Burgers' Equation

April 21,2019

Abstract

This paper covers some topics about Burgers equation. Starting from a traffic flow model, Burgers equation emerges. It is then solved by Cole-Hopf transformation before giving asymptotic results of the exact solution. Finally, the Input-to-State Stability(ISS) properties of Burgers equation are analyzed, and numerical experiments concludes this course project.

A simple model of traffic flow

Consider a fleet of cars driving on a highway, and let $\rho(x,t), v(x,t)$ denote the density and the velocity respectively. For any interval $[a,b]$ on the road, the total number of cars in this segment is $\int_a^b \rho(x,t)dx$. Therefore, from time t to $t+\Delta t$, we have

$$\begin{aligned}\int_a^b \rho(x,t+\Delta t)dx - \int_a^b \rho(x,t)dx &= \int_t^{t+\Delta t} \rho(a,t)v(a,t)dt - \int_t^{t+\Delta t} \rho(b,t)v(b,t)dt \\ &= \rho(a,t_1)v(a,t_1)\Delta t - \rho(b,t_2)v(b,t_2)\Delta t. \quad \text{(Mean Value Theorem)}\end{aligned}$$

Therefore,

$$\int_a^b \frac{\rho(x,t+\Delta t)-\rho(x,t)}{\Delta t}dx = \rho(a,t_1)v(a,t_1) - \rho(b,t_2)v(b,t_2).$$

Let $\Delta t \to 0$, we have

$$\int_a^b \rho_t dx = \rho(a,t)v(a,t) - \rho(b,t)v(b,t) = -\int_a^b (\rho v)_x dx,$$

i.e.

$$\int_a^b [\rho_t + (\rho v)_x]dx = 0.$$

As a,b are arbitrary, we have

$$\rho_t + (\rho v)_x = 0. \qquad \text{(continuity equation)}$$

Assume that $v = v_{\max}(1-\frac{\rho}{\rho_{\max}}) - C\frac{\rho_x}{\rho}$, where $\rho_{\max}$ is the maximum density, $v_{\max}$ is the maximum velocity, and C is a positive constant(this term is in order to reflect the fact that drivers would reduce their speed for an increasing density ahead), then

$$\rho_t + v_{\max}\rho_x - 2\frac{\rho_{\max}}{v_{\max}}\rho\rho_x - C\rho_{xx} = 0.$$

Make

$$\begin{cases} \widehat{x} = -x + v_{\max}t \\ \widehat{t} = t \end{cases},$$

then

$$\begin{aligned} \rho_t(\widehat{x},\widehat{t}) &= \rho_{\widehat{x}} \cdot (v_{\max}) + \rho_{\widehat{t}} \\ \rho_x(\widehat{x},\widehat{t}) &= \rho_{\widehat{x}} \cdot (-1) = -\rho_{\widehat{x}} \\ \rho_{xx}(\widehat{x},\widehat{t}) &= \rho_{\widehat{x}\widehat{x}} \end{aligned} ,$$

hence, $\rho_t + 2\frac{v_{\max}}{\rho_{\max}}\rho\rho_{\widehat{x}} - C\rho_{\widehat{x}\widehat{x}} = 0$
Let

$$\begin{cases} \rho' = \frac{2}{\rho_{\max}}\rho \\ x' = \frac{1}{v_{\max}}\widehat{x} \\ t' = \widehat{t} \end{cases} ,$$

then

$$\rho'_{t'} + \rho'\rho'_{x'} = \mu\rho'_{x'x'}, \quad \text{where } \mu = \frac{C}{v_{\max}^2}.$$

When $C = 0$, we get the inviscid Burgers equation $\rho'_{t'} + \rho'\rho'_{x'} = 0$. For more information about Burgers' equation and traffic flow model, refer to [1, 2].

A simplification of Navier-Stokes Equation

Consider the incompressible Navier-Stokes equation, see [3],

$$\rho(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v}) = -\nabla p + \upsilon \Delta \mathbf{v} + \mathbf{F}$$

where ρ is the density, **v** is the velocity, p is the pressure, υ is the fluid viscosity and **F** is an external force.

Assume that there are no external forces, and the pressure term is negligible, then the Navier-Stokes equation for 1D problem becomes the viscid Burgers equation, shown as

$$u_t + uu_x = \mu u_{xx}, \tag{1}$$

where $\mu = \frac{\upsilon}{\rho}$ is the kinematic viscosity.

When the viscosity μ is zero, then equation (1) becomes the following inviscid Burgers equation, shown as

$$u_t + uu_x = 0. \tag{2}$$

Cole-Hopf transformation and the exact solution

Consider now the viscid Burgers equation (1) with initial condition

$$u(x, 0) = u_0(x). \tag{3}$$

Define the Cole-Hopf transformation(see [4]), as

$$u = -2\mu\frac{w_x}{w}, \tag{4}$$

then

$$u_t = \frac{2\mu(w_t w_x - w w_{xt})}{w^2}, uu_x = \frac{4\mu w_x(w w_{xx} - w_x^2)}{w^3}$$

and

$$\mu u_{xx} = -\frac{2\mu^2(2w_x^3 - 3w w_{xx} w_x + w^2 w_{xxx})}{w^3}.$$

Substituting these expressions into equation (1), we get

$$w_x(w_t - \mu w_{xx}) = w(w_{xt} - \mu w_{xxx}) = w(w_t - \mu w_{xx})_x .$$

Therefore, if $w(x,t)$ solves the heat equation

$$w_t - \mu w_{xx} = 0, \tag{5}$$

then $u(x,t)$ given by transformation (4) solves the viscid Burgers equation (1).

As for the initial condition (3), the new variable $w(x,t)$ must satisfy

$$-2\mu\frac{w_x(x,0)}{w(x,0)} = u_0(x)$$

i.e.

$$\frac{dw(x,0)}{w(x,0)} = -\frac{1}{2\mu}u_0(x)dx.$$

Integrating form both sides from 0 to x,

$$\int_0^x \frac{dw(s,0)}{w(s,0)} = -\frac{1}{2\mu}\int_0^x u_0(s)ds, \tag{6}$$

hence,

$$w(x,0) = e^{-\frac{1}{2\mu}\int_0^x u_0(s)ds}. \tag{7}$$

Note that the lower limit of the integral in (6) can be changed from 0 to any other convenient value. As a result of the transformation, we only need to deal with the heat diffusion problem, satisfying equation (5) and (7). With the heat kernel expression

$$E(x,t) = \frac{1}{\sqrt{4\pi\mu t}}e^{-\frac{x^2}{4\mu t}},$$

we have

$$\begin{aligned} w(x,t) &= \int_{-\infty}^{+\infty} w(\xi,0)E(x-\xi,t)d\xi \\ &= \frac{1}{\sqrt{4\pi\mu t}}\int_{-\infty}^{+\infty} e^{-\frac{1}{2\mu}\int_0^\xi u_0(\eta)d\eta - \frac{(x-\xi)^2}{4\mu t}}\, d\xi \\ &=: \frac{1}{\sqrt{4\pi\mu t}}\int_{-\infty}^{+\infty} e^{-\frac{1}{2\mu}G}d\xi, \end{aligned}$$

where

$$G(x,t,\xi) = \int_0^\xi u_0(\eta)d\eta + \frac{(x-\xi)^2}{2t}.$$

It then follows that

$$\frac{\partial w}{\partial x} = -\frac{1}{2\mu\sqrt{4\pi\mu t}}\int_{-\infty}^{+\infty}\frac{x-\xi}{t}e^{-\frac{1}{2\mu}G}\, d\xi.$$

and the exact solution of Burgers initial value problem is obtained, which is

$$u(x,t) = \frac{\int_{-\infty}^{+\infty}\frac{x-\xi}{t}e^{-\frac{1}{2\mu}G(x,t,\xi)}\, d\xi}{\int_{-\infty}^{+\infty} e^{-\frac{1}{2\mu}G(x,t,\xi)}\, d\xi}. \tag{8}$$

