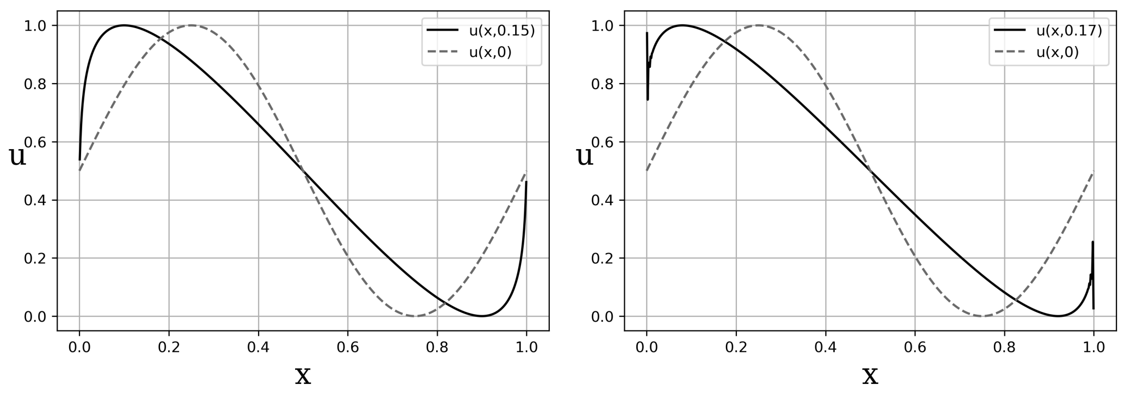


Figure 22: Solutions to the traffic flow equation for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ and $\Delta x = 1/640$ before (left) and after (right) encountering a shock.

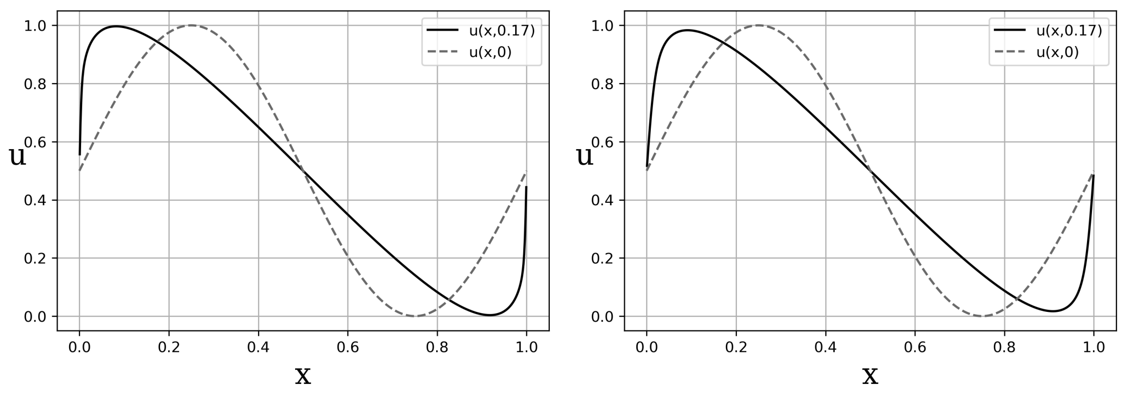


Figure 23: Solutions to the viscous traffic flow equation for the initial data $u_0(x) = 0.5\sin(2\pi x)+0.5$ with entropy $u^2/2$ and $\Delta x = 1/640$ with $\epsilon = 0.001$ (left) and $\epsilon = 0.005$ (right).

In Figure 23 we have plotted solutions to the viscous equation (26). We notice that the viscous term eliminates the oscillations in our computed solution, at the cost of dissipation of the solution.

Unlike the case for Burgers' equation, we have in Figure 24 that the error when $\epsilon = 0.001$ is not of second order, but when we increase ϵ to 0.005 we get more accurate convergence. This means we may want higher values of ϵ even though it introduces more dissipation, as can be seen in Table 12. In Section 4.2, using Burgers' equation, we argued that the solution with least dissipation more accurately mimics the entropy conservative solution. In the case of the traffic equation however, the least dissipative solution is not accurate according to our measures, so we are left with a more dissipative approximate solution.

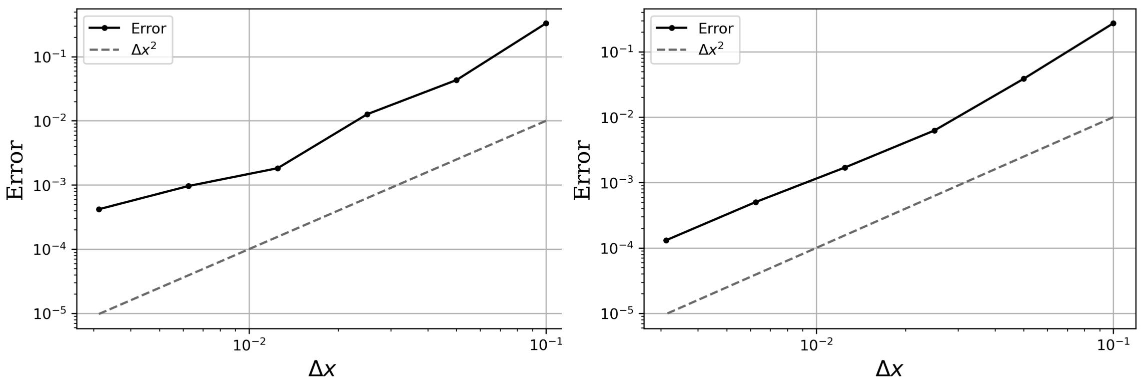


Figure 24: Error comparison for the viscous traffic flow equation for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ and $\epsilon = 0.001$ (left) and $\epsilon = 0.005$ (right).

Table 12: Difference in entropy in the numerical solutions using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function.

