NEURAL NETWORK METHODS:
SCALAR HYPERBOLIC CONSERVATION LAWS

Zhiqiang Cai[1]

Collaborators: Min Liu[2], Jingshuang Chen[3], Junpyo Choi[1], Brooke Hejnal[1]

PURDUE UNIVERSITY

Department of Mathematics [1]
School of Mechanical Engineering [2]

Outline

- Neural Network (as a "new" class of approximating functions)
- Scalar Hyperbolic Conservation Laws
- Least-Squares Neural Network (LSNN) Method (a space-time approach)
- Evolving Neural Network (ENN) Method (an approach emulating physics)

Neural Network (NN): a "new" class of approximating functions

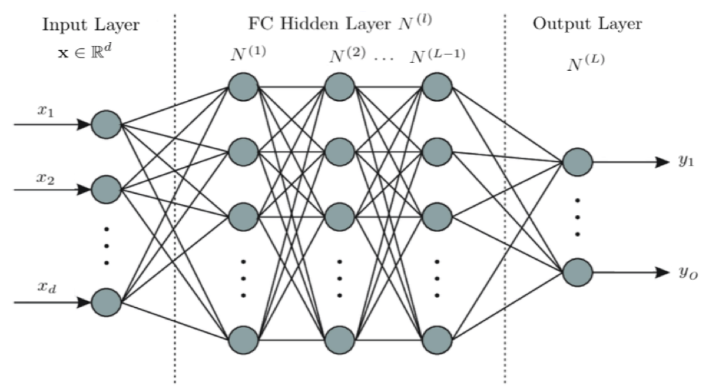
Fully-connected (Multi-Layer Perceptron) NN (Rosenblatt 1958)

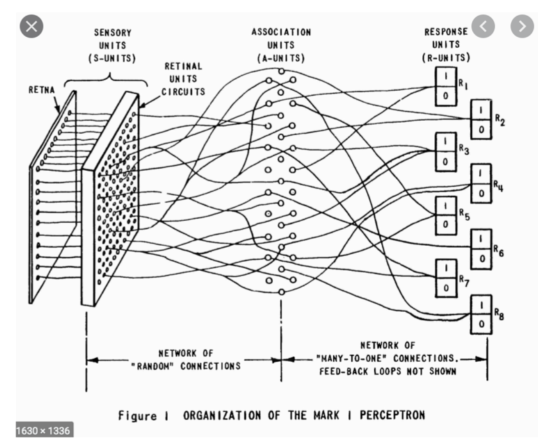
- **NN function**

$$v(\mathbf{x}) = c_0 + \sum_{j=1}^{n_l} c_j \, x_j^{(l)}(\mathbf{x})$$

Let $\mathbf{x}^{(0)} = \mathbf{x}$ and $x_i^{(k)}(\mathbf{x}) = \sigma\left(\mathbf{w}_i^{(k)}\mathbf{x}^{(k-1)} + b_i^{(k)}\right)$

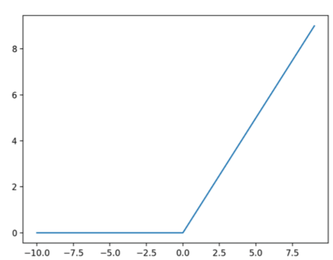
for $i = 1, \ldots, n_k$ and $k = 1, \ldots, l$





- **ReLU Activation function**

$$\sigma(t) = \begin{cases} t, & t > 0, \\ 0, & t \leq 0. \end{cases}$$



C^0 Linear Elements on fixed and moving meshes

- C^0 Linear Element on a **fixed** mesh in [a,b]

$$\mathcal{S}_1^0(\Delta) = \text{span}\,\{\phi_i(x)\}_{i=0}^n = \left\{\sum_{i=0}^n c_i\phi_i(x)\,:\, c_i \in \mathcal{R}\right\} \qquad \phi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}}, & x \in (x_{i-1}, x_i), \\ \dfrac{x_{i+1} - x}{x_{i+1} - x_i}, & x \in (x_i, x_{i+1}), \\ 0, & \text{otherwise} \end{cases}$$

- C^0 Linear Element on a **moving** mesh in [a,b]

$$\mathcal{S}_1^0(n) = \left\{\sum_{i=0}^n c_i\phi_i(x; x_{i-1}, x_i, x_{i+1})\,:\, c_i \in \mathcal{R},\ x_i \in [a,b]\right\} \qquad u(x) = x^{0.01},\ x \in [0,1]$$

$$= \left\{c_0 + c_1(x-a) + \sum_{i=2}^n c_i\sigma(x - x_i)\,:\, c_i \in \mathcal{R},\ x_i \in (a,b)\right\}$$

One hidden-layer NN in R^d

- **One hidden-layer NN** (C⁰ piecewise linear function)

$$\mathcal{M}_n(d) = \left\{ c_0 + \sum_{i=1}^{n} c_i \sigma(\boldsymbol{\omega}_i \mathbf{x} + b_i) \,:\, c_i, b_i \in \mathcal{R},\ \boldsymbol{\omega}_i \in \mathcal{S}^{d-1} \right\}$$

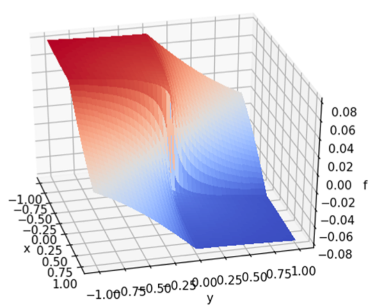
- **Breaking Hyper-Planes**

$$\mathcal{P}_i \,:\, \boldsymbol{\omega}_i \mathbf{x} + b_i = 0 \quad \text{for} \quad i = 1, \ldots, n$$

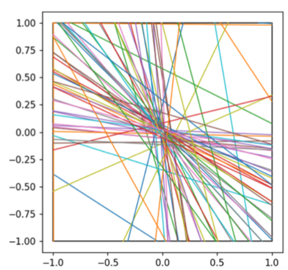
- **Linearly Independence**

$\{\sigma(\boldsymbol{\omega}_i \mathbf{x} + b_i)\}_{i=1}^{n}$ are linearly independent if $\{\mathcal{P}_i\}_{i=1}^{n}$ are distinct.

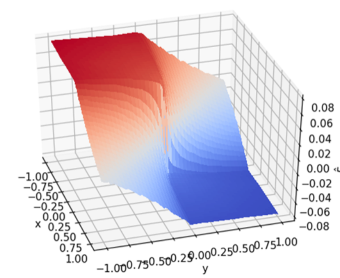
- **Physical Partition of NN approximation to Kellogg function**



(a) Target function f(x,y)



(h) Optimum break lines (69 neurons, 1286 elements)



(i) Optimum NN model of 69 neurons, $\xi = 0.008476$

Scalar Hyperbolic Conservation Laws

- **Scalar Nonlinear Hyperbolic Conservation Laws**

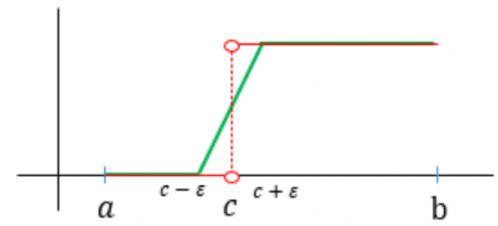
$$
\begin{cases}
u_t(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot \mathbf{f}(u) = 0, & \text{in } \Omega \times I, \\
u = g, & \text{on } \Gamma_-, \\
u(\mathbf{x},0) = u_0(\mathbf{x}), & \text{in } \Omega,
\end{cases}
$$

- **Numerical Difficulties**
    - PDE theory
    - Discontinuous solution with unknown interfaces

Approximation to Unit Step Function with Unknown Interface

- **Unit step function and its CPWL approximation**

$$f_c(x) = \begin{cases} 0, & a < x < c, \\ 1, & c < x < b \end{cases} \qquad p_c(x) = \begin{cases} 0, & a < x \le c-\varepsilon, \\ \dfrac{x-(c-\varepsilon)}{2\varepsilon}, & c-\varepsilon \le x \le c+\varepsilon, \\ 1, & c+\varepsilon \le x < b \end{cases}$$



$$\|f_c - p_c\|_{L^\infty(I)} = \frac{1}{2} \quad \text{and} \quad \|f_c - p_c\|_{L^r(I)} = \frac{\varepsilon^{1/r}}{2^{1-1/r}(1+r)^{1/r}}$$

- **How to compute or approximate $p_c(x)$ when c is unknown?**

(1) On **fixed** quasi-uniform mesh

- very fine mesh-size: $h = \varepsilon$
- overshooting, oscillation, etc.