Asymptotic behaviour of the solution

Suppose $G(x,t,\xi)$ has only one stationary point ξ_0, with Taylor's formula, we expand $G(x,t,\xi)$ and $\frac{x-\xi}{t}$ at ξ_0 respectively,

$$\begin{aligned} G(x,t,\xi) &= G(x,t,\xi_0) + G'(x,t,\xi_0)(\xi-\xi_0) + \frac{1}{2}G''(x,t,\xi)(\xi-\xi_0)^2 + \cdots \\ &= G(x,t,\xi_0) + \frac{1}{2}G''(x,t,\xi)(\xi-\xi_0)^2 + \cdots \\ \frac{x-\xi}{t} &= \frac{x-\xi_0}{t} + (-\frac{1}{t})(\xi-\xi_0) + \cdots \end{aligned}$$

Thus, asymptotically,

$$\int_{-\infty}^{+\infty} \frac{x-\xi}{t} e^{-\frac{1}{2\mu}G(x,t,\xi)}\, d\xi \sim \frac{x-\xi_0}{t} e^{-\frac{1}{2\mu}G(x,t,\xi_0)} \int_{-\infty}^{+\infty} e^{-\frac{1}{4\mu}G''(x,t,\xi)(\xi-\xi_0)^2} d\xi$$

The symbol $\sim$ means in a limiting or asymptotic sense. Let $z^2 = \frac{1}{4\mu}|G''(x,t,\xi)|(\xi-\xi_0)^2$, then $dz = \sqrt{\frac{|G''(x,t,\xi)|(\xi-\xi_0)^2}{4\mu}} d\xi$, thus,

$$\int_{-\infty}^{+\infty} \frac{x-\xi}{t} e^{-\frac{1}{2\mu}G(x,t,\xi)}\, d\xi \sim \frac{x-\xi_0}{t} e^{-\frac{1}{2\mu}G(x,t,\xi_0)} \sqrt{\frac{4\mu\pi}{|G''(x,t,\xi)|}}$$

$$\int_{-\infty}^{+\infty} G(x,t,\xi)\, d\xi \sim e^{-\frac{1}{2\mu}G(x,t,\xi_0)} \sqrt{\frac{4\mu\pi}{|G''(x,t,\xi)|}}.$$

Therefore, we conclude that

$$u(x,t) \sim \frac{x-\xi_0}{t} \quad \text{as} \quad \mu \to 0.$$

For this asymptotic estimation, refer to Method of Steepest Descent in [5, 6].
As $G'(x,t,\xi_0) = u_0(\xi_0) - \frac{x-\xi_0}{t} = 0$, then the asymptotic solution may be rewritten as

$$u(x,t) = u_0(\xi_0), \quad x = \xi_0 + u_0(\xi_0)t. \tag{9}$$

Notice that this is the same solution we obtained with method of characteristics for the inviscid Burgers' equation. This solution is only valid before t_b, where t_b denotes the breaking time of a gradient catastrophe. In some cases, (9) gives multivalued solution after a sufficient time, and discontinuities or a shock wave solution must be introduced.(Weak solution, shock wave solution, Rankine Hugoniot(jump) condition, entropy condition). When this stage is reached, The explanation is that there are two stationary points for $G(x,t,\xi)$, denoted by ξ_1 and ξ_2 with $\xi_1 > \xi_2$ respectively. Then

$$u(x,t) \sim \frac{\frac{x-\xi_1}{t}|G''(x,t,\xi_1)|^{-\frac{1}{2}} e^{-\frac{1}{2\mu}G(x,t,\xi_1)} + \frac{x-\xi_2}{t}|G''(x,t,\xi_2)|^{-\frac{1}{2}} e^{-\frac{1}{2\mu}G(x,t,\xi_2)}}{|G''(x,t,\xi_1)|^{-\frac{1}{2}} e^{-\frac{1}{2\mu}G(x,t,\xi_1)} + |G''(x,t,\xi_2)|^{-\frac{1}{2}} e^{-\frac{1}{2\mu}G(x,t,\xi_2)}} \tag{10}$$

When $G(x,t,\xi_1) \neq G(x,t,\xi_2)$, one or the other term of $e^{-\frac{1}{2\mu}G(x,t,\xi)}$ is overwhelmingly large when $\mu \to 0$. Suppose when $G(x,t,\xi_1) < G(x,t,\xi_2)$, we have

$$u(x,t) \sim \frac{x-\xi_1}{t},$$

then when $G(x,t,\xi_1) > G(x,t,\xi_2)$,

$$u(x,t) \sim \frac{x-\xi_2}{t}.$$

The criteria of $G(x,t,\xi_1) \lesseqgtr G(x,t,\xi_2)$ will determine the asymptotic solution for given (x,t). The changeover will occur at those (x,t) for which $G(x,t,\xi_1) = G(x,t,\xi_2)$, i.e.

$$\int_0^{\xi_1} u_0(\eta)d\eta + \frac{(x-\xi_1)^2}{2t} = \int_0^{\xi_2} u_0(\eta)d\eta + \frac{(x-\xi_2)^2}{2t}.$$

As $G'(x,t,\xi_1) = 0$ and $G'(x,t,\xi_2) = 0$, this condition may be rewritten as

$$\int_{\xi_1}^{\xi_2} u_0(\eta)d\eta = \frac{1}{2}(u_0(\xi_1) + u_0(\xi_2))(\xi_1 - \xi_2).$$

This is exactly the shock fitting rule for determining the shock path, which can be derived from the jump condition analytically, refer to [7].
Geometrically, it has a more intuitive explanation based on the equal area principle, refer to Figure 1 and Figure 2.