Time	$t = 0.07$	$t = 0.14$	$t = 0.21$	$t = 0.28$	$t = 0.35$
Entropy difference, $\epsilon = 0.001$	-0.1	-0.22	-0.36	-0.55	-0.99
Entropy difference, $\epsilon = 0.005$	-0.54	-1.1	-1.8	-2.6	-3.8

4.4 Oil recovery equation

Another scenario where we could apply the notion of entropy solutions is for enhanced oil recovery models. When extracting oil from from inside permeable rocks, only 20-30% of the available oil can be extracted. By injecting water into the rock bed, the heavier water displaces the oil so that it can be extracted. This can be modeled using the flux

$$f(u) = \frac{qu^2}{u^2 + (1-u)^2}, \tag{31}$$

where the constant q is the *total flow rate*, which we can leave to be 1. The solution u denotes the saturation of water in the system, while $1 - u$ denotes the saturation of oil. Details on how to find this flux can be found in [1]. This is the first flux that is not concave or convex, meaning we expect solutions to be a combination of shocks and rarefaction waves. We note that this problem is in applications not solved with periodic boundary conditions.

The flux is rather complex compared to the previous problems, and it turns out that there is no general numerical flux for the entropies u^p/p for even p. For $p = 2$ the entropy conservative flux is found using Proposition 1 and is given by

$$\begin{aligned}
f_S(u_l,u_r) &= \frac{\left(u_l\cdot\frac{u_l^2}{(2u_l^2-2u_l+1)} - \int_0^{u_l} s\cdot\frac{2s(1-s)}{(2s^2-2s+1)^2}\,ds\right)}{u_l-u_r} \\
&\quad - \frac{\left(u_r\cdot\frac{u_r^2}{(2u_r^2-2u_r+1)} - \int_0^{u_r} s\cdot\frac{2s(1-s)}{(2s^2-2s+1)^2}\,ds\right)}{u_l-u_r} \\
&= \frac{\left(\frac{u_l^3}{(2u_l^2-2u_l+1)} - \frac{1}{2}\left(\frac{u_l-1}{2u_l^2-2u_l+1} - \frac{\ln\left(2u_l^2-2u_l+1\right)}{2}\right) + \frac{1}{2}\right)}{u_l-u_r} \\
&\quad - \frac{\left(\frac{u_r^3}{(2u_r^2-2u_r+1)} - \frac{1}{2}\left(\frac{u_r-1}{2u_r^2-2u_r+1} - \frac{\ln\left(2u_r^2-2u_r+1\right)}{2}\right) + \frac{1}{2}\right)}{u_l-u_r} \\
&= \frac{\left(\frac{u_l+1}{2} + \frac{\ln\left(2u_l^2-2u_l+1\right)}{4}\right) - \left(\frac{u_r+1}{2} + \frac{\ln\left(2u_r^2-2u_r+1\right)}{4}\right)}{u_l-u_r} \\
&= \frac{\ln\left(2u_l^2-2u_l+1\right) - \ln\left(2u_r^2-2u_r+1\right)}{4(u_l-u_r)} + \frac{1}{2}.
\end{aligned}$$

Since we cannot simplify the denominator further, we define the numerical flux as

$$f_S(u_l,u_r) = \begin{cases} \frac{u_l^2}{u_l^2+(1-u_l)^2} & \text{if } u_l = u_r, \\ \frac{\ln\left(2u_l^2-2u_l+1\right)-\ln\left(2u_r^2-2u_r+1\right)}{4(u_l-u_r)} + \frac{1}{2} & \text{if } u_l \neq u_r. \end{cases} \tag{32}$$

The solution using the quadratic entropy is pictured in Figure 25 for a short time step $t = 0.03$. The sinusoidal initial function is used so we can compare how the solution behaves compared to the previous problems. In reality, one would use an initial function akin to that in Figure 26, but since it is not periodic, we cannot implement it. To get an idea of how this initial function could behave, we will later look at a modified periodic Gaussian pulse.

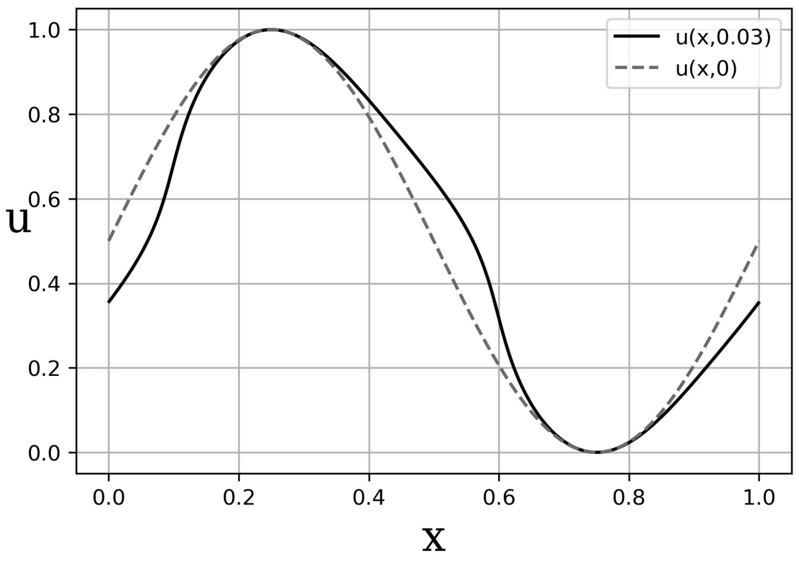


Figure 25: Solution to the oil recovery equation at time $t = 0.03$ for the initial condition $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$ and $\Delta x = 1/640$.

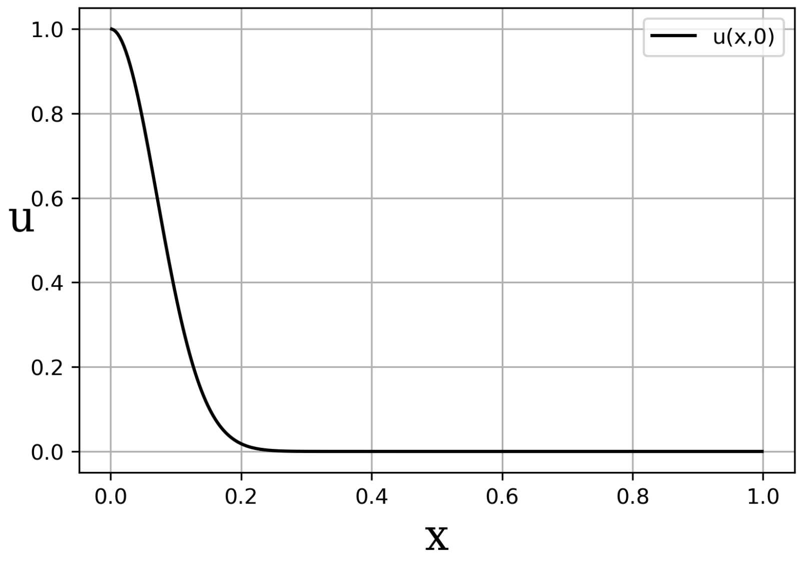


Figure 26: Possible initial condition the oil recovery model equation for real world applications.