(2) On **moving** mesh (neural network)

- two neurons
- no overshooting or oscillation

$$p_c(x) = \frac{1}{b_2 - b_1}\left[\sigma(x-b_1) - \sigma(x-b_2)\right], \qquad b_1 = c-\varepsilon,\ b_2 = c+\varepsilon$$

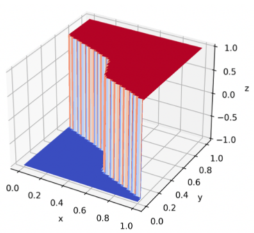
Approximation to Unit Step Function with Unknown Interface in R^d

- **Piecewise Constant function with unknow interface**

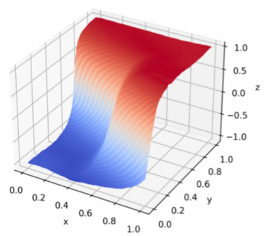
C., J. Choi, and M. Liu (2022) (d=2, 3, l=2; d=4,...,8, l=3)

Let $\chi(x)$ be a piecewise constant function with C^0 piecewise smooth interface I, then there exists a CPWL function p(x) generated by a DNN with L=$\lceil \log_2(d+1) \rceil$ hidden layers such that for any given $\varepsilon > 0$, we have

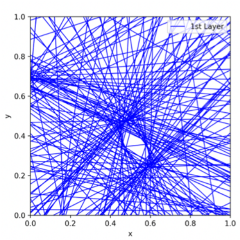
$$|||\chi - p|||_{\boldsymbol{\beta}} \le \sqrt{2|I|}\,\left|\alpha_1 - \alpha_2\right|\sqrt{\varepsilon},$$



(b) Exact solution

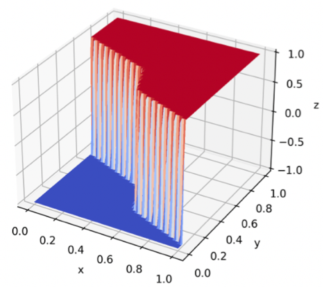


(c) 2 layer NN approximation



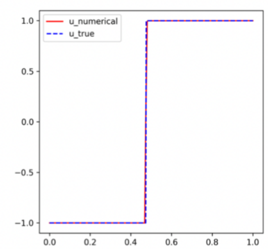
(g) 2 layer NN breaking lines

2-300-1

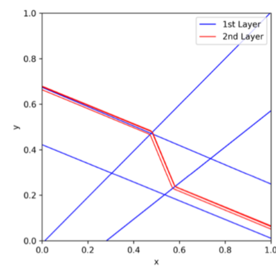


(d) 3 layer NN approximation

2-5-5-1



(f) 3 layer NN trace on $y = x$



(h) 3 layer NN breaking lines

Approximation to Unit Step Function with Unknown Interface in R^d

- **Piecewise Constant function with unknow interface**

C., J. Choi, and M. Liu (2022) (d=2, 3, L=2; d=4,...,8, L=3)

Let χ(x) be a piecewise constant function with C^0 piecewise smooth interface I, then there exists a CPWL function p(x) generated by a DNN with L=$\lceil \log_2(d+1) \rceil$ hidden layers such that for any given $\varepsilon > 0$, we have

$$|||\chi - p|||_{\boldsymbol{\beta}} \leq \sqrt{2|I|}\, \big|\alpha_1 - \alpha_2\big| \sqrt{\varepsilon},$$

P. Petersen and F. Voigtlaender (2018) (For C^1 and d=2, L=36)

Theorem 3.5. *For $r \in \mathbb{N}$, $d \in \mathbb{N}_{\geq 2}$, and $p, \beta, B > 0$, there are constants $c = c(d, r, p, \beta, B) > 0$ and $s = s(d, r, p, \beta, B) \in \mathbb{N}$, such that for any $K \in \mathcal{K}_{r,\beta,d,B}$ and any $\varepsilon \in (0, 1/2)$, there is a neural network Φ_ε^K with at most $(3 + \lceil \log_2 \beta \rceil) \cdot (11 + 2\beta/d)$ layers, and at most $c \cdot \varepsilon^{-p(d-1)/\beta}$ nonzero, (s, ε)-quantized weights such that*

$$\|\mathrm{R}_\varrho(\Phi_\varepsilon^K) - \chi_K\|_{L^p([-1/2,1/2]^d)} < \varepsilon \quad \text{and} \quad \|\mathrm{R}_\varrho(\Phi_\varepsilon^K)\|_{\sup} \leq 1.$$

Remark 3.6. *Theorem 3.5 establishes approximation rates for piecewise constant functions. It should be noted that the number of required layers is fixed and only depends on the dimension d and the regularity parameter β; in particular, it does* not *depend on the approximation accuracy ε.*

Physics-Informed Neural Network (PINN), a statistical approach

Psichogios-Ungar (92), Lagaris-Likas-Ftiadis (98), Rasissi-Perdikaris-Karniadakis (19), ...

PDE: $\mathcal{L}(u) = 0$ in $\Omega \in \mathcal{R}^d$ and $\mathcal{B}(u) = 0$ on $\partial\Omega$

training data: $\{x_i^u\}_{i=1}^{N_u} \subset \Omega$ and $\{x_i^b\}_{i=1}^{N_b} \subset \partial\Omega$

l^2 residual:

$$L(u) = \frac{1}{N_u}\sum_{i=1}^{N_u}\left(\mathcal{L}(u(x_i^u))\right)^2 + \frac{1}{N_b}\sum_{i=1}^{N_b}\left(\mathcal{B}(u(x_i^b))\right)^2$$

(mean squares error)

PINN:

$$u_{\mathcal{N}} = \arg\min_{v\in\mathcal{N}} L(v)$$

What are issues to use Neural Network in scientific computing?

Issues for NN-based Methods

- What is a proper **equivalent** formulation of a given PDE?
- How to choose NN architecture for a given problem?
- Numerical Issues (**unlike finite elements**)
  - Numerical Integration (important): adaptive numerical integration
  - Numerical Differentiation (critical): proper discrete differential operator
  - Algebraic solver (training NN) (critical): iterative solvers ???

