

# Controlling Oscillations in High-order Accurate Methods Through Artificial Neural Networks

Deep Ray

EPFL, Switzerland

[deep.ray@epfl.ch](mailto:deep.ray@epfl.ch)

<http://deepray.github.io>

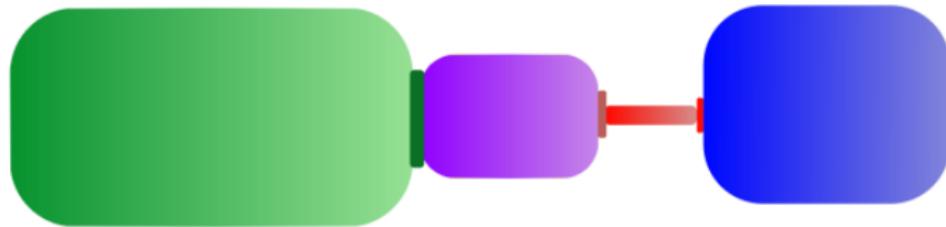


*with Jan S. Hesthaven, Niccolò Discacciati and Jian Yu*

SIAM CSE  
Spokane, 28 February 2019

# Motivation

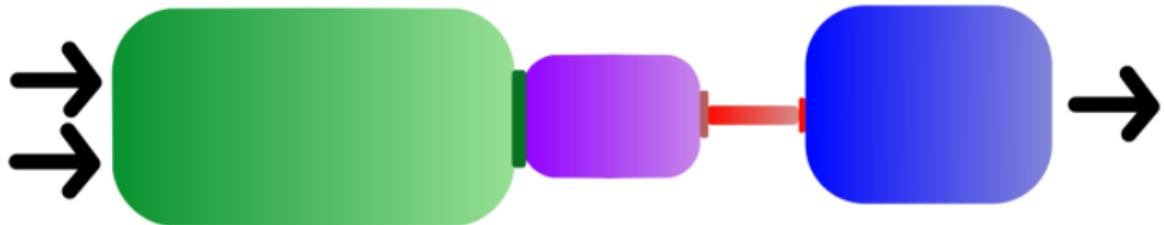
Robust numerical machinery developed over the past several decades



# Motivation

Robust numerical machinery developed over the past several decades

Gives satisfactory results, but ...

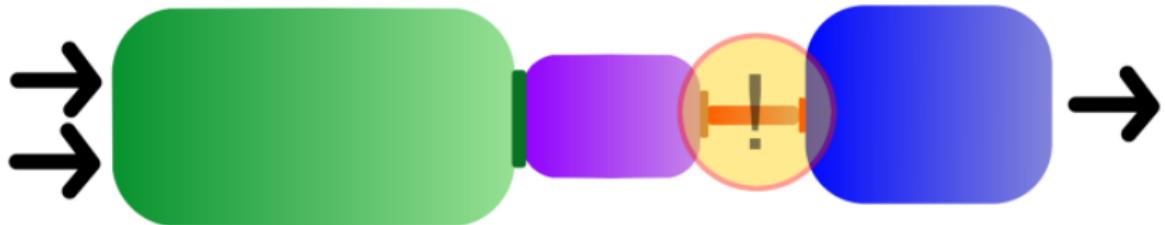


## Motivation

Robust numerical machinery developed over the past several decades

Gives satisfactory results, but ...

Bottlenecks exist – computationally expensive subcomponents, problem dependent parameters, etc