We verify that the error for this solution is of second order in Figure 27. To confirm that the solution is in fact entropy conservative, the entropy error is shown in Table 13. We observe that the entropy is not dissipative, with an entropy error of $+8.7 \cdot 10^{-9}$ at $t = 0.03$. We take this value to mean the entropy is conserved since we are solving with default tolerances in odeint.

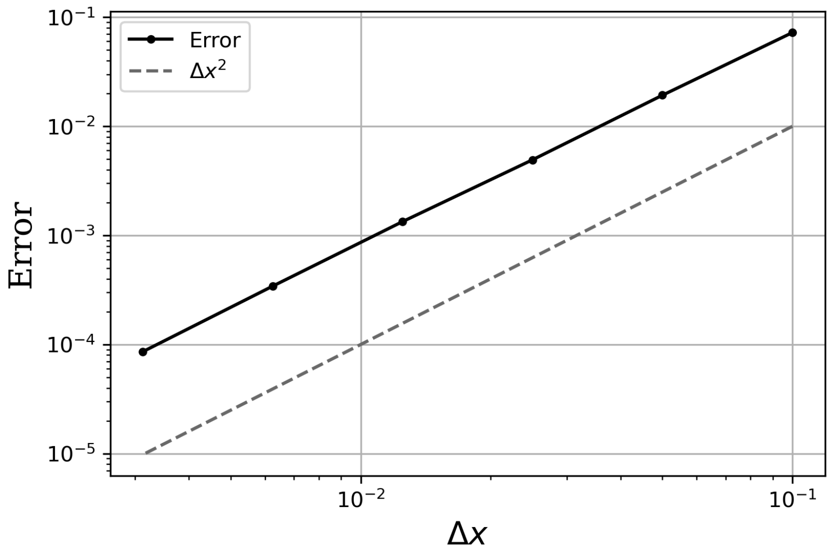


Figure 27: Errors for different step sizes in the oil recovery equation at time $t = 0.03$ for the initial condition $u_0(x) = 0.5\sin(2\pi x) + 0.5$ with entropy $u^2/2$.

Table 13: Difference in entropy in the numerical solution using $\eta(u) = u^2/2$ and $\Delta x = 1/640$ compared to the initial function.

Time	$t = 0.07$	$t = 0.14$	$t = 0.21$	$t = 0.28$	$t = 0.35$
Entropy error	$+7.3 \cdot 10^{-9}$	$+7.4 \cdot 10^{-9}$	$+7.2 \cdot 10^{-9}$	$+7.3 \cdot 10^{-9}$	$+8.7 \cdot 10^{-9}$

We calculate the numerical flux for $\eta(u) = u^4/4$ for comparisons. For this, we use Proposition 1 with many simplifications to get

$$\begin{aligned} f_S(u_l, u_r) &= \frac{\left(u_l^3 \cdot \frac{u_l^2}{(2u_l^2-2u_l+1)} - \int_0^{u_l} s^3 \cdot \frac{2s(1-s)}{(2s^2-2s+1)^2}\, ds\right)}{u_l - u_r} \\ &\quad - \frac{\left(u_r^3 \cdot \frac{u_r^2}{(2u_r^2-2u_r+1)} - \int_0^{u_r} s^3 \cdot \frac{2s(1-s)}{(2s^2-2s+1)^2}\, ds\right)}{u_l - u_r} \\ &= \frac{\left(\frac{u_l^5}{(2u_l^2-2u_l+1)} - \left(\frac{3\arctan(2u_l-1)-u_l^2-2u_l}{4} - \frac{u_l}{8u_l^2-8u_l+4} - \frac{3\pi}{4}\right)\right)}{u_l - u_r} \\ &\quad - \frac{\left(\frac{u_r^5}{(2u_r^2-2u_r+1)} - \left(\frac{3\arctan(2u_r-1)-u_r^2-2u_r}{4} - \frac{u_r}{8u_r^2-8u_r+4} - \frac{3\pi}{4}\right)\right)}{u_l - u_r} \\ &= \frac{3(u_l + u_r + 1)}{4(u_l^2 + u_l u_r + u_r^2)} + \frac{1}{2} - \frac{3\arctan(2u_l - 1) - 3\arctan(2u_r - 1)}{4(u_l^3 - u_r^3)}. \end{aligned}$$

Since we cannot simplify the denominator further, we define a separate case for when $u_l = u_r$,

$$f_S(u_l, u_r) = \begin{cases} \frac{u_l^2}{u_l^2+(1-u_l)^2} & \text{if } u_l = u_r, \\ \frac{3(u_l+u_r+1)}{4(u_l^2+u_l u_r+u_r^2)} + \frac{1}{2} - \frac{3\arctan(2u_l-1)-3\arctan(2u_r-1)}{4(u_l^3-u_r^3)} & \text{if } u_l \neq u_r. \end{cases}$$

The complexity of this flux made some computations very heavy. For the sinusoidal used for the previous flux, the solver did not produce a solution in a reasonable amount of time. Instead, the function $u_0(x) = 0.4\sin(2\pi x) + 0.5$ will be used used for comparisons to avoid function values close to 0, that seem to be causing issues. In Figure 28 we see that both fluxes produce representable solutions to the problem, but qualitative differences can be seen in the estimated error comparison in Figure 29. The error for both fluxes converges to second order, but in the case of quadratic entropy, the error values are lower overall, but the order is overestimated for a few steps.