Least-Squares Neural Network (LSNN) method

- **Linear advection-reaction problem**

$$u_{\boldsymbol{\beta}} + \gamma\, u = f \text{ in } \Omega, \qquad u\big|_{\Gamma_-} = g$$

- **Least-squares formulation** Find $u \in V_{\boldsymbol{\beta}}(\Omega) = \{v \in L^2(\Omega) : v_{\boldsymbol{\beta}} \in L^2(\Omega)\}$ such that

$$\mathcal{L}(u; \mathbf{f}) = \min_{v \in V_{\boldsymbol{\beta}}} \mathcal{L}(v; \mathbf{f})$$

where $\mathcal{L}(v; \mathbf{f}) = \|v_{\boldsymbol{\beta}} + \gamma\, v - f\|^2_{0,\Omega} + \|v - g\|^2_{-\boldsymbol{\beta}}$

- **Coercivity and continuity** there exists positive constants α and M such that

$$\alpha\, |||v|||^2_{\boldsymbol{\beta}} \le \mathcal{L}(v; \mathbf{0}) \le M\, |||v|||^2_{\boldsymbol{\beta}}$$

De Sterck-Manteuffel-McCormick-Olson, 2004

Least-squares neural network (LSNN) method

- **LSNN method** find $u_N \in \mathcal{M}(d,n)$ such that

$$\mathcal{L}(u_N, \mathbf{f}) = \min_{v \in \mathcal{M}(d,n)} \mathcal{L}(v, \mathbf{f})$$

where $\mathcal{M}(d, 1, \lceil \log_2(d+1) \rceil + 1, n) = \mathcal{M}(d,n)$

- **Quasi-optimal approximation**

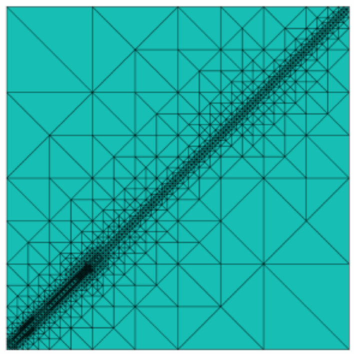
$$|||u - u_N|||_{\boldsymbol{\beta}} \le \left(\frac{M}{\alpha}\right)^{1/2} \inf_{v \in \mathcal{M}(d,n)} |||u - v|||_{\boldsymbol{\beta}}.$$

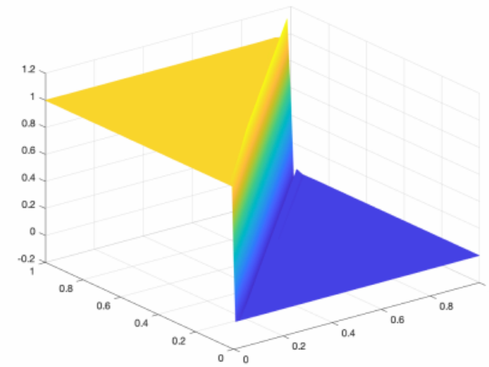
- **A priori error estimate**

$$|||u - u_N|||_{\boldsymbol{\beta}} \le C \left(|\alpha_1 - \alpha_2| \sqrt{\varepsilon} + \inf_{v \in \mathcal{M}(d,n)} |||\hat{u} + p - v|||_{\boldsymbol{\beta}} \right)$$

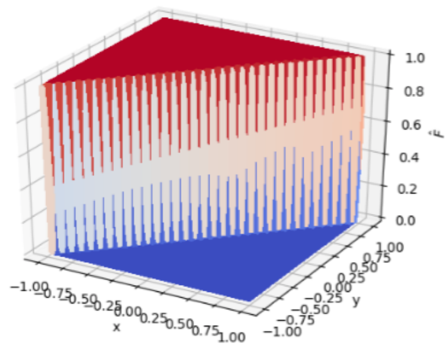
C.-Chen-Liu, JCP, 443 (2021), 110514; C.-Choi-Liu (2022)

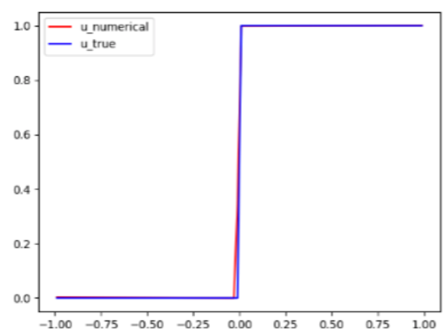
Famous Transport Equation $u_t+u_x=0$

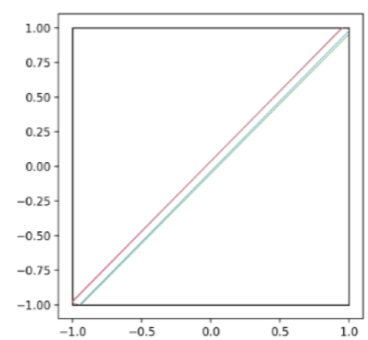




Liu-Zhang, CMAME, 2020

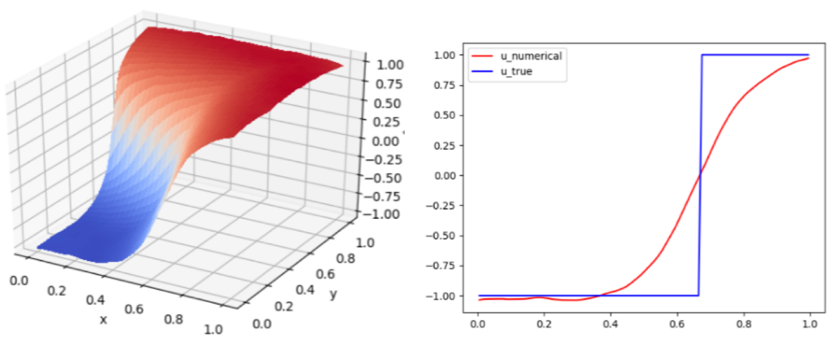




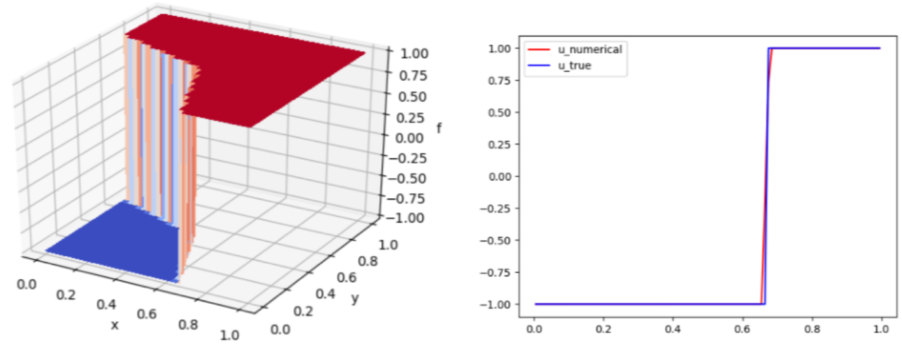


(2-6-1)

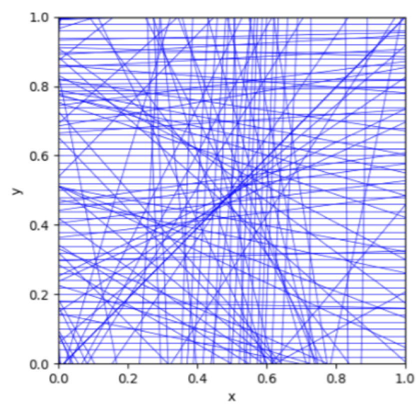
C.-Chen-Liu, JCP, 2021



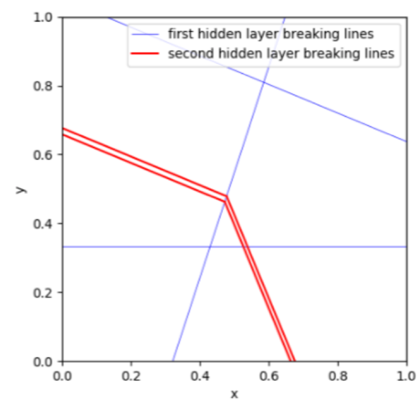
(2-200-1)



(2-5-5-1)



(e) 2-layer NN breaking lines



(f) 3-layer NN breaking lines

C.-Chen-Liu, LSNN method for linear advection-reaction equation, JCP, 443(2021), 110514.