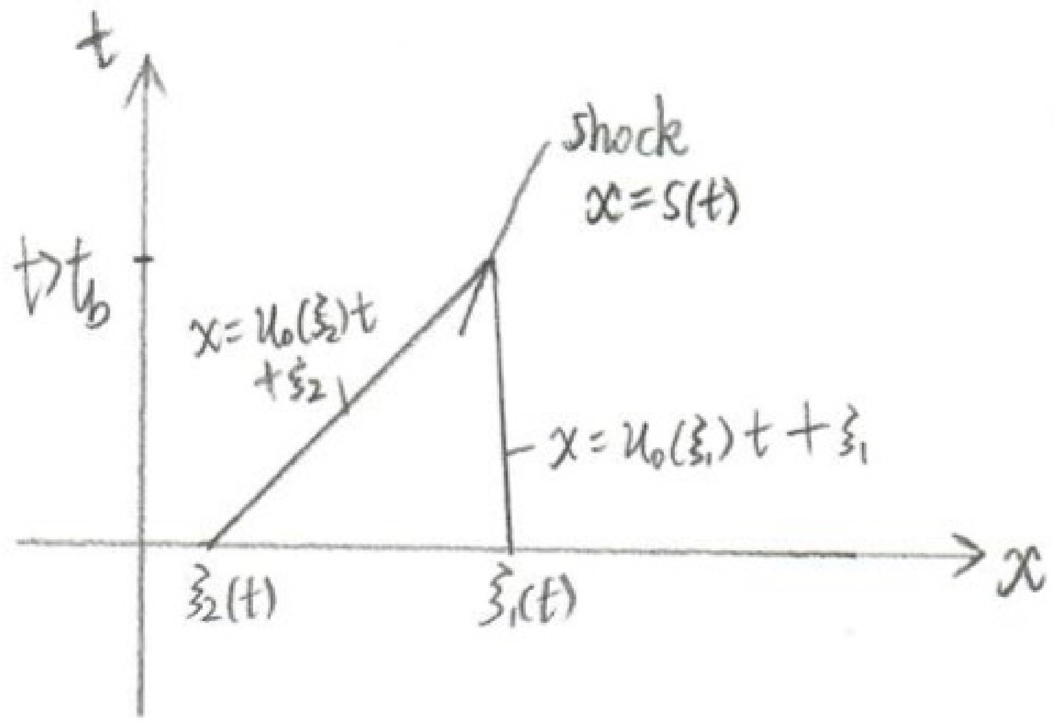

Figure 1: Two characteristics intersect on the shock path

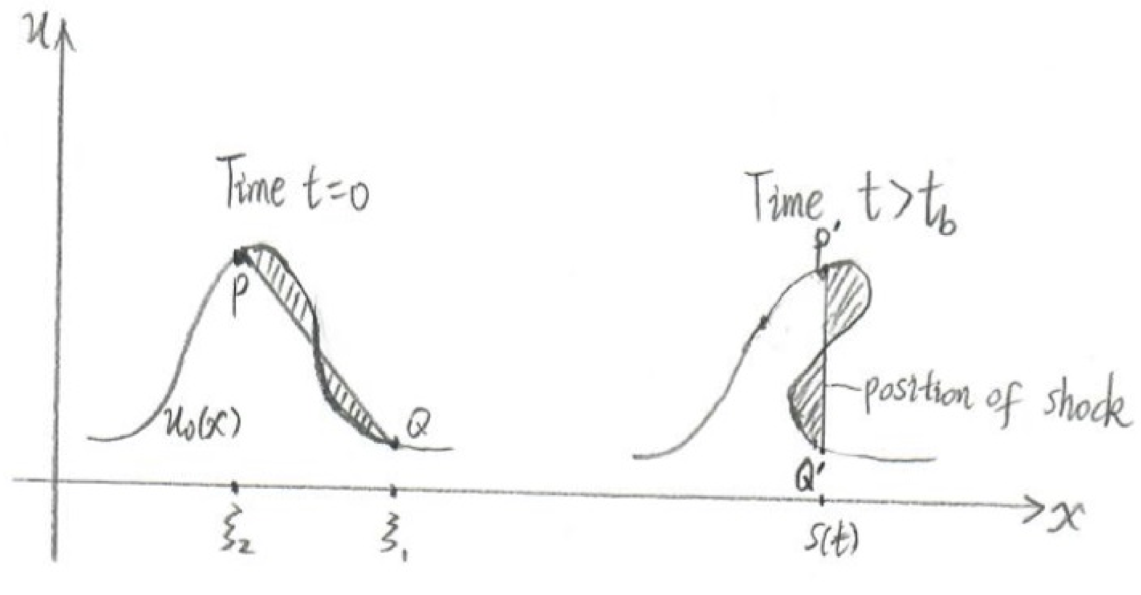

Figure 2: initial wave profile evolving into a multivalued wavelet

Input-to-State Stability(ISS) properties for Burgers Equation

Let us move on to another topic. Consider the following system for Burgers' equation with Dirichlet boundary conditions:

$$\begin{cases} u_t - \mu u_{xx} + \nu u u_x = u_0(x,t) \quad \text{in} \quad (0,1) \times R_+ \\ u(0,t) = 0, u(1,t) = d(t), \\ u(x,0) = u_0(x), \end{cases} \tag{11}$$

where $\mu > 0, \nu > 0$ are constants, $d(t)$ is the disturbance on the boundary, which can represent actuation or sensing errors, and $f(x,t)$ is the disturbance distributed over the domain. We assume that $f \in \mathcal{H}^{\theta,\frac{\theta}{2}}([0,1],\overline{R_+})$, and $d \in \mathcal{H}^{1+\frac{\theta}{2}}(\overline{R_+})$ for some $\theta \in (0,1)$, where $\mathcal{H}$ represents the Holder space([8, 9]). For the existence and uniqueness of the solution to system (11), the following theorem holds.

Theorem 1. *Assume that $u_0 \in \mathcal{H}^{2+\theta}([0,1])$ with $u_0(0) = 0, u_0(1) = d(0), \mu u_0''(0) + f(0,0) = 0$ and $\mu u_0''(1)+f(1,0) = d'(0)$. For any $T > 0$, there exists a unique classical solution $u \in \mathcal{H}^{2+\theta,1+\frac{\theta}{2}}([0,1]\times[0,T]) \subset C^{2,1}([0,1] \times [0,T])$ of system (11).*

The proof of this theorem follows from Theorem 6.1 in [8](pages 452-453). It is based on the linearization of the considered system and the Leray-Schauder theorem on fixed points.

Before listing the well-posedness result of system (11), we begin with some concepts and definitions, refer to [10, 11].

De Giorgi class Let $\Omega \subset R^n$ be an open bounded set, and γ be a constant. The De Giorgi class $DG^+(\Omega,\gamma)$ consists of functions $u \in W^{1,2}(\Omega)$ which satisfy, for every ball $B_r(y) \subset \Omega$, every $0 < r' < r$, and $k \in R$, the following inequality:

$$\int_{B_{r'}(y)} |\nabla(u-k)_+|^2 dx \leq \frac{\gamma}{(r-r')^2} \int_{B_r(y)} |(u-k)_+|^2 dx,$$

where $(u-k)_+ = \max\{u-k, 0\}$.

The main idea of De Giorgi iteration is to estimate $|A_k|$, the measure of $\{x \in \Omega; u(x) \geq k\}$, and derive $|A_k| = 0$ with some k for functions u in De Giorgi class. The following iteration given in [12] is useful.

Lemma 1. *Suppose that ϕ is a non-negative decreasing function on $[k_0,\infty)$ satisfying*

$$\phi(h) \leq (\frac{M}{h-k})^\alpha \phi^\beta(k), \quad \forall h > k \geq k_0,$$

where $M > 0, \alpha > 0, \beta > 1$ are constants. Then the following holds

$$\phi(k_0 + l_0) = 0,$$

with $l_0 = 2^{\frac{\beta}{\beta-1}} M \phi(k_0)^{\frac{\beta-1}{\alpha}}$.

Class $\mathcal{K}$ and Class $\mathcal{K}_\infty$ A continuous function $\alpha : [0,a) \to [0,\infty)$ belongs to class $\mathcal{K}$ if α is strictly increasing and $\alpha(0) = 0$. If, in addition, $a = \infty$ and $\alpha(r) \to \infty$ as $r \to \infty$, then α is called a class $\mathcal{K}_\infty$ function.

Class $\mathcal{KL}$ A continuous function $\beta : [0,a) \times [0,\infty) \to [0,\infty)$ belongs to class $\mathcal{KL}$ if for each fixed s, the mapping $r \mapsto \beta(r,s)$ is a class $\mathcal{K}$ function, and for each fixed r, the mapping $s \mapsto \beta(r,s)$ is decreasing in s and $\beta(r,s) \to 0$ as $s \to \infty$.