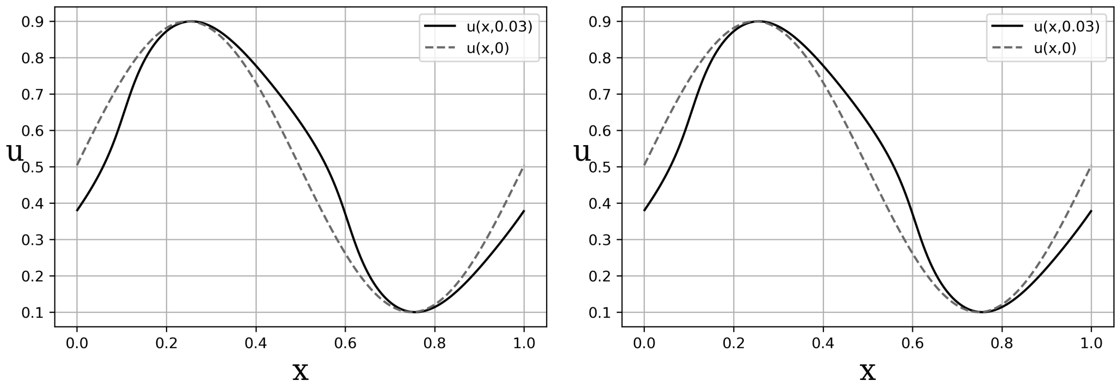


Figure 28: Solution to the oil recovery model using entropies $\eta(u) = u^2/2$ (left) and $\eta(u) = u^4/4$ (right) for the initial data $u_0(x) = 0.4\sin(2\pi x) + 0.5$ at time $t = 0.03$ with $\Delta x = 1/640$.

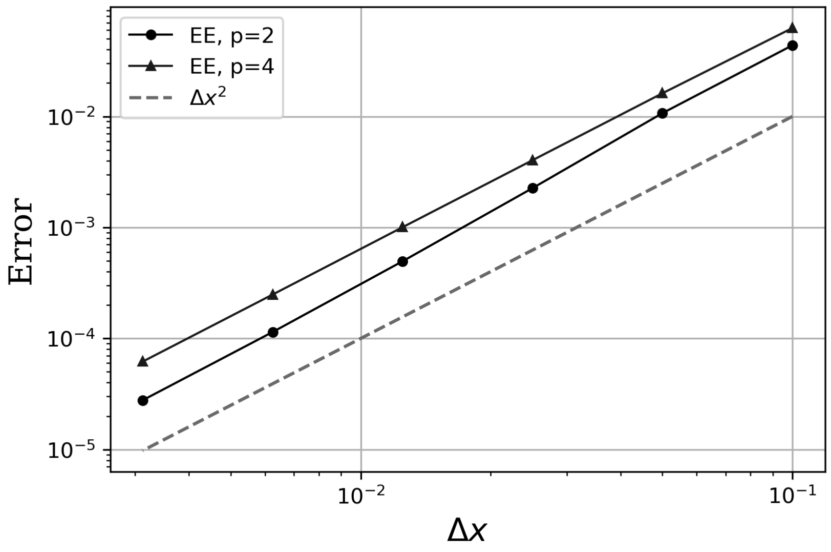


Figure 29: Error in the solution to the oil recovery model using entropies $\eta(u) = u^2/2$ and $\eta(u) = u^4/4$ for the initial data $u_0(x) = 0.4\sin(2\pi x) + 0.5$ at time $t = 0.03$.

Similarly to the two previous equations, the oil recovery model encounters shocks where the solution breaks down. In Figure 30 we see oscillations where the solution breaks down due to the lack of entropy dissipation. To get a more representable solution we need to solve the viscous approximation (26).

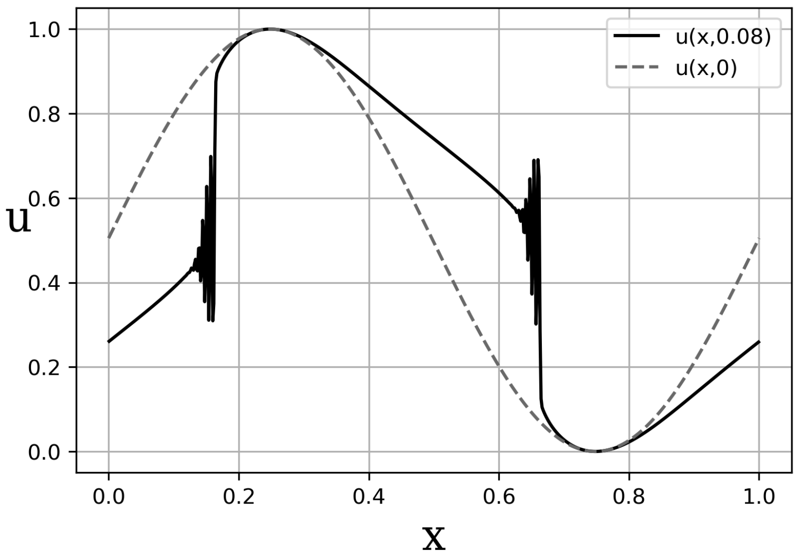


Figure 30: Solution to the oil recovery model using entropy $\eta(u) = u^2/2$ for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ after encountering a shock at time $t = 0.08$ with $\Delta x = 1/640$.

For the viscous solution we need to use a big enough ϵ so that the error is of second order while keeping it small enough to mimic the entropy conservative solution. In Figure 31, we see that the solutions to the viscous equation handle the shocks better, both solutions are free from oscillations and are producing seemingly representable solutions. The solution with smaller ϵ is less smooth at the shock but follows the entropy conservative solution better outside of the shock.

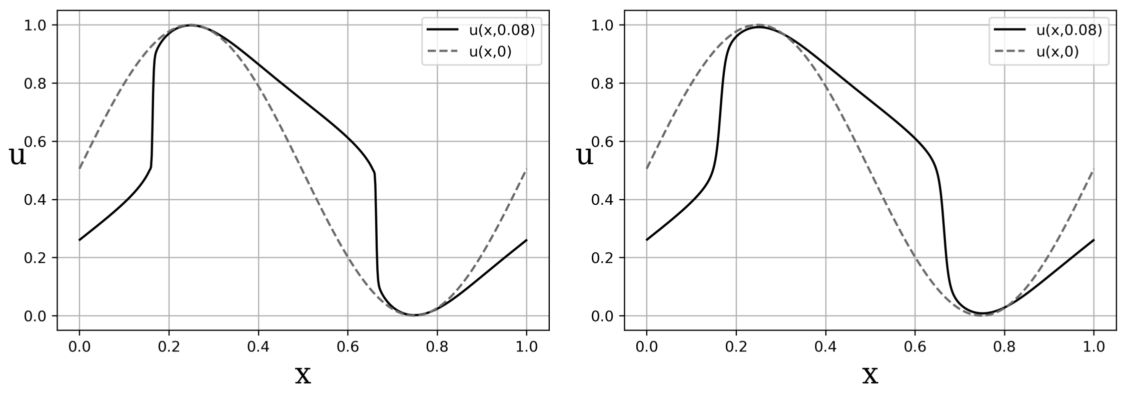


Figure 31: Solutions to the viscous oil recovery model using entropy $\eta(u) = u^2/2$ for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ at time $t = 0.08$ with $\epsilon = 0.001$ (left) and $\epsilon = 0.005$ (right) using $\Delta x = 1/640$.