Least-Squares Neural Network (LSNN) method

- **Scalar nonlinear hyperbolic conservation laws**

$$u_t(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot \mathbf{f}(u) = 0, \text{ in } \Omega \times I, \quad u\big|_{\Gamma_-} = g, \quad u(\mathbf{x},0)\big|_{\Omega} = u_0(\mathbf{x})$$

- **Least-squares formulation**

Find $u \in V_{\mathbf{f}} = \big\{v \in L^2(\Omega \times I) | \, (\mathbf{f}(v), v) \in H(\text{div}; \Omega \times I)\big\}$ such that

$$\mathcal{L}\big(u; \mathbf{g}\big) = \min_{v \in V_{\mathbf{f}}} \mathcal{L}\big(v; \mathbf{g}\big)$$

where $\mathcal{L}(v;\mathbf{g}) = \|v_t + \nabla_{\mathbf{x}} \cdot \mathbf{f}(v)\|^2_{0,\Omega\times I} + \|v - g\|^2_{0,\Gamma_-} + \|v(\mathbf{x},0) - u_0(\mathbf{x})\|^2_{0,\Omega}$

- **Least-squares neural network (LSNN) method**

find $u_n \in \mathcal{M}(d,l) \subset V_{\mathbf{f}}$ such that $\quad u_n(\mathbf{x},t) = \underset{v \in \mathcal{M}(d,l)}{\arg\min}\, \mathcal{L}(v;\mathbf{g})$

Discrete Divergence Operator

- **Divergence operator**

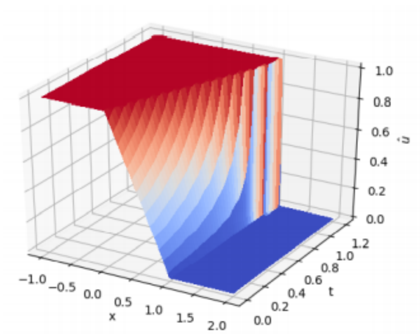
$$0 = u_t + \nabla \cdot \mathbf{f}(u) = \textbf{div}\,(\mathbf{f}(u), u) = \textbf{div}\,\mathbf{F}(u)$$

- **Discrete divergence operator**
  - + **based on conservative numerical schemes (C.-Chen-Liu, ANM(2022))**
  - + **new discrete divergence operator (C.-Chen-Liu, J Comput Appl Math (2023))**

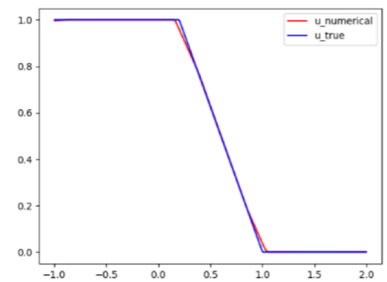
Let $\mathcal{T}$ be a partition of the domain $\Omega \subset \mathbb{R}^{d+1}$.

For any $K \in \mathcal{T}$, let $\mathbf{z}_K$ be the centroid of K.

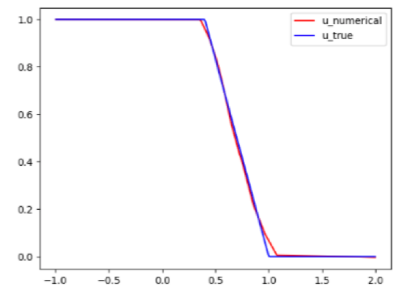
$$\mathbf{div}_{\mathcal{T}}\mathbf{F}\big(u(\mathbf{z}_K)\big) \approx \mathrm{avg}_K \mathbf{div}\,\mathbf{f}(u) = \frac{1}{|K|}\int_{\partial K} \mathbf{F}(u) \cdot \mathbf{n}\, dS$$



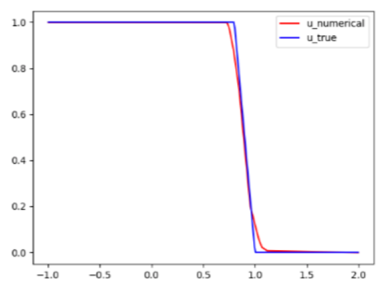
(a) Exact solution u on $\Omega \times I$



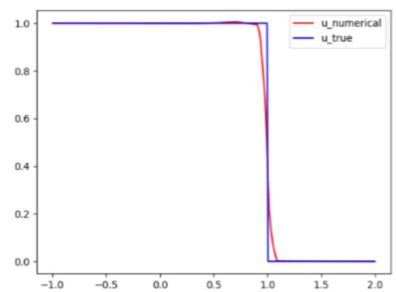
(a) Traces of exact solution and approximation $u_{1,\mathcal{T}}$ on the plane $t = 0.2$



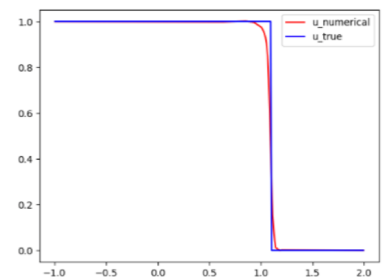
(c) Traces of exact and numerical solutions $u_{2,\mathcal{T}}$ on the plane $t = 0.4$



(d) Traces of exact solution and approximation $u_{4,\mathcal{T}}$ on the plane $t = 0.8$



(e) Traces of exact solution and approximation $u_{5,\mathcal{T}}$ on the plane $t = 1.0$

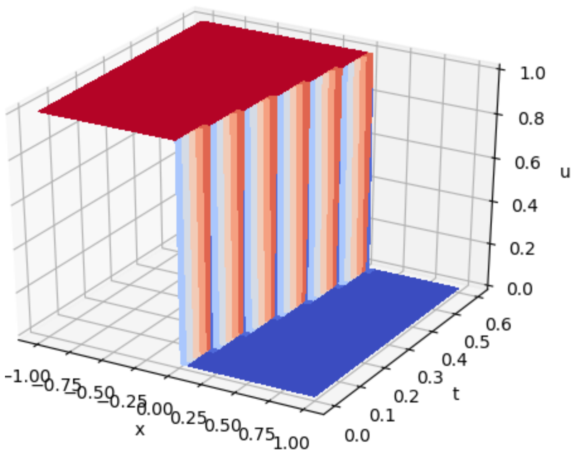


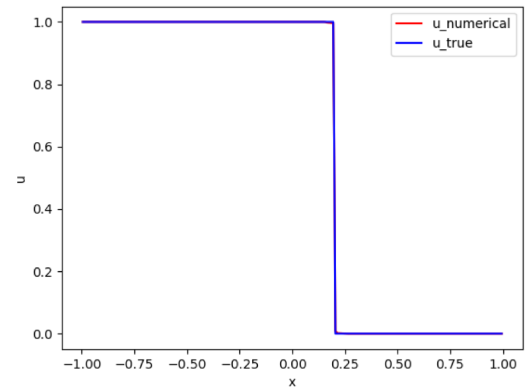
(f) Traces of exact solution and approximation $u_{6,\mathcal{T}}$ on the plane $t = 1.2$