Here are some examples of class $\mathcal{KL}$ functions:

1. $\beta(r,s) = \frac{r}{1+rs}, r \geq 0, s \geq 0;$
2. $\beta(r,s) = \frac{r}{\sqrt{2r^2 s+1}}, r \geq 0, s \geq 0.$

ISS and EISS System (11) is said to be Input-to-State Stable(ISS) in $L^q(q \geq 2)$ with respect to(w.r.t.) boundary disturbances $d(t)$ and in-domain disturbances $f(x,t)$, if there exist functions $\beta \in \mathcal{KL}$ and $\gamma_1, \gamma_2 \in \mathcal{K}$ such that the solution to (11) satisfies

$$\begin{aligned} ||u(\cdot,t)||_{L^q(0,1)} \leq \beta(||u_0||_{L^q(0,1)}, t) + \gamma_1(\max_{[0,t]} |d(s)|) \\ + \gamma_2(\max_{[0,1]\times[0,t]} |f(x,s)|), \quad \forall t \geq 0. \end{aligned} \tag{12}$$

Moreover, it is said to be exponential input-to-state stable(EISS) if there exist $\beta' \in \mathcal{K}_\infty$ and a constant $\lambda > 0$ such that

$$\beta(||u_0||_{L^q(0,1)}, t) < \beta'(||u_0||_{L^q(0,1)})e^{-\lambda t}$$

in (12).

In order to use the technique of splitting and the method of De Giorgi iteration in the investigation of the ISS properties for the considered system, while guaranteeing the well-posedness of Theorem 1, we assume the compatibility condition $u_0(0) = u_0''(0) = u_0(1) = u_0''(1) = d(0) = d'(0) = f(0,0) = f(1,0) = 0$ always holds, then the ISS property for system (11) is stated in the following theorem.

Theorem 2. *System* (11) *is EISS in L^2 norm w.r.t boundary disturbances $d(t) \in \mathcal{H}^{1+\frac{\theta}{2}}(\overline{R_+})$ and in-domain disturbances $f(x,t) \in \mathcal{H}^{\theta,\frac{\theta}{2}}([0,1], \overline{R_+})$ satisfying $\sup_{\overline{R_+}} |d(s)| + \frac{18\sqrt{2}}{\mu} \sup_{[0,1]\times\overline{R_+}} |f(x,s)| < \frac{\mu}{\nu}$. And we have*

$$||u(\cdot,t)||^2 \leq 2||u_0||^2 + 4\max_{[0,t)} |d(s)|^2 + \frac{2592}{\mu^2} \sup_{[0,1]\times[0,t]} |f(x,s)|^2.$$

Let w be the unique solution of the following system:

$$\begin{cases} w_t - \mu w_{xx} + \nu w w_x = f(x,t) \quad \text{in} \quad (0,1) \times R_+ \\ w(0,t) = 0, w(1,t) = d(t), \\ w(x,0) = 0. \end{cases} \tag{13}$$

Then $v = u - w$ is the unique solution of the following system:

$$\begin{cases} v_t - \mu v_{xx} + \nu v v_x + \nu(wv)_x = 0 \quad \text{in} \quad (0,1) \times R_+ \\ v(0,t) = 0, v(1,t) = 0, \\ v(x,0) = u_0(x). \end{cases} \tag{14}$$

For system (13), the following estimate holds.

Lemma 2. *Suppose that $\mu > 0, \nu > 0$. For every $t > 0$, one has*

$$\max_{[0,1]\times[0,t]} |w(x,s)| \leq \max_{[0,t]} |d(s)| + \frac{18\sqrt{2}}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)|.$$

For system (14), we have the following estimate.

Lemma 3. *Suppose that $\mu > 0, \nu > 0$, and $\sup_{\overline{R_+}} |d(t)| + \frac{18\sqrt{2}}{\mu} \sup_{[0,1]\times\overline{R_+}} |f(x,t)| < \frac{2\mu}{5\nu}$. For every $t > 0$, one has*

$$||v(\cdot,t)||^2 \leq ||u_0||^2.$$

Proof of Theorem 2:

Proof. Note that $u = w + v$, we get from Lemma 2 and Lemma 3 that:

$$\begin{aligned} ||u(\cdot,t)||^2 &\leq 2||w(\cdot,t)||^2 + 2||v(\cdot,t)||^2 \\ &\leq 2(\max_{[0,1]\times[0,t]} |w(x,s)|)^2 + 2||v(\cdot,t)||^2 \\ &\leq 2||u_0||^2 + 2(\max_{[0,t]} |d(s)| + \frac{18\sqrt{2}}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)|)^2. \end{aligned}$$

Thus, the estimate holds and system (11) is EISS. □

Proof of Lemma 2:

Proof. For any $t > 0$, let $k_0 = \max\{\max_{[0,t]} d(s), 0\}$, then $(w(0,s)-k)_+ = (w(1,s)-k)_+ = 0$ for $k \geq k_0$. Let $I_k(s) = \int_0^1 ((w(x,s)-k)_+)^2 dx$, and suppose that $I_k(t_0) = \max_{[0,t]} I_k(s)$. Due to $I_k(0) = 0$ and $I_k(s) \geq 0$, we can assume that $t_0 > 0$ without loss of generality. Define $\eta(x,s) = (w(x,s)-k)_+ \chi_{[t_1,t_2]}(s)$, where $\chi_{[t_1,t_2]}(s)$ is the character function on $[t_1,t_2]$ with $0 \leq t_1 < t_2 \leq t_0$. For ϵ sufficiently small , choose $t_1 = t_0 - \epsilon, t_2 = t_0$, and multiply system (13) by η, we get

$$\frac{1}{2}\int_{t_0-\epsilon}^{t_0} \frac{d}{dt}\int_0^1 ((w-k)_+)^2 dxds + \mu \int_{t_0-\epsilon}^{t_0}\int_0^1 |((w-k)_+)_x|^2 dxds$$
$$+\nu \int_{t_0-\epsilon}^{t_0}\int_0^1 ww_x(w-k)_+ dxds \leq \int_{t_0-\epsilon}^{t_0}\int_0^1 |f|(w-k)_+ dxds.$$

Note that

$$\frac{1}{2}\int_{t_0-\epsilon}^{t_0} \frac{d}{dt}\int_0^1 ((w-k)_+)^2 dxds = \frac{1}{2}(I_k(t_0) - I_k(t_0-\epsilon)) \geq 0,$$

and

$$\lim_{\epsilon\to 0^+}\int_{t_0-\epsilon}^{t_0}\int_0^1 ww_x(w-k)_+ dxds = 0,$$

we get

$$\mu\int_0^1 |((w(x,t_0)-k)_+)_x|^2 dx \leq \int_0^1 |f(x,t_0)|(w(x,t_0)-k)_+ dx.$$

Using the fact that

$$(\int_a^b |u|^p dx)^{\frac{1}{p}} \leq (b-a)^{\frac{1}{p}}(\frac{2}{b-a}||u||^2 + (b-a)||u_x||^2)^{\frac{1}{2}} \quad \forall p \geq 1,$$

when $u \in C^1([a,b];R)$,refer to [13], and Pioncare's inequality,see [14], we have

$$(\int_0^1 |((w(x,t_0)-k)_+)_x|^p dx)^{\frac{2}{p}} \leq 9\int_0^1 |((w(x,t_0)-k)_+)_x|^2 dx$$
$$\leq \frac{9}{\mu}\int_0^1 |f(x,t_0))|(w(x,t_0)-k)_+ dx. \quad (\forall p > 2)$$

Let $A_k(s) = \{x \in (0,1); w(x,s) > k\}$, and $\phi_k = \sup_{(0,t)} |A_k(s)|$,then

$$(\int_{A_k(t_0)} |(w(x,t_0)-k)_+|^p dx)^{\frac{2}{p}} \leq \frac{9}{\mu}\int_{A_k(t_0)} |f(x,t_))|(w(x,t_0)-k)_+ dx.$$
$$\leq \frac{9}{\mu}(\int_{A_k(t_0)} |(w(x,t_0)-k)_+|^p dx)^{\frac{1}{p}})(\int_{A_k(t_0)} |f(x,t_0))|^q dx)^{\frac{1}{q}}$$