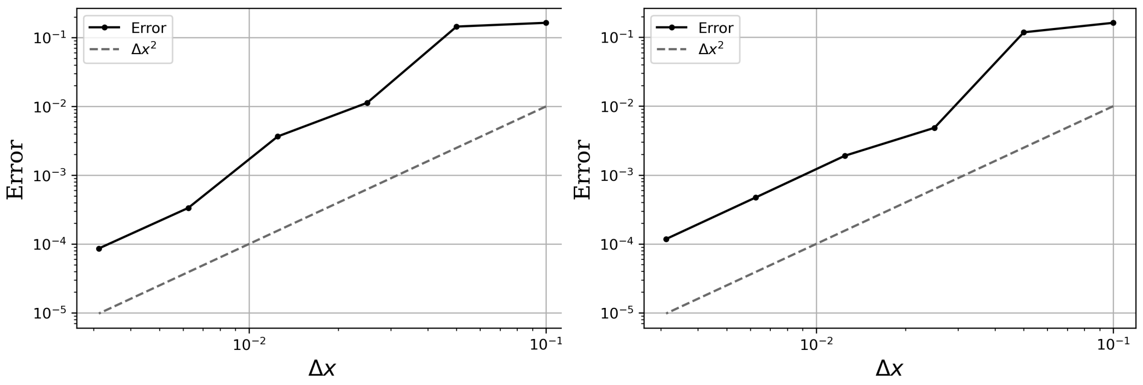


Figure 32: Errors in the solutions to the viscous oil recovery model using entropy $\eta(u) = u^2/2$ for the initial data $u_0(x) = 0.5\sin(2\pi x) + 0.5$ at time $t = 0.08$ with $\epsilon = 0.001$ (left) and $\epsilon = 0.005$ (right).

To verify accuracy of the solutions we show the estimation of the order in Figure 32. In the less dissipative solution, the error does not reach second order until the last step and since we do not have information for the following stepsizes, it is hard to trust that it would be of second order. We observe better convergence in the solution with more dissipation. We notice that the solution with less dissipation sees smaller errors for the smaller stepsizes, but the convergence as seen in Figure 32 is not as convincing as the more dissipative solution. As for the entropy difference seen in Table 14, we notice that the solution using a smaller ϵ is considerably less dissipative, more accurately mimicking the entropy conservative solution. Since solutions here encounter shocks fairly quickly, the dissipation does not have a major impact on the solution before the shock. If one was to look at a longer time span, a smaller ϵ would still be beneficial for the solution prior to a shock to have the least amount of dissipation as possible.

Table 14: Difference in entropy in the numerical solutions using $\eta(u) = u^2/2$ compared to the initial function.

Time	$t = 0.07$	$t = 0.14$	$t = 0.21$	$t = 0.28$	$t = 0.35$
Entropy difference, $\epsilon = 0.001$	-0.05	-0.1	-0.18	-0.29	-0.58
Entropy difference, $\epsilon = 0.005$	-0.25	-0.53	-0.85	-1.27	-1.85

As stated earlier, this problem is most often not treated with periodic boundary conditions. To mimic the initial condition in Figure 26, we use half of a Gaussian pulse that is completed with a linear function to make it periodic. The reason the initial condition is fixed at 1 shortly before $x = 0.5$ is to simulate water being added to the system. Because of this, we focus on the right hand side of the function to get an idea of how the original initial function would develop. In Figure 33, the solution encounters a shock after only 0.03 units of time. To understand how the solution behaves we therefore look at the solution to the viscous problem 26 instead, with $\epsilon = 0.005$. In Figure 34 we observe that as time progresses, the saturation of water u increases as the water displaces the oil. The shock that was formed at $t = 0.03$ continues to move and is what simulates the water displacing oil. Adding water to the system means the saturation at $x = 0.5$ stays the same, similarly to how this would work in applications.

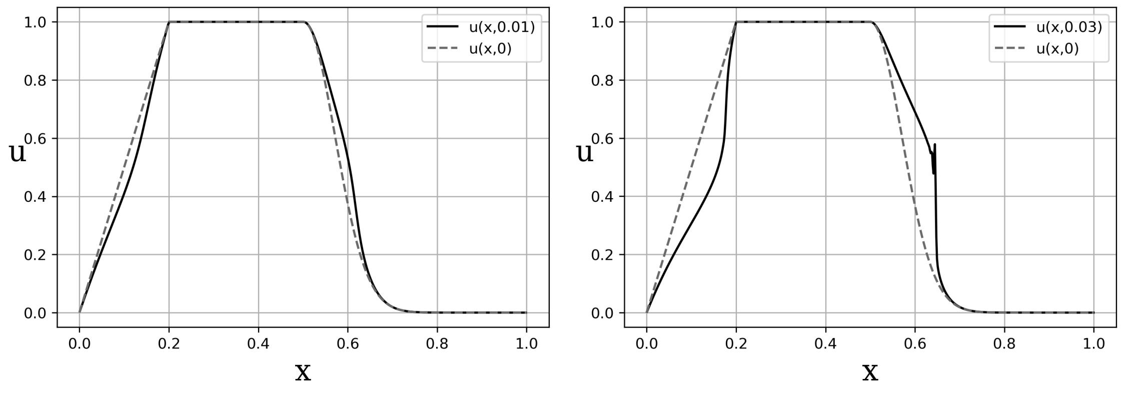


Figure 33: Solutions to the oil recovery model using entropy $\eta(u) = u^2/2$ at times $t = 0.01$ and $t = 0.03$ with $\Delta x = 1/640$.