C.-Chen-Liu, Applied Numer. Math., 174 (2022), 163-176

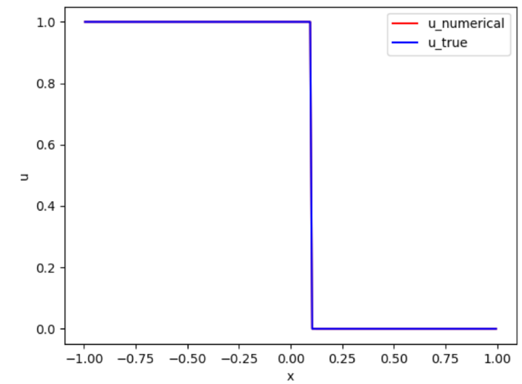
Inviscid Burger Equation $f(u) = \frac{1}{2}u^2$

Riemann Problem Shock formation: exact solution

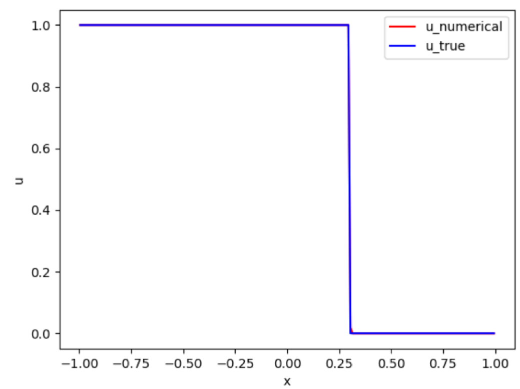




t=0.2



t=0.4

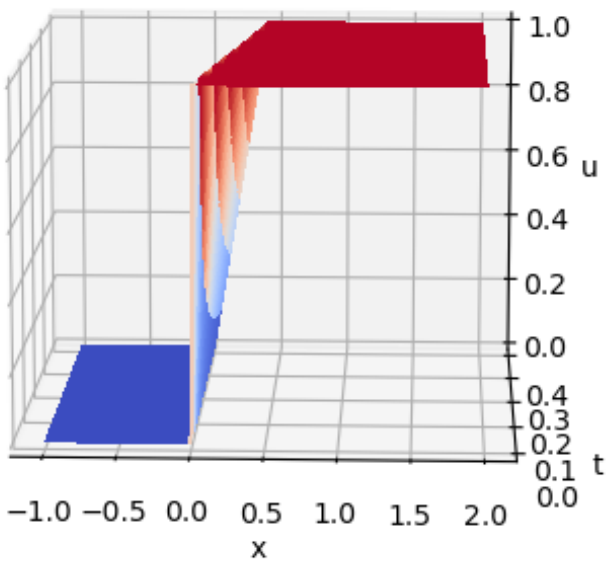


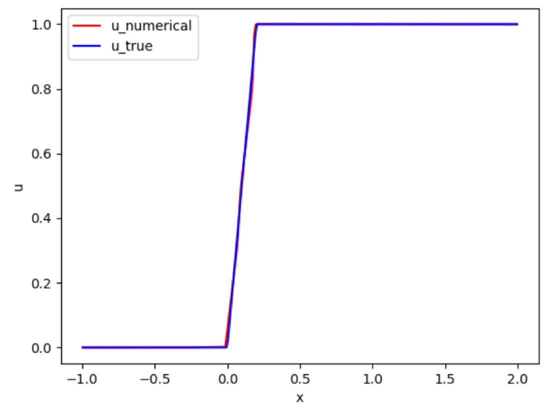
t=0.6 (2-10-10-1)

PURDUE UNIVERSITY | Department of Mathematics

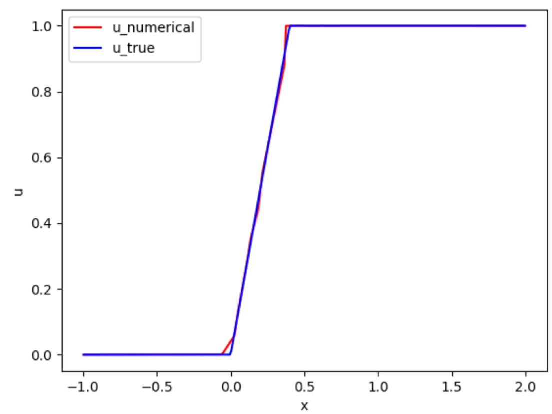
C.-Chen-Liu, arXiv: 2110.10895 [math.NA]

Riemann Problem
Rarefaction wave: exact solution





t=0.2



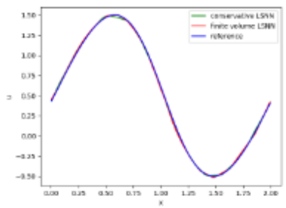
t=0.4

(2-10-10-1)

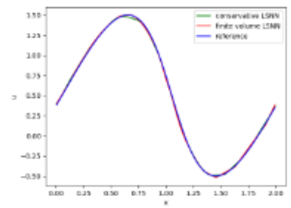
Inviscid Burgers equation with smooth initial

$$u_0(x) = 0.5 + \sin(\pi x).$$

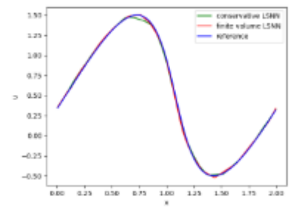
(2-30-30-1)



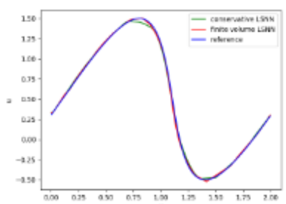
(a) Traces of reference and numerical solutions $u_{1,\mathcal{T}}$ on the plane $t = 0.05$



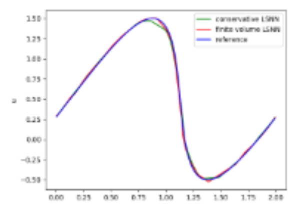
(b) Traces of reference and numerical solutions $u_{2,\mathcal{T}}$ on the plane $t = 0.1$



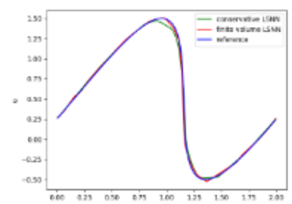
(c) Traces of reference and numerical solutions $u_{3,\mathcal{T}}$ on the plane $t = 0.15$



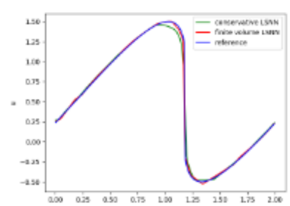
(d) Traces of reference and numerical solutions $u_{4,\mathcal{T}}$ on the plane $t = 0.2$



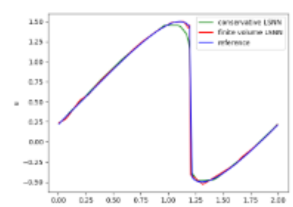
(e) Traces of reference and numerical solutions $u_{5,\mathcal{T}}$ on the plane $t = 0.25$



(f) Traces of reference and numerical solutions $u_{6,\mathcal{T}}$ on the plane $t = 0.3$



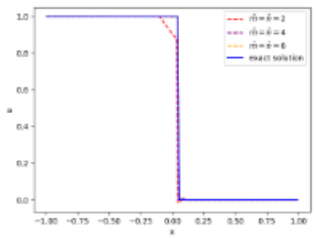
(g) Traces of reference and numerical solutions $u_{7,\mathcal{T}}$ on the plane $t = 0.35$