(Holder's Inequality)

Thus,

$$(\int_{A_k(t_0)} |(w(x,t_0)-k)_+|^p dx)^{\frac{1}{p}} \leq \frac{9}{\mu}(\int_{A_k(t_0)} |f(x,t_0))|^q dx)^{\frac{1}{q}}$$
$$\leq \frac{9}{\mu}|A_k(t_0)|^{\frac{1}{q}} \max_{[0,1]\times[0,t]} |f(x,s)|$$
$$\leq \frac{9}{\mu}\max_{[0,1]\times[0,t]} |f(x,s)|\phi_k^{\frac{1}{q}}$$

Moreover, with the definition of $I_k(s)$, we get

$$I_k(t_0) \leq (\int_{A_k(t_0)} |(w(x,t_0)-k)_+|^p dx)^{\frac{2}{p}})|A_k(t_0)|^{\frac{p-2}{p}} \text{(Holder's Inequality)}$$
$$\leq (\frac{9}{\mu}\max_{[0,1]\times[0,t]} |f(x,s)|)^2 \phi_k^{3-\frac{4}{p}}.$$

Recalling that $I_k(t_0) = \max\limits_{[0,t]} I_k(s)$, we have

$$I_k(s) \le I_k(t_0) \le (\frac{9}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)|)^2 \phi_k^{3-\frac{4}{p}}. \tag{15}$$

On the other hand,noticing that $|A_h(s)| < A_k(s)$ when $h > k$, we have

$$I_k(s) \ge \int_{A_h(s)} ((w(x,s)-k)_+)^2 dx \ge (h-k)^2 |A_h(s)| \tag{16}$$

Then we infer from (15) and (16) that

$$(h-k)^2 \phi_h \le (\frac{9}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)|)^2 \phi_k^{3-\frac{4}{p}},$$

i.e.

$$\phi_h \le (\frac{9}{\mu} \frac{\max\limits_{[0,1]\times[0,t]} |f(x,s)|}{h-k})^2 \phi_k^{3-\frac{4}{p}}.$$

As $p > 2$, we have $3 - \frac{4}{p} > 1$. By De Giorgi iteration in Lemma 1, we obtain

$$\phi_{k_0+l_0} = \sup_{[0,t]} |A_{k_0+l_0}| = 0,$$

where $l_0 = 2^{\frac{3p-4}{2p-4}} \frac{9}{\mu} \max\limits_{[0,1]\times[0,t]} |f(x,s)| \phi_{k_0}^{1-\frac{2}{p}} \le \frac{9}{\mu} 2^{\frac{3p-4}{2p-4}} \max\limits_{[0,1]\times[0,t]} |f(x,s)|$. Then

$$\begin{aligned} w(x,s) &\le k_0 + l_0 \\ &\le \max\{\max_{[0,t]} d(s), 0\} + \frac{9}{\mu} 2^{\frac{3p-4}{2p-4}} \max_{[0,1]\times[0,t]} |f(x,s)| \\ &\le \max\{\max_{[0,t]} d(s), 0\} + \frac{18\sqrt{2}}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)| \quad \text{as } p \to \infty. \end{aligned}$$

In order to prove the lower boundedness of $w(x,t)$, set $\overline{w} = -w$, we have

$$\begin{cases} \overline{w}_t - \mu \overline{w}_{xx} + \nu \overline{w}\overline{w}_x = -f(x,t) \quad \text{in} \quad (0,1) \times R_+ \\ \overline{w}(0,t) = 0, \overline{w}(1,t) = -d(t), \\ \overline{w}(x,0) = 0. \end{cases}$$

Proceeding as above, the De Giorgi iteration gives

$$-w(x,s) = \overline{w}(x,s) \le \max\{\max_{[0,t]} -d(s), 0\} + \frac{18\sqrt{2}}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)|.$$

Hence,

$$\max_{[0,1]\times[0,t]} |w(x,s)| \le \max_{[0,t]} |d(s)| + \frac{18\sqrt{2}}{\mu} \max_{[0,1]\times[0,t]} |f(x,s)|.$$

□

Proof of Lemma 3:

Proof. Multiplying System (14) by v and Integrating over $(0,1)$, we get

$$\int_0^1 v_t v dx + \mu \int_0^1 v_x^2 dx + \nu \int_0^1 v^2 v_x dx + \nu \int_0^1 (wv)_x v dx = 0$$

□

Note that $\int_0^1 v^2 v_x dx = 0$ and $\int_0^1 (wv)_x v dx = wv^2|_{x=0}^{x=1} - \int_0^1 wvv_x dx = -\int_0^1 wvv_x dx$, we deduce that

$$\begin{aligned}
&\frac{1}{2}\frac{d}{dt}||v(\cdot,t)||^2 + \mu||v_x(\cdot,t)||^2 \leq \nu\int_0^1 |wvv_x|dx \\
\leq&\frac{\nu}{2}\max_{[0,1]\times[0,t]}|w(x,s)|(||v(\cdot,t)||^2 + ||v_x(\cdot,t)||^2) \\
\leq&\frac{\nu}{2}(\max_{[0,t]}|d(s)| + \frac{18\sqrt{2}}{\mu}\max_{[0,1]\times[0,t]}|f(x,s)|)(||v(\cdot,t)||^2 + ||v_x(\cdot,t)||^2) \\
<&\frac{\nu}{2}\frac{2\mu}{5\nu}(||v(\cdot,t)||^2 + ||v_x(\cdot,t)||^2) \\
<&\mu||v_x(\cdot,t)||^2 \quad \text{(Poincare's inequality)}
\end{aligned}$$

Thus,

$$\frac{d}{dt}||v(\cdot,t)||^2 < 0,$$

hence,

$$||v(\cdot,t)||^2 \leq ||v(\cdot,0)||^2 = ||u_0||^2.$$

iISS System (11) is said to be ISS w.r.t. boundary disturbances $d(t)$ and integral input-to-state stable(iISS) w.r.t. in-domain disturbances $f(x,t)$ in L^q-norm($q \geq 2$), if there exist functions $\beta \in \mathcal{KL}, \theta \in \mathcal{K}_\infty$ and $\gamma_1, \gamma_2 \in \mathcal{K}$ such that the solution to (11) satisfies

$$\begin{aligned}
||u(\cdot,t)||_{L^q(0,1)} \leq& \beta(||u_0||_{L^q(0,1)}, t) + \gamma_1(\max_{[0,t]}|d(s)|) \\
&+ \theta(\int_0^t \gamma_2(||f(x,s)||)ds), \quad \forall t \geq 0.
\end{aligned} \tag{17}$$

Theorem 3. *System (11) is EISS in L^2 norm w.r.t boundary disturbances $d(t) \in \mathcal{H}^{1+\frac{\theta}{2}}(\overline{R_+})$ satisfying $\sup_{t\in\overline{R_+}}|d(t)| < \frac{\mu}{\nu}$, and EiISS w.r.t in-domain disturbances $f(x,t) \in \mathcal{H}^{\theta,\frac{\theta}{2}}([0,1], \overline{R_+})$, with the following estimate for any $t > 0$:*

$$||u(\cdot,t)||^2 \leq 2||u_0||^2 + 2\max_{[0,t)}|d(s)|^2 + \frac{2}{\epsilon}\int_0^t ||f(\cdot,s)||^2 ds.$$