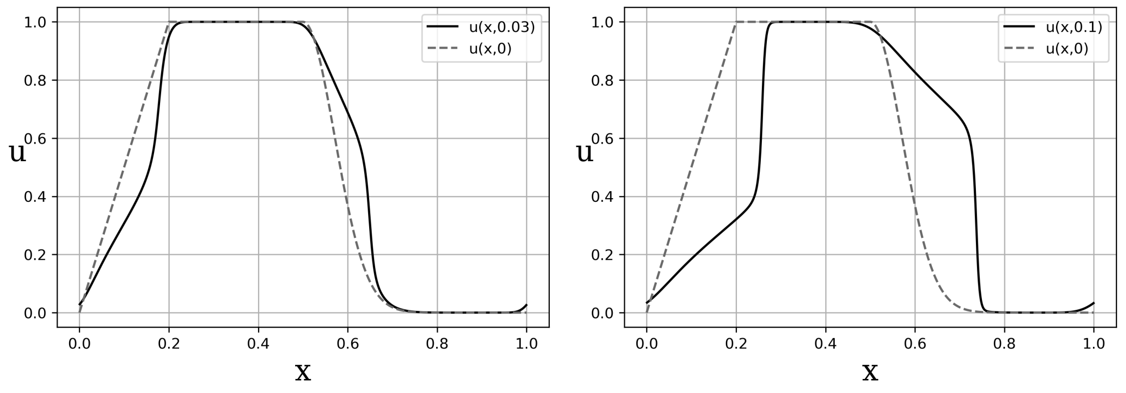


Figure 34: Solutions to the viscous oil recovery model using entropy $\eta(u) = u^2/2$ at times $t = 0.03$ and $t = 0.1$ with $\epsilon = 0.005$ and $\Delta x = 1/640$.

5 Summary and conclusion

We introduced one dimensional scalar conservation laws with periodic boundary equations as equations describing phenomena with a conserved quantity in the system. In Section 2 we presented theory of scalar conservation laws where we observed using characteristics that smooth solutions cease to exist after a certain amount of time for nonlinear fluxes. We then introduced weak solutions that allow discontinuities, but we found that weak solutions lack uniqueness. We looked at a few entropy conditions that guarantee uniqueness in the solution, but focused on a generalization that was derived from the viscous approximation to the scalar conservation law. We found that the entropy inequality led to several important properties for entropy solutions that we could mimic in the discrete case.

The discontinuities in our system made it appropriate to create a numerical method using the finite volume method. This was since the finite volume method approximates cell averages, meaning we have Riemann problems at the boundaries of each cell. By constructing an entropy stable numerical flux, we could prove a discrete conservation of entropy that guaranteed stability for our finite volume method. To implement this we set up a semi-discretization of the problem and solved the system of equations using SciPy's odeint function.

We looked at several examples of conservation laws that could be solved with an entropy stable finite volume method. By analyzing the errors and measuring the entropy we could determine the accuracy of our solutions and compare different entropies for the same problem. We showed that the conservation of entropy produces faulty solutions when a shock is encountered. To get an idea of how the solutions behaved past this shock, we solved the viscous approximation of the conservation law. These were solved using different amounts of dissipation and compared to each other and the entropy conservative solution.

From the numerical experiments we found that the entropy conservative finite volume method worked well for solutions until a shock was encountered. We noticed second order convergence and conservation of entropy in all solutions leading up to a shock. Comparing the entropies u^p/p, for even p, we saw that the errors increased with p, and the estimated order of convergence became slower as p increased. In the test cases that we looked at, the quadratic entropy with $p = 2$ often gave the best qualitative results, and was thus used when looking at entropy dissipative solutions. These dissipative solutions were needed because of the entropy conservative solutions encountering shocks, where oscillations would form and the solutions would break down due to the conservation of entropy. When looking at the entropy dissipative solutions, it was unclear how much dissipation was the optimal amount. We saw that the solutions with less dissipation clearly mimicked the entropy conservative solutions better, however, their accuracy was not always satisfactory. Because of this, the preferred amount of dissipation differed in each problem, but a general result would be the lowest dissipation that creates an accurate solution. Overall, the results from using entropy stable finite volume method were positive, the method produced accurate solutions and was easy to adjust to each problem.

6 Further studies

In this thesis we focused on one-dimensional scalar conservation laws. As a topic for further studies, one could instead look at systems of conservation laws that arise in areas such as fluid dynamics, with an example being the Euler equations, which describe the behavior of a compressible fluid in the absence of viscous forces.

Another possible area of study is introducing more dimensions. It turns out that theory for one-dimensional scalar conservation laws cannot generally be used in the multi-dimensional case. In the one-dimensional case we saw that shocks and discontinuities could appear. For the multi-dimensional case, however, the situation is more complicated. On top of the previous singularities, we would have to study the appearance of more complicated wave movement, a few examples are presented in [7], such as vorticity waves, focusing waves and concentration waves.

We studied the homogeneous conservation law where the right hand side is zero. Physical examples are, however, rarely discontinuous, and discontinuities are smoothed out by natural viscosity or heat transfer. Viscosity is, as we saw in this thesis, added as a second derivative and controlled by some small constant ϵ. If natural viscosity appears, the amount would be controlled by the problem. An example for this are the Navier-Stokes equations in fluid dynamics that has natural viscosity in the form of a laplacian.

7 Appendix

7.1 Derivation of the Rankine-Hugoniot jump condition

If we have a shock wave $\sigma(t)$ and we take two points $a < \sigma(t)$ and $b > \sigma(t)$ in the conservation identity (2), we can split the identity into two integrals

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_a^b u(x,t)dx = \lim_{\epsilon\to 0^+}\frac{\mathrm{d}}{\mathrm{d}t}\int_a^{\sigma(t)-\epsilon} u(x,t)dx + \lim_{\epsilon\to 0^+}\frac{\mathrm{d}}{\mathrm{d}t}\int_{\sigma(t)+\epsilon}^b u(x,t)dx = f(u(a,t)) - f(u(b,t)).$$

We denote $\sigma(t)-\epsilon = \sigma^-$, $\sigma(t)+\epsilon = \sigma^+$ for small ϵ and use Leibnitz' rule for differentiation under the integral sign on the new integrals to get

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_a^{\sigma^-} u(x,t)dx + \frac{\mathrm{d}}{\mathrm{d}t}\int_{\sigma^+}^b u(x,t)dx = \int_a^{\sigma^-}\frac{\partial u}{\partial t}(x,t)dx + u(\sigma^-,t\frac{\mathrm{d}\sigma(t)}{\mathrm{d}t} + \int_{\sigma^-}^b \frac{\partial u}{\partial t}(x,t)dx - u(\sigma^+,t)\frac{\mathrm{d}\sigma(t)}{\mathrm{d}t}.$$

Finally, we let $a \to \sigma^-$ and $b \to \sigma^+$ so that the integrals disappear, and we are left with

$$u(\sigma^-,t)\frac{\mathrm{d}\sigma(t)}{\mathrm{d}t} - u(\sigma^+,t)\frac{\mathrm{d}\sigma(t)}{\mathrm{d}t} = f(u(\sigma^-,t)) - f(u(\sigma^+,t))$$

$$\implies \frac{\mathrm{d}\sigma(t)}{\mathrm{d}t} = \frac{f(u^-(t)) - f(u^+(t))}{u^-(t) - u^+(t)}, \tag{33}$$

where $u^+(t) = u(\sigma^+,t)$ and $u^-(t) = u(\sigma^-(t)$.