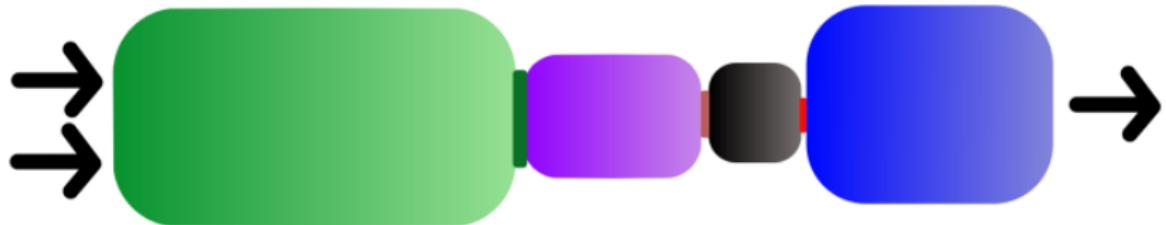


# Motivation

Robust numerical machinery developed over the past several decades

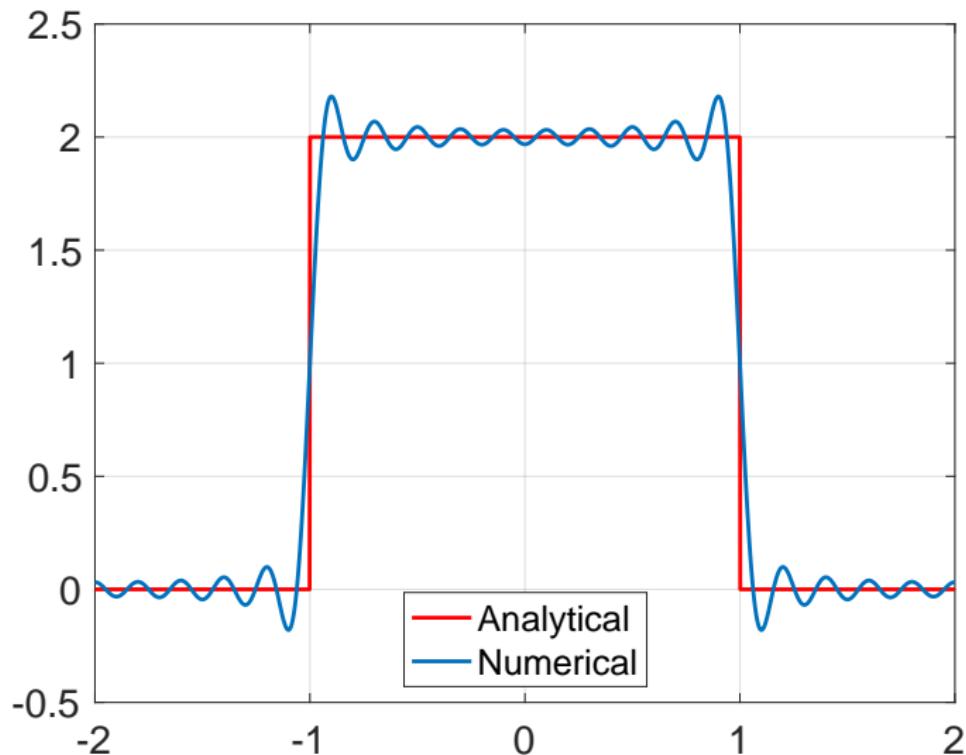
Gives satisfactory results, but ...

Bottlenecks exist – computationally expensive subcomponents, problem dependent parameters, etc



**Strategy:** Replace (**only**) the problematic part with Deep Learning black box

# The menace of Gibbs oscillations



# The menace of Gibbs oscillations

What can be done:

- Trouble-cell indicator and limiting
- Add artificial viscosity based on an estimation of local smoothness

# The menace of Gibbs oscillations

What can be done:

- Trouble-cell indicator and limiting  
→ Classification
- Add artificial viscosity based on an estimation of local smoothness

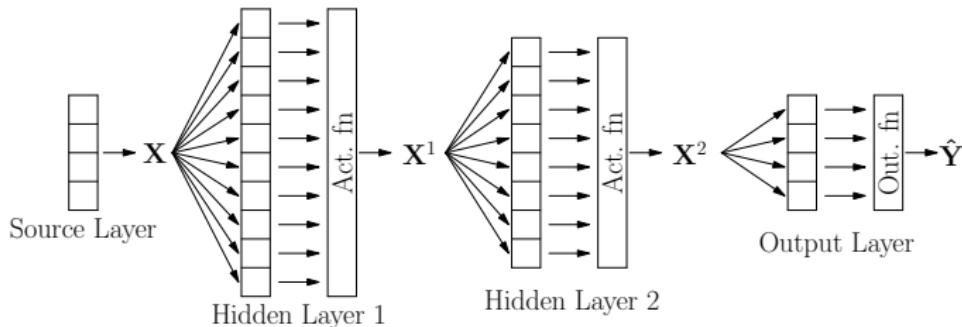
# The menace of Gibbs oscillations

What can be done:

- Trouble-cell indicator and limiting  
→ Classification
- Add artificial viscosity based on an estimation of local smoothness  
→ Regression

# Neural networks

Several layers of neurons



Feed forward network - **multilayer perceptron (MLP)**

Weights/biases computed using labeled data ( $\mathbf{X}, \mathbf{Y}$ )

→ supervised learning

Choose: architecture, training/validation/test data sets, loss function ...