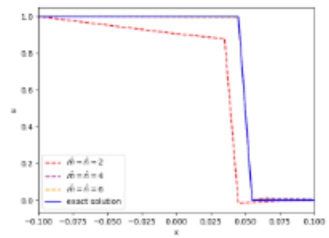
(h) Traces of reference and numerical solutions $u_{8,\mathcal{T}}$ on the plane $t = 0.4$

FIG. 3. *Approximation results of Burgers' equation with a sinusoidal initial condition*

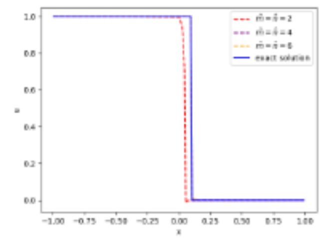
Riemann Problem with Higher order flux $f(u) = \frac{1}{4}u^4$



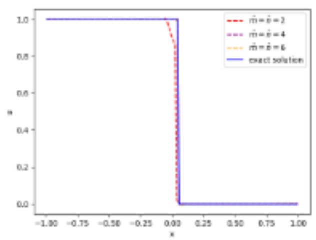
(a) Traces of exact and numerical solutions $u_{1,\mathcal{T}}$ using the trapezoidal rule on the plane $t = 0.2$



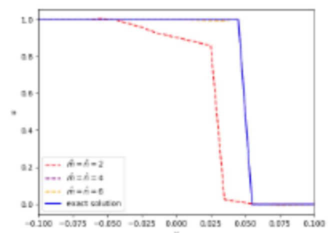
(b) Zoom-in plot near the discontinuous interface of sub-figure (a)



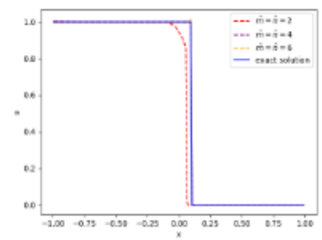
(c) Traces of exact and numerical solutions $u_{2,\mathcal{T}}$ using the trapezoidal rule on the plane $t = 0.4$



(d) Traces of exact and numerical solutions $u_{1,\mathcal{T}}$ using the mid-point rule on the plane $t = 0.2$



(e) Zoom-in plot near the discontinuous interface of sub-figure (d)

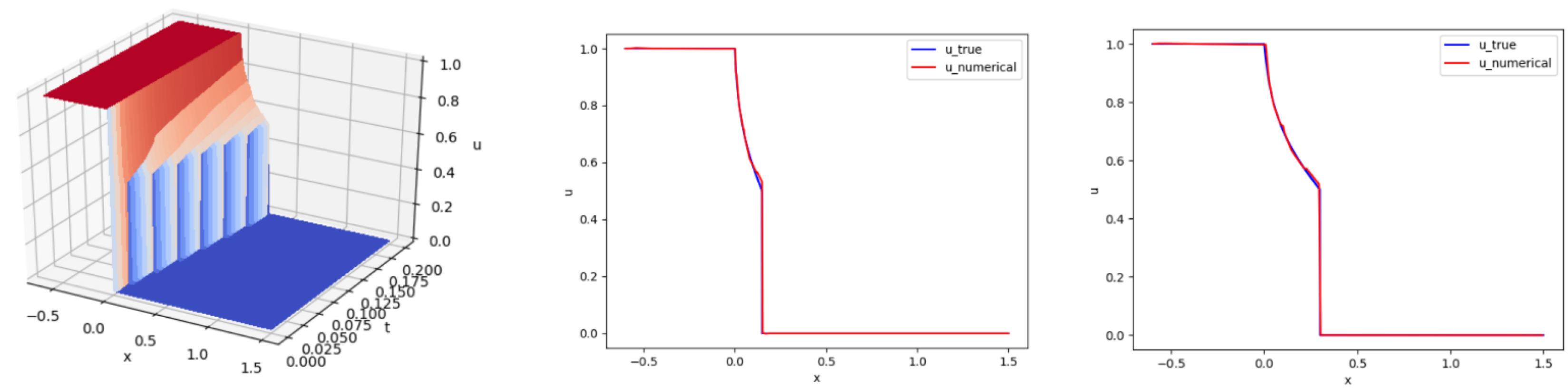


(f) Traces of exact and numerical solutions $u_{2,\mathcal{T}}$ using the mid-point rule on the plane $t = 0.4$

FIG. 5. *Numerical results of the problem with* $f(u) = \frac{1}{4}u^4$ *using the composite trapezoidal and mid-point rules*

(2-10-10-1)

Buckley-Leverett Problem $f(u) = u(1-u)/[u^2 + a(1-u)^2]$



(a) Numerical solution u_N on Ω (b) Traces at $t = 0.1$ (c) Traces at $t = 0.2$

FIG. 6. *Numerical results of Buckley-Leverett Riemann problem*

Evolving Neural Network (ENN) Method (C. and B. Hejnal)

- **One-Dimensional Scalar Nonlinear Hyperbolic Conservation Laws**

$$\begin{cases} \dfrac{\partial}{\partial t}u(x,t) + \dfrac{\partial}{\partial x}f\big(u(x,t)\big) & = & 0, & \text{in } \ \Omega \times (0,T), \\ u(x,t) & = & g(t), & \text{on } \ \Gamma_-, \\ u(x,0) & = & u_0(x), & \text{in } \ \Omega \end{cases}$$

- **Characteristic Line**

$$\begin{cases} \dfrac{d}{dt}x(t) & = & f'\big(u(x(t),t)\big) \\ x(t_0) & = & x_0 \end{cases}$$

$$\frac{d}{dt}u(x(t),t) = 0$$

$$x(t) = x_0 + (t - t_0)\, f'\big(u(x_0, t_0)\big)$$

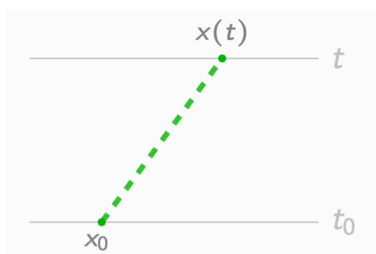
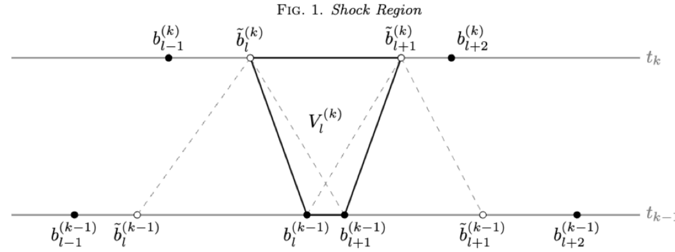


FIG. 1. *Shock Region*



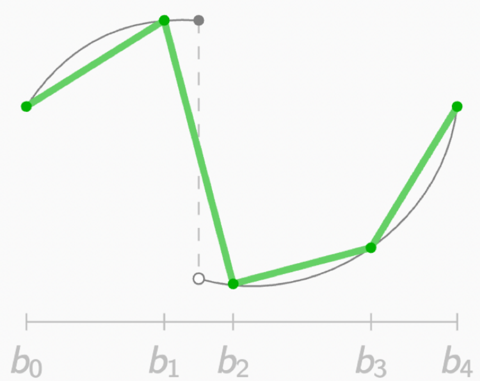
Representation of Initial Data

Set of Neural Network Functions:

$$M_n = \left\{ N(x) = c_{-1} + \sum_{i=0}^{n} c_i \sigma(\omega_i x - b_i) \,:\, b_i,\, c_i,\, \omega_i \in \mathbb{R} \right\}$$