7.2 Implementation into code

To start the implementation, I use the following imports.

```python
import numpy as np
import scipy.sparse as sps
import scipy.integrate as scint
from scipy.integrate import odeint
from matplotlib.pyplot import *
import time

```

The first part of the implementation is the entropy conservative flux. It is defined as a function of two variables and is changed according to the entropy and flux being used. The definition for this function is given in each section about the experiments. The next part of the implementation is to set up the right hand side of the equations

$$\frac{d\hat{u}}{dt} = -2(\mathbf{Q}\circ\mathbf{F})\mathbf{1},$$

$$\frac{d\hat{u}}{dt} = -2(\mathbf{Q}\circ\mathbf{F})\mathbf{1} - \epsilon\mathbf{K}\hat{u},$$

which will be used to solve the equations. The function F is for the original conservation law, and FV is for the viscous approximation.

```python
def F(u, t, N, dx, fs, eps):  #RHS for the conservation law.
    Q = sps.diags([-1*np.ones(N-1), 0, np.ones(N-1)],[-1,0,1]).toarray()
    Q[0,-1] = -1
    Q[-1,0] = 1
    F = np.zeros((N, N))
    for r in range(N):
        for c in range(N):
            if Q[r,c]!=0:
                F[r,c] = fs(u[r], u[c])
    QF1 = -(Q*F)@np.ones(N)
```

```
            return QF1*N

    def FV(u, t, N, dx, fs, eps):  #RHS for the viscous approximation to the conservation law.
        Q = sps.diags([-1*np.ones(N-1), 0, np.ones(N-1)],[-1,0,1]).toarray()
        Q[0,-1] = -1
        Q[-1,0] = 1
        F = np.zeros((N, N))
        for r in range(N):
            for c in range(N):
                if Q[r,c]!=0:
                    F[r,c] = fs(u[r], u[c])
        K = sps.diags([-1*np.ones(N-1), 2*np.ones(N), -1*np.ones(N-1)],[-1,0,1]).toarray()
        K[0,-1] = -1
        K[-1,0] = -1
        K = K/dx
        QF1 = -(Q*F)@np.ones(N)-eps*K@u
        return QF1/dx

```

We also need to define the initial data, some examples that will be used in the discussion are:

```
    u0 = lambda x: 0.5*np.sin(2*np.pi*x)+0.5
    u0 = lambda x: np.e**(-100*(x-0.5)**2)

```

When solving the equation, we solve for many stepsizes so that we can compare errors. Since the finite volume method approximates the average function value over an interval, we take the average function value of the initial data in line 19.

```
1       eps = 5*1.e-3
2       N = np.array([10, 20, 40, 80, 160, 320, 640]) #x
3       M = 10                                        #t
4   
5       xend = 1
6       tend = 0.1
7   
8       DX = xend/N       #List of stepsizes
9   
10      U = []            #List of solutions
11  
12      for n in N:
13          x = np.linspace(0, xend, n+1)
14          dx = xend/n
15          xplot = (x+dx/2)[:-1]
16  
17          t = np.linspace(0, tend, M)
18  
19          u0h = [1/dx*scint.quad(u0, x[i], x[i+1])[0] for i in range(n)] #Average value of initial data.
20  
21          t1 = time.time()
22  
23          u, dic = odeint(F, u0h, t, args = (n, fs, eps), full_output = True)
24  
25          t2 = time.time()
26  
27          print("N =", n, "Time =", t2-t1)
28  
29          sol = u[-1]
30  
31          U.append(sol)               #Store solution at latest time.
32  
33          Plot_Sol_vs_IC(sol, u0, xplot, xend) #Plot solution at latest time vs initial data. (See function definition below)
34  
```

With the solutions computed, the next step is to estimate the order of the error.

```
    def Estimate_p(N, U):
        Err = []              #List with errors computed.
        P = []                #List converging to p.

        I = []
        l = N[0]

        for k in range(len(U)):
            USum = []
            m = round(N[k]/N[0])
            for i in range(l):
                usum = sum(U[k][m*i:m*(i+1)])/m
                USum.append(usum)
            I.append(USum)

        for k in range(len(I)-1):
            err = 0
            for i in range(l):
                err += abs(I[k][i]-I[k+1][i])     #Global error (sum of errors in the intervals) for each solution compared to the next solution.
            Err.append(err)         #Store in error list

        for i in range(len(Err)-1):
            P.append(np.log2(Err[i]/Err[i+1]))  #Store approximated value for p.

        return Err, P

```

We use some plotting functions to present the results in an appropriate manner.

```
    def Plot_Sol_vs_IC(sol, u0, xplot, xend): # Plot last time step with initial data.
        figure(dpi=300)
        plot(xplot, sol, label="u(x,{})".format(tend), c="k")
        u00 = [u0(x) for x in np.linspace(0,xend,641)]
        plot(np.linspace(0, xend, 641), u00, label="u(x,0)", ls="--", c="dimgrey")
        grid()
        xlabel("x", fontsize=20, fontname="serif")
        ylabel("u", rotation=0, fontsize=20, fontname="serif")
        legend()

    def Plot_Error_loglog(DX, Err): # Plot error in a loglog plot.
        pdx = DX[:-1]
        figure(dpi=300)
        loglog(pdx, Err, label="Error", marker='.', c="k")
        loglog(pdx, pdx**2, label="$\Delta x^2$", ls="--", c="dimgrey")
        xlabel("$\Delta x$", fontsize=15, fontname="serif")
        ylabel("Error", fontsize=15, fontname="serif")
        grid()
        legend()

```

7.3 Error computations and estimation of p

When working with numerical approximations of exact solutions, the error will depend on some small parameter $h = \Delta x$. A numerical method of order p will for a numerical solution $\tilde{u}$ and exact solution u satisfy

$$|\tilde{u} - u| \leq Ch^p.$$

where C is a constant independent of h. Using this we can note that if the error $\tilde{u} - u$ depends smoothly on h then

$$\tilde{u} - u = Ch^p + O(h^{p+1}).$$

To estimate this value p and verify that our solution is correct and converges as we refine the grid, we need to compare the error in several step sizes. Since we are working with local averages where each $\hat{u}_i$ is an approximation to the average value in the ith interval, we need to compare our solutions in the same intervals for each resolution of the grid as stated in [8]. This can be quite difficult because the intervals half in size every time we half Δx, meaning it is easy to misalign the comparison and introduce unwanted errors. Since we half these intervals each time we half Δx, we need to average the approximated value over the new intervals so we can compare to the value over the whole interval.