In order to prove Theorem 3, consider the following two systems:

$$\begin{cases} w_t - \mu w_{xx} + \nu w w_x = 0 \quad \text{in} \quad (0,1)\times R_+ \\ w(0,t) = 0, w(1,t) = d(t), \\ w(x,0) = 0. \end{cases} \tag{18}$$

and

$$\begin{cases} v_t - \mu v_{xx} + \nu v v_x + \nu(wv + vw)_x = f(x,t) \quad \text{in} \quad (0,1)\times R_+ \\ v(0,t) = 0, v(1,t) = 0, \\ v(x,0) = u_0(x). \end{cases} \tag{19}$$

where $v = u - w$.
For system (18), it is a special case of system (13). And for system (19), we have the following estimate:

Lemma 4. *Suppose that $\mu > 0, \nu > 0$, and $\sup_{\overline{R_+}}|d(t)| < \frac{\mu}{\nu}$. For every $t > 0$, one has*

$$||v(\cdot,t)||^2 \leq ||u_0||^2 + \frac{1}{\epsilon}\int_0^t ||f(\cdot,s)||^2 ds, \quad \forall \epsilon \in (0,\mu).$$

Based on the results of Lemma 2 and Lemma 4, the estimate in Theorem 3 holds.
Proof of Lemma 4:

Proof. Multiply system (19) by v and integrating over $(0,1)$, we get

$$\int_0^1 vv_t dx + \mu \int_0^1 v_x^2 dx + \nu \int_0^1 v^2 v_x dx + \nu \int_0^1 (wv)_x v dx = \int_0^1 f(x,t) dx.$$

Then

$$\begin{aligned}
&\frac{1}{2}\frac{d}{dt}||v(\cdot,t)||^2 + \mu ||v_x(\cdot,t)||^2 \\
\leq &\nu \int_0^1 |wvv_x| dx + \int_0^1 f(x,t) v dx \\
\leq &\frac{\nu}{2} \max_{[0,t]} |d(s)| (||v(\cdot,t)||^2 + ||v_x(\cdot,t)||^2) \\
&+ \frac{1}{2\epsilon}||f(\cdot,t)||^2 + \frac{\epsilon}{2}||v(\cdot,t)||^2 \qquad \text{(Young's Inequalty)} \\
< &\frac{\mu}{5}(||v(\cdot,t)||^2 + ||v_x(\cdot,t)||^2) + \frac{1}{2\epsilon}||f(\cdot,t)||^2 + \frac{\epsilon}{2}||v(\cdot,t)||^2
\end{aligned}$$

where we choose ϵ sufficiently small.
Thus,

$$\frac{d}{dt}||v(\cdot,t)||^2 \leq \frac{1}{\epsilon}||f(\cdot,t)||^2.$$

Integrating from 0 to t, we get

$$\begin{aligned}
||v(\cdot,t)||^2 &\leq ||v(\cdot,0)||^2 + \frac{1}{\epsilon}\int_0^t ||f(\cdot,s)||^2 ds \\
&\leq ||u_0||^2 + \frac{1}{\epsilon}\int_0^t ||f(\cdot,s)||^2 ds.
\end{aligned}$$

□

Numerical experiments of Burgers Equation

In the section, we use finite differences and the Lax-Wendroff method(see [15]) to obtain the solution of inviscid time-dependent Burgers equation. The source code written in matlab is in the end of this article. Here we only give partial result of this method,shown as follows:

```
21-Apr-2019 14:29:01

FD1D_BURGERS_LAX:
  MATLAB version
  Solve the non-viscous time-dependent Burgers equation,
  using the Lax-Wendroff method.

  Equation to be solved:

    du/dt + u * du/dx = 0

  for x in [ a, b ], for t in [t_init, t_last]

  with initial conditions:

    u(x,o) = u_init

  and boundary conditions:
```

```
  u(a,t) = u_a(t), u(b,t) = u_b(t)

-1.000000 <= X <= 1.000000
Number of nodes = 41
DX = 0.050000

0.000000 <= t <= 1.000000
Number of time steps = 80
DT = 0.012500

X:

       -1.000000       -0.950000       -0.900000       -0.850000       -0.800000
       -0.750000       -0.700000       -0.650000       -0.600000       -0.550000
       -0.500000       -0.450000       -0.400000       -0.350000       -0.300000
       -0.250000       -0.200000       -0.150000       -0.100000       -0.050000
        0.000000        0.050000        0.100000        0.150000        0.200000
        0.250000        0.300000        0.350000        0.400000        0.450000
        0.500000        0.550000        0.600000        0.650000        0.700000
        0.750000        0.800000        0.850000        0.900000        0.950000
        1.000000

STEP = 0
TIME = 0.000000
STABILTY = 0.125000
```