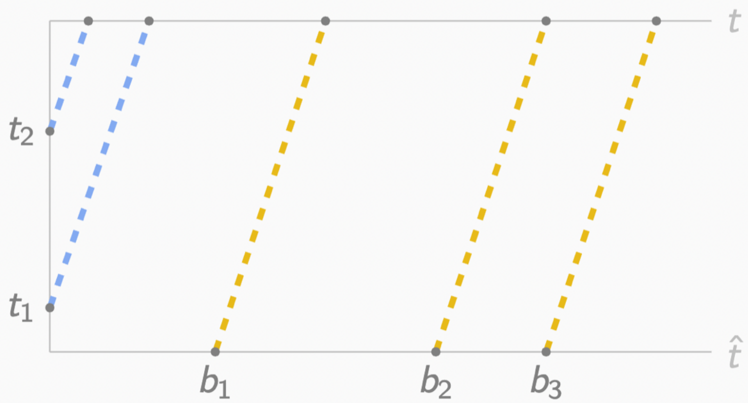
Least-Squares Approximation: Find $u_N^{(0)} \in M_n$ such that

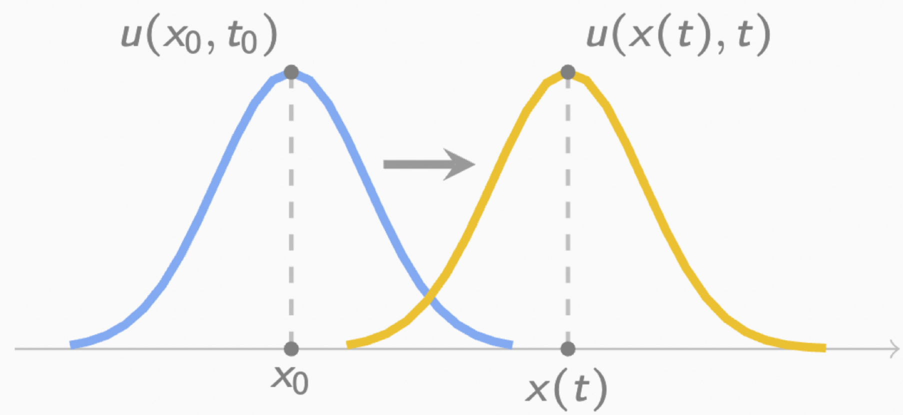
$$\left\| u_N^{(0)} - u_0 \right\|_{L^2(\Omega)} = \min_{v \in M_n} \left\| v - u_0 \right\|_{L^2(\Omega)}$$



Characteristic Scheme

- Propagate breaking points of the initial and boundary data

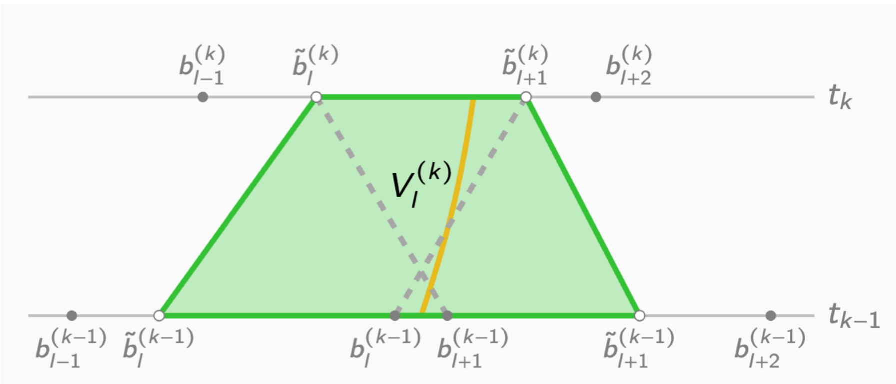




- Error Estimate

$$\left\| u(\cdot, t_k) - u_N^{(k)} \right\|_{L^p(\Omega)} \leq \left(\left\| u_0 - u_N^{(0)} \right\|_{L^p(\Omega)}^p + \| g - g_N \|_{L^p(I)}^p \right)^{1/p}$$

Finite Volume Characteristic Scheme



$$\int_{\partial V_l^{(k)}} \begin{pmatrix} \boldsymbol{f}(\boldsymbol{u}) \\ \boldsymbol{u} \end{pmatrix} \cdot \hat{\boldsymbol{n}}\, ds = 0$$

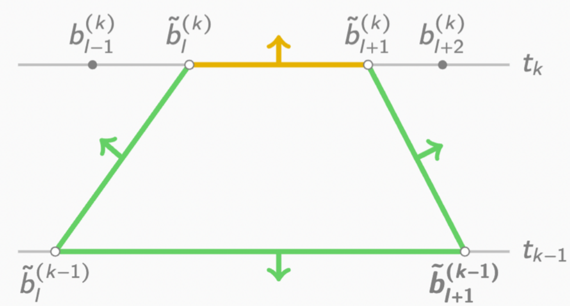
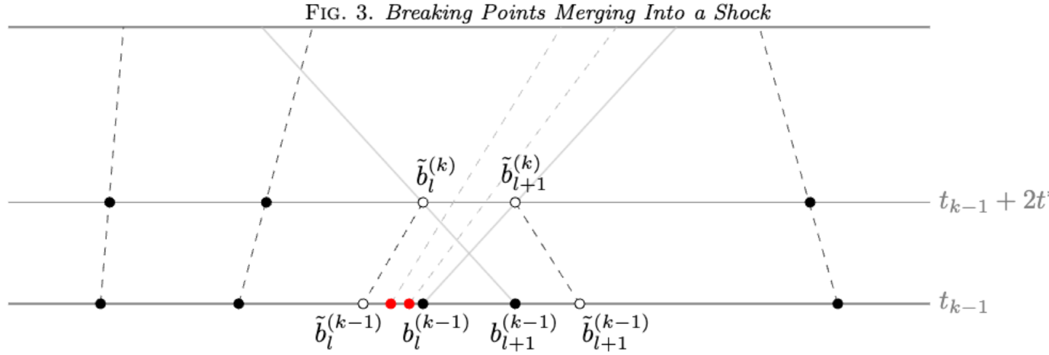
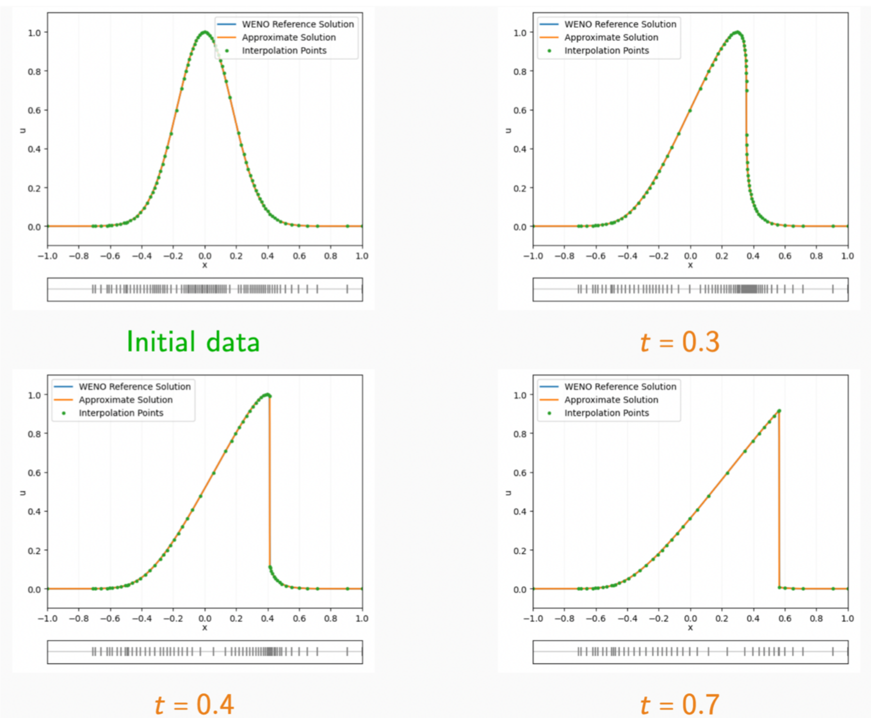