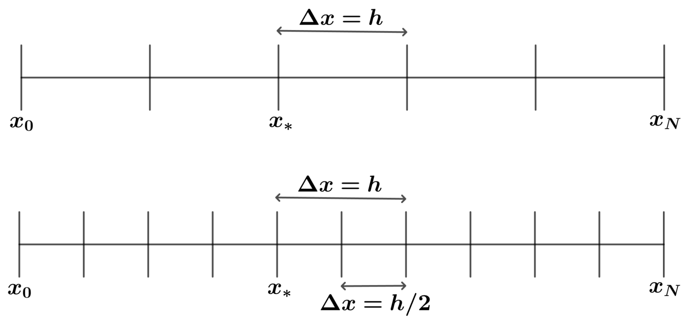


Figure 35: Halving of the intervals in which we compute the average of the solution.

In Figure 35 we would have to take the average of the approximated value in two intervals of size $h/2$ to compare with a point in the interval of size h. Similarly, if we were to half the interval size again to some $\Delta x = h/4$ we would have to compare the average of the approximated value in 4 intervals to one interval of size h or two intervals of size $h/2$.

The approximation of local averages $\hat{u}_i$ over the intervals $[x_i - h/2, x_i + h/2]$ for some $\Delta x = h$ are given by

$$\hat{u}_i(h) = \frac{1}{h}\int_{x_i-h/2}^{x_i+h/2} u(x)\,dx + C(x_i)h^p + O(h^{p+1}).$$

For the initial grid with smallest step size $\Delta x = h$, we let x_* be the left edge of the ith interval in which we are computing the error. The average value in the ith interval is then

$$I_h = \frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + C(x_* + h/2)h^p + O(h^{p+1}).$$

If we now half the step size, we need to take an average of the approximated average in the intervals $[x_*, x_* + h/2]$ and $[x_* + h/2, x_* + h]$

$$\begin{aligned} I_{h/2} = \frac{1}{2}\bigg[&\frac{2}{h}\int_{x_*}^{x_*+h/2} u(x)\,dx + C(x_* + h/4)(h/2)^p + O(h^{p+1}) \\ &+ \frac{2}{h}\int_{x_*+h/2}^{x_*+h} u(x)\,dx + C(x_* + 3h/4)(h/2)^p + O(h^{p+1})\bigg] \\ =& \frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + C(x_* + h/2)(h/2)^p + O(h^{p+1}). \end{aligned}$$

Similarly, if we half the step size again we sum up and average four intervals we get

$$I_{h/4} = \frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + C(x_* + h/2)(h/4)^p + O(h^{p+1}).$$

To estimate the value of p, we look at the following quotient using I_h, $I_{h/2}$, $I_{h/4}$ and for simplicity denote $C := C(x_* + h/2)$

$$\begin{aligned}\frac{I_h - I_{h/2}}{I_{h/2} - I_{h/4}} &= \frac{\frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + Ch^p + O(h^{p+1}) - \frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + C(h/2)^p + O(h^{p+1})}{\frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + C(h/2)^p + O(h^{p+1}) - \frac{1}{h}\int_{x_*}^{x_*+h} u(x)\,dx + C(h/4)^p + O(h^{p+1})} \\ &= \frac{Ch^p - C(h/2)^p + O(h^{p+1})}{C(h/2)^p - C(h/4)^p + O(h^{p+1})} = \frac{1 - 2^{-p} + O(h)}{2^{-p} - 2^{-2p} + O(h)} = 2^p + O(h).\end{aligned}$$

From here we can take the base 2 logarithm of the quotient to obtain a value that should equal to p and that is how we get our estimate. This can of course be done for any small h so it also works if we continue with $h/8$, $h/16$, ... The most accurate results are obtained when the grid is finer, so it is beneficial to look at the sequence of quotients and check that it converges to p

$$\log_2\left(\frac{I_h - I_{h/2}}{I_{h/2} - I_{h/4}}\right), \quad \log_2\left(\frac{I_{h/2} - I_{h/4}}{I_{h/4} - I_{h/8}}\right), \quad \log_2\left(\frac{I_{h/4} - I_{h/8}}{I_{h/8} - I_{h/16}}\right), \quad \log_2\left(\frac{I_{h/8} - I_{h/16}}{I_{h/16} - I_{h/32}}\right), \quad \dots . \tag{34}$$

References

[1] Siddhartha Mishra. Scalar conservation laws, 2014.

[2] Peter D Lax. *Hyperbolic systems of conservation laws and the mathematical theory of shock waves*. SIAM, 1973.

[3] Jesse Chan. Entropy stable reduced order modeling of nonlinear conservation laws. *Journal of Computational Physics*, 423:109789, 2020.

[4] Randall J LeVeque and Randall J Leveque. *Numerical methods for conservation laws*, volume 214. Springer, 1992.

[5] John Nash. Continuity of solutions of parabolic and elliptic equations. *American Journal of Mathematics*, 80(4):931–954, 1958.

[6] Siddhartha Mishra, U Fjordholm, and R Abgrall. Numerical methods for conservation laws and related equations. *Lecture notes for Numerical Methods for Partial Differential Equations*, 57:58, 2019.

[7] Gui-Qiang G Chen. Multidimensional conservation laws: overview, problems, and perspective. *Nonlinear conservation laws and applications*, pages 23–72, 2011.

[8] Olof Runborg. Verifying numerical convergence rates, 2012.

Bachelor’s Theses in Mathematical Sciences 2023:K21
ISSN 1654-6229

LUNFNA-4049-2023

Numerical Analysis
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lu.se/