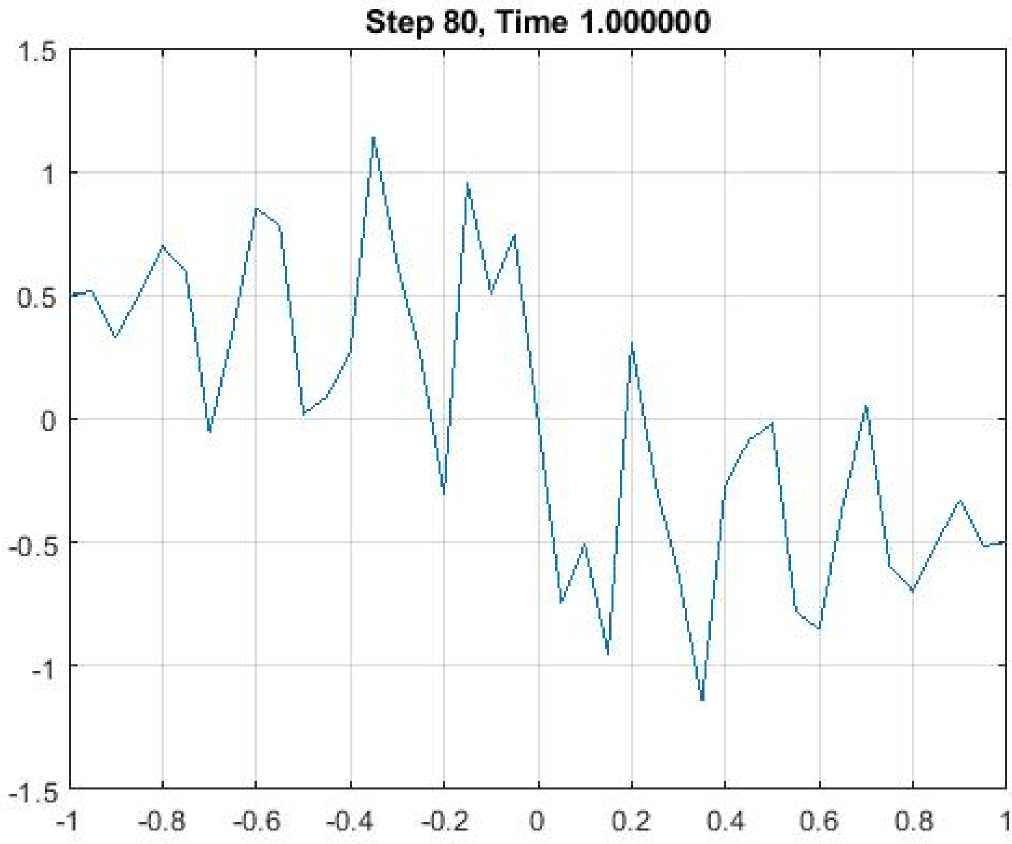

Figure 3:

```
          0.5        0.48368       0.466525       0.448495       0.429553
     0.409666         0.3888       0.366932       0.344042        0.32012
     0.295167       0.269197       0.242238       0.214334       0.185547
     0.155958       0.125666      0.0947863       0.063451      0.0318045
            0     -0.0318045      -0.063451     -0.0947863      -0.125666
    -0.155958      -0.185547      -0.214334      -0.242238      -0.269197
    -0.295167       -0.32012      -0.344042      -0.366932        -0.3888
    -0.409666      -0.429553      -0.448495      -0.466525       -0.48368
         -0.5

STEP = 1
TIME = 0.012500
STABILTY = 0.125000

          0.5       0.491773        0.47473       0.456785       0.437893
     0.418014        0.39711       0.375146       0.352098       0.327947
     0.302688       0.276327       0.248889       0.220416       0.190969
     0.160634       0.129515      0.0977389       0.065452       0.032815
  5.98005e-19      -0.032815      -0.065452     -0.0977389      -0.129515
    -0.160634      -0.190969      -0.220416      -0.248889      -0.276327
    -0.302688      -0.327947      -0.352098      -0.375146       -0.39711
    -0.418014      -0.437893      -0.456785       -0.47473      -0.491773
         -0.5

STEP = 2
TIME = 0.025000
STABILTY = 0.125000

          0.5       0.497809       0.483033       0.465196       0.446381
     0.426538       0.405622       0.383591        0.36041       0.336053
     0.310505       0.283766       0.255853       0.226805       0.196683
     0.165575       0.133592       0.100874      0.0675796      0.0338904
  1.19612e-18     -0.0338904     -0.0675796      -0.100874      -0.133592
    -0.165575      -0.196683      -0.226805      -0.255853      -0.283766
    -0.310505      -0.336053       -0.36041      -0.383591      -0.405622
    -0.426538      -0.446381      -0.465196      -0.483033      -0.497809
         -0.5


.......
.......
.......


STEP = 79
TIME = 0.987500
STABILTY = 0.284547

          0.5       0.486051       0.350438       0.585817       0.674504
     0.478441     -0.0459906       0.540337       0.882731       0.578412
   -0.0601122       0.161666       0.560493        1.13819       0.292521
      0.23613      -0.078433        1.01461       0.355727       0.725162
  1.63746e-15      -0.725162      -0.355727       -1.01461       0.078433
     -0.23613      -0.292521       -1.13819      -0.560493      -0.161666
    0.0601122      -0.578412      -0.882731      -0.540337      0.0459906
    -0.478441      -0.674504      -0.585817      -0.350438      -0.486051
```

```
          -0.5

  STEP = 80
  TIME = 1.000000
  STABILTY = 0.285560

           0.5        0.516579         0.32841        0.501757        0.696799
      0.594191      -0.0579838        0.354658        0.851096        0.779783
     0.0197099       0.0872451        0.265955         1.14224         0.62925
      0.256108       -0.306763        0.950214        0.507699        0.744095
   1.72333e-15       -0.744095       -0.507699       -0.950214        0.306763
     -0.256108        -0.62925        -1.14224       -0.265955      -0.0872451
    -0.0197099       -0.779783       -0.851096       -0.354658       0.0579838
     -0.594191       -0.696799       -0.501757        -0.32841       -0.516579
          -0.5

FD1D_BURGERS_LAX:
  Normal end of execution.
```

References

[1] Landajuela, Mikel. "Burgers equation." BCAM Internship: Basque center for applied mathematics (2011).

[2] Toshimitsu Musha, Hideyo Higuchi. Traffic Current Fluctuation and the Burgers Equation. Japanese Journal of Applied Physics, Volume 17, Number 5,(1978).

[3] Faber, T. E. Fluid Dynamics for Physicists. New York: Cambridge University Press, (1995).

[4] Evans, Lawrence C. Partial Differential Equations: Second Edition. University of California, Berkeley, (2010).

[5] Herron, Isom H., and Michael R. Foster. Partial differential equations in fluid dynamics. Cambridge University Press, (2008).

[6] Logan, J. David. An introduction to nonlinear partial differential equations. Vol. 89. John Wiley & Sons, (2008).

[7] Whitham, Gerald Beresford. Linear and nonlinear waves. Vol. 42. John Wiley & Sons, (1974).

[8] Ladyzhenskaia, Olga Aleksandrovna, Vsevolod Alekseevich Solonnikov, and Nina N. Ural'ceva. Linear and quasi-linear equations of parabolic type. Vol. 23. American Mathematical Soc, (1968).

[9] Driver, Bruce K. "Analysis tools with applications." Lecture Notes, (2003).

[10] DiBenedetto, Emmanuele, and Ugo Gianazza. "Some properties of DeGiorgi classes." arXiv preprint arXiv:1604.07699 (2016).

[11] Terrell, William J. Stability and stabilization: an introduction. Princeton University Press, (2009).

[12] Wu, Zhuoqun, Jingxue Yin, and Chunpeng Wang. Elliptic & parabolic equations. World Scientific, (2006).

[13] Zheng Jun, and Guchuan Zhu. "A De Giorgi iteration-based approach for the establishment of ISS properties of a class of semi-linear parabolic PDEs with boundary and in-domain disturbances." arXiv preprint arXiv:1710.09917 (2017).

[14] Krstic, Miroslav. "On global stabilization of Burgers' equation by boundary control." Systems & Control Letters 37, no. 3 (1999): 123-141.

[15] Sod, Gary A. "A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws." Journal of computational physics 27, no. 1 (1978): 1-31.