Fig. 3. *Breaking Points Merging Into a Shock*



Shock Formation (exponential initial profile)

Inviscid Burgers' Equation



Initial data; $t = 0.3$; $t = 0.4$; $t = 0.7$

Time	$\frac{\lVert \tilde{u}(\cdot,t_k) - u_N^{(k)} \rVert_{L^2(\Omega)}}{\lVert \tilde{u}(\cdot,t_k) \rVert_{L^2(\Omega)}}$	n_k
0.0	6.6207×10^{-4}	83
0.2	7.2902×10^{-4}	83
0.4	1.2718×10^{-2}	61
0.6	2.1803×10^{-2}	47
0.8	2.0423×10^{-2}	40
1.0	1.4822×10^{-2}	37

ENN

- **83** breaking points
- **418** time steps

WENO

- **2000** mesh points
- **5000** time steps

Shock Formation (sinusoidal initial profile)

Inviscid Burgers' Equation

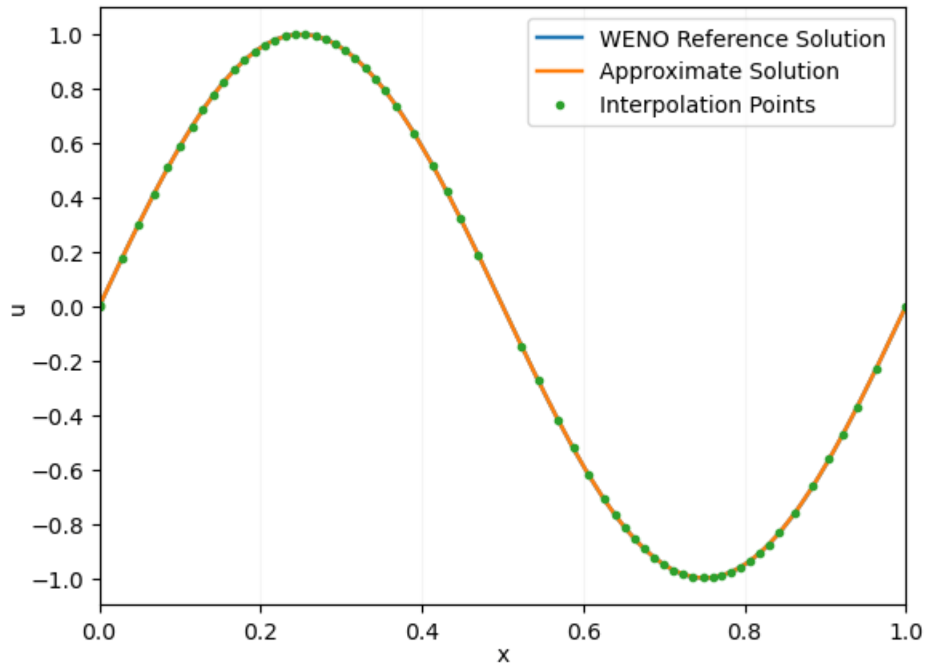
Time	$\frac{\|\tilde{u}(\cdot,t_k)-u_N^{(k)}\|_{L^2(\Omega)}}{\|\tilde{u}(\cdot,t_k)\|_{L^2(\Omega)}}$	n_k
0.0	6.6923×10^{-4}	78
0.1	7.8352×10^{-4}	78
0.2	4.0166×10^{-2}	56
0.3	5.1491×10^{-2}	38
0.4	5.3515×10^{-2}	30
0.5	5.4162×10^{-2}	25

ENN

- **78** breaking points
- **587** time steps

WENO

- **1000** mesh points
- **2500** time steps

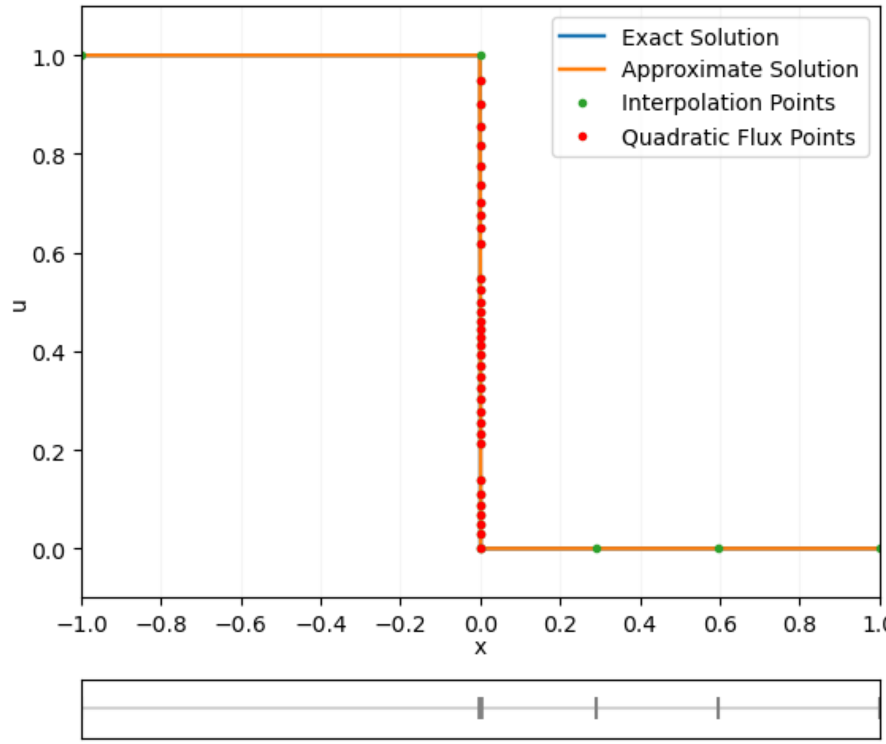


Compound Wave

Buckley-Leverett Equation

$$f(u) = \frac{u^2}{u^2 + \frac{1}{2}(1-u)^2}$$

Time	$\frac{\lVert u(\cdot,t_k) - u_N^{(k)} \rVert_{L^2(\Omega)}}{\lVert u(\cdot,t_k) \rVert_{L^2(\Omega)}}$	n_k
0.00	1.3732×10^{-2}	40
0.25	6.9084×10^{-3}	16
0.50	5.8335×10^{-3}	15



Summary

- **NN provides a new class of approximating functions**

  "Moving" mesh vs uniform mesh and adaptive mesh

- **Scalar hyperbolic conservation laws**

  LSNN (a space-time approach)

  No numerical artifacts such as overshooting, oscillation, or smearing
  Complicated and expensive iterative solvers

  ENN (a time-marching approach)

  Super accurate and efficient for 1D scalar HCLs comparing with existing methods
  Extension to multi-dimension?

THANK YOU