## Part I: ANNs as troubled-cell indicators

# Conservation laws

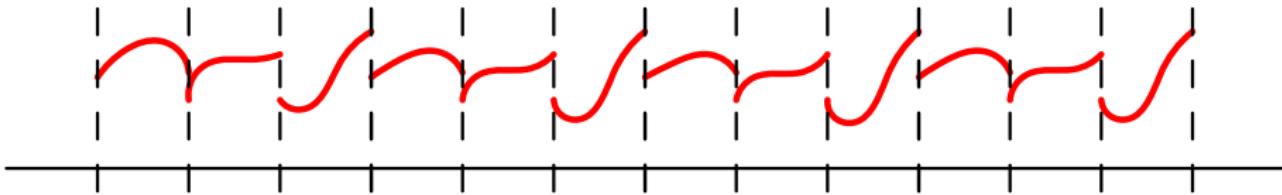
Consider the system of conservation laws

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}) = 0$$
$$\mathbf{u}(x, 0) = \mathbf{u}_0(x)$$

Non-linearity  $\implies$  Discontinuities in finite time

Solve using DG methods

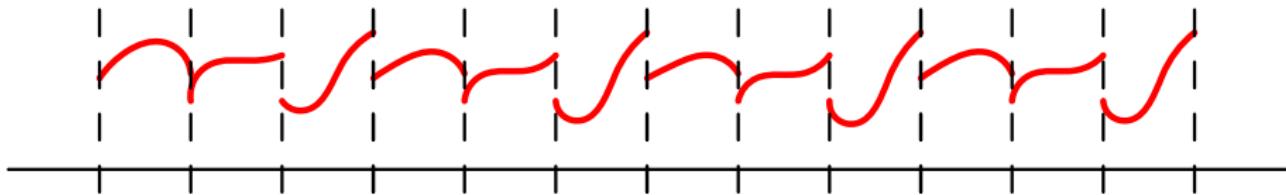
→ solution approximated locally using polynomials



# Limiting

Strategy for limiting

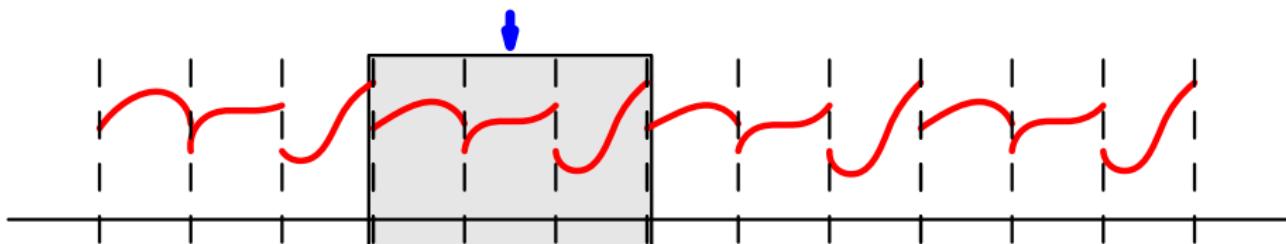
- ① Identify troubled-cells



# Limiting

Strategy for limiting

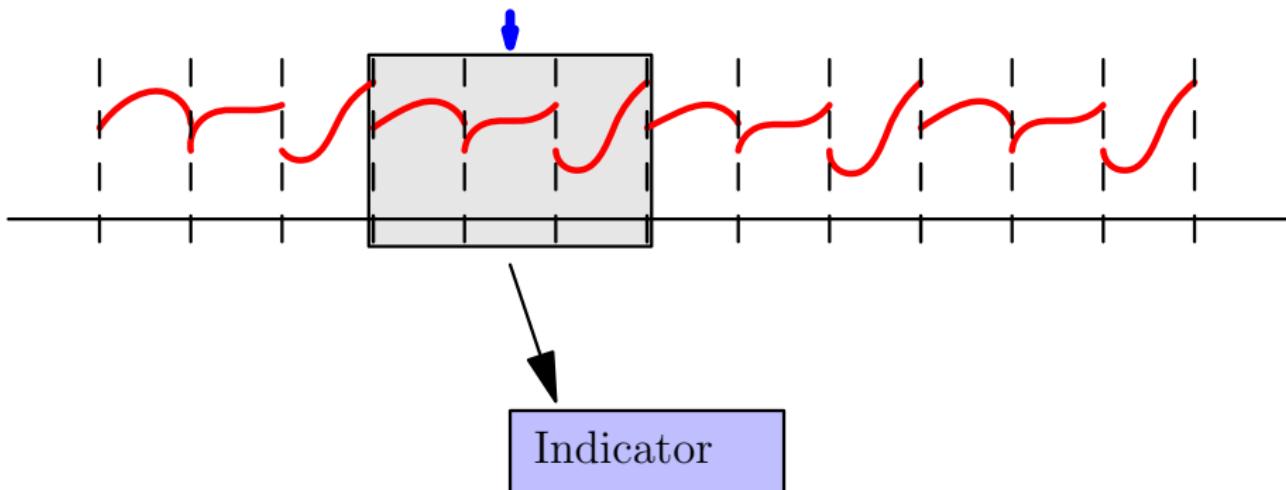
- ① Identify troubled-cells



# Limiting

Strategy for limiting