Source code(Matlab)

```
function fd1d_burgers_lax ( )

%*****************************************************************************80
%
%% FD1D_BURGERS_LAX solves the nonviscous Burgers equation using Lax-Wendroff.
%
%  Discussion:
%
%    The non-viscous time-dependent Burgers equation is:
%
%      du/dt + u du/dx = 0
%
%    which can be written in conservative form as
%
%      du/dt + 1/2 d/dx ( u^2 ) = 0
%
%    or
%
%      du/dt + dF/dx = 0
%
%    For the Burgers equation, we define
%
%      F(x,t) = 1/2 u^2,
%      A(x,t) = dF/dx = u
%
%    and then the Lax-Wendroff method approximates the solution
%    using the iteration:
%
%      u(x,t+dt) = u(t) - dt dF/dx + 1/2 dt^2 d/dx A dF/dx
%
%    which can be written:
%
%      u(x,t+dt) = u(x,t) - dt ( F(x+dx,t) - F(x-dx,t) ) / ( 2 * dx )
%        + 1/2 dt^2/dx^2 ( A(x+dx/2,t) * ( F(x+dx,t) - F(x,t) )
%                        - A(x-dx/2,t) * ( F(x,t) - F(x-dx,t) )
%
%    where we approximate:
%
%      A(x+dx/2,t) = 1/2 ( u(x+dx,t) + u(x,t) )
%      A(x-dx/2,t) = 1/2 ( u(x,t) + u(x-dx,t) )
%
%    There is a stability condition that applies here, which requires that
%
%      dt * max ( abs ( u ) ) / dx <= 1
%
%  Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
%  Modified:
%
%    21 August 2010
%
%  Author:
%
%    John Burkardt
```

```
%
%  Parameters:
%
%    None
%
  timestamp ( );

  fprintf ( 1, '\n' );
  fprintf ( 1, 'FD1D_BURGERS_LAX:\n' );
  fprintf ( 1, '  MATLAB version\n' );
  fprintf ( 1, '  Solve the non-viscous time-dependent Burgers equation,\n' );
  fprintf ( 1, '  using the Lax-Wendroff method.\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '  Equation to be solved:\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '    du/dt + u * du/dx = 0\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '  for x in [ a, b ], for t in [t_init, t_last]\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '  with initial conditions:\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '    u(x,o) = u_init\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '  and boundary conditions:\n' );
  fprintf ( 1, '\n' );
  fprintf ( 1, '    u(a,t) = u_a(t), u(b,t) = u_b(t)\n' );
%
%  Set and report the problem parameters.
%
  n = 41;
  a = -1.0;
  b = +1.0;
  dx = ( b - a ) / ( n - 1 );
  step_num = 80;
  t_init = 0.0;
  t_last = 1.0;
  dt = ( t_last - t_init ) / step_num;

  fprintf ( 1, '\n' );
  fprintf ( 1, '  %f <= X <= %f\n', a, b );
  fprintf ( 1, '  Number of nodes = %d\n', n );
  fprintf ( 1, '  DX = %f\n', dx );
  fprintf ( 1, '\n' );
  fprintf ( 1, '  %f <= t <= %f\n', t_init, t_last );
  fprintf ( 1, '  Number of time steps = %d\n', step_num );
  fprintf ( 1, '  DT = %f\n', dt );

  x = r8vec_even ( n, a, b );

  fprintf ( 1, '\n' );
  fprintf ( 1, '  X:\n' );
  fprintf ( 1, '\n' );
  for ilo = 1 : 5 : n
    ihi = min ( ilo + 4, n );
    for i = ilo : ihi
      fprintf ( 1, '  %16f', x(i,1) );
    end
```

```
    fprintf ( 1, '\n' );
  end
%
%  Set the initial condition,
%  and apply boundary conditions to first and last entries.
%
  step = 0;
  t = t_init;
  un(1:n,1) = u_init ( n, x, t );
  un(1,1) = u_a ( x(1,1), t );
  un(n,1) = u_b ( x(n,1), t );

  stability = ( dt / dx ) * max ( abs ( un(1:n,1) ) );
  report ( step, step_num, n, x, t, un, stability );

  if ( true )
    plot ( x, un );
    grid ( 'on' );
    title ( sprintf ( 'Step %d, Time %f', step, t ) );
  end
%
%  March in time.
%
  c1 = - ( 0.5 * dt   / dx );
  c2 = - ( 0.5 * dt^2 / dx^2 );

  for step = 1 : step_num

    t = ( ( step_num - step ) * t_init   ...
        + (            step ) * t_last ) ...
        / ( step_num        );

    uo(1:n,1) = un(1:n,1);

    un(2:n-1,1) = uo(2:n-1,1) ...
      -       ( dt   / dx   ) * ( uo(3:n,1).^2 - uo(1:n-2,1).^2 ) ...
      + 0.5 * ( dt^2 / dx^2 ) * (  0.5 * ( uo(3:n,1)   + uo(2:n-1,1) ) ...
                                  .* ( uo(3:n,1).^2   - uo(2:n-1,1).^2 ) ...
                                 - 0.5 * ( uo(2:n-1,1) + uo(1:n-2,1) ) ...
                                  .* ( uo(2:n-1,1).^2 - uo(1:n-2,1).^2 ) );

    un(1,1) = u_a ( x(1,1), t );
    un(n,1) = u_b ( x(n,1), t );

    stability = ( dt / dx ) * max ( abs ( un(1:n,1) ) );
    report ( step, step_num, n, x, t, un, stability );

    if ( true )
      plot ( x, un );
      grid ( 'on' );
      title ( sprintf ( 'Step %d, Time %f', step, t ) );
    end

  end
%
%  Terminate.
%
```

```
  fprintf ( 1, '\n' );
  fprintf ( 1, 'FD1D_BURGERS_LAX:\n' );
  fprintf ( 1, '  Normal end of execution.\n' );

  return
end
function a = r8vec_even ( n, alo, ahi )

%*****************************************************************************80
%
%% R8VEC_EVEN returns N real values, evenly spaced between ALO and AHI.
%
%  Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
%  Modified:
%
%    24 January 2004
%
%  Author:
%
%    John Burkardt
%
%  Parameters:
%
%    Input, integer N, the number of values.
%
%    Input, real ALO, AHI, the low and high values.
%
%    Output, real A(N), N evenly spaced values.
%    Normally, A(1) = ALO and A(N) = AHI.
%    However, if N = 1, then A(1) = 0.5*(ALO+AHI).
%
  if ( n == 1 )

    a(1,1) = 0.5 * ( alo + ahi );

  else

    a(1:n,1) = ( (n-1:-1:0) * alo + (0:n-1) * ahi ) / ( n - 1 );

  end

  return
end
function report ( step, step_num, n, x, t, u, stability )

%*****************************************************************************80
%
%% REPORT prints or plots or saves the data at the current time step.
%
%  Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
%  Modified:
```

```
%
%    18 August 2010
%
%  Author:
%
%    John Burkardt
%
%  Parameters:
%
%    Input, integer STEP, the index of the current step,
%    between 0 and STEP_NUM.
%
%    Input, integer STEP_NUM, the number of steps to take.
%
%    Input, integer N, the number of nodes.
%
%    Input, real X(N), the coordinates of the nodes.
%
%    Input, real T, the current time.
%
%    Input, real U(N), the initial values U(X,T).
%
%    Input, real STABILITY, the stability factor, which should be
%    no greater than 1.
%
  fprintf ( 1, '\n' );
  fprintf ( 1, '  STEP = %d\n', step );
  fprintf ( 1, '  TIME = %f\n', t );
  fprintf ( 1, '  STABILTY = %f\n', stability )
  fprintf ( 1, '\n' );
  for ilo = 1 : 5 : n
    ihi = min ( ilo + 4, n );
    for i = ilo : ihi
      fprintf ( 1, '  %14g', u(i) );
    end
    fprintf ( 1, '\n' );
  end

  return
end
function ua = u_a ( x, t )

%*****************************************************************************80
%
%% U_A sets the boundary condition for U at A.
%
%  Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
%  Modified:
%
%    18 August 2010
%
%  Author:
%
%    John Burkardt
```

```
%
%  Parameters:
%
%    Input, real X, T, the position and time.
%
%    Output, real UA, the prescribed value of U(X,T).
%
  ua = + 0.5;

  return
end
function ub = u_b ( x, t )

%*****************************************************************************80
%
%% U_B sets the boundary condition for U at B.
%
%  Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
%  Modified:
%
%    18 August 2010
%
%  Author:
%
%    John Burkardt
%
%  Parameters:
%
%    Input, real X, T, the position and time.
%
%    Output, real UB, the prescribed value of U(X,T).
%
  ub = - 0.5;

  return
end
function u = u_init ( n, x, t )

%*****************************************************************************80
%
%% U_INIT sets the initial condition for U.
%
%  Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
%  Modified:
%
%    18 August 2010
%
%  Author:
%
%    John Burkardt
%
```

```
% Parameters:
%
%    Input, integer N, the number of nodes.
%
%    Input, real X(N), the coordinates of the nodes.
%
%    Input, real T, the current time.
%
%    Output, real U(N), the initial values U(X,T).
%
  ua = u_a ( x(1,1), t );
  ub = u_b ( x(n,1), t );

  q = 2.0 * ( ua - ub ) / pi;
  r = ( ua + ub ) / 2.0;
%
% S can be varied.  It is the slope of the initial condition at the midpoint.
%
  s = 1.0;
  u(1:n,1) = ( 2 * x(1:n,1) - x(n,1) - x(1,1) ) ...
           / (                x(n,1) - x(1,1) );

  u(1:n,1) = - q * atan ( s * u(1:n,1) ) + r;

  return
end

function timestamp ( )

%*****************************************************************************80
%
%% TIMESTAMP prints the current YMDHMS date as a timestamp.
%
% Licensing:
%
%    This code is distributed under the GNU LGPL license.
%
% Modified:
%
%    14 February 2003
%
% Author:
%
%    John Burkardt
%
  t = now;
  c = datevec ( t );
  s = datestr ( c, 0 );
  fprintf ( 1, '%s\n', s );

  return
end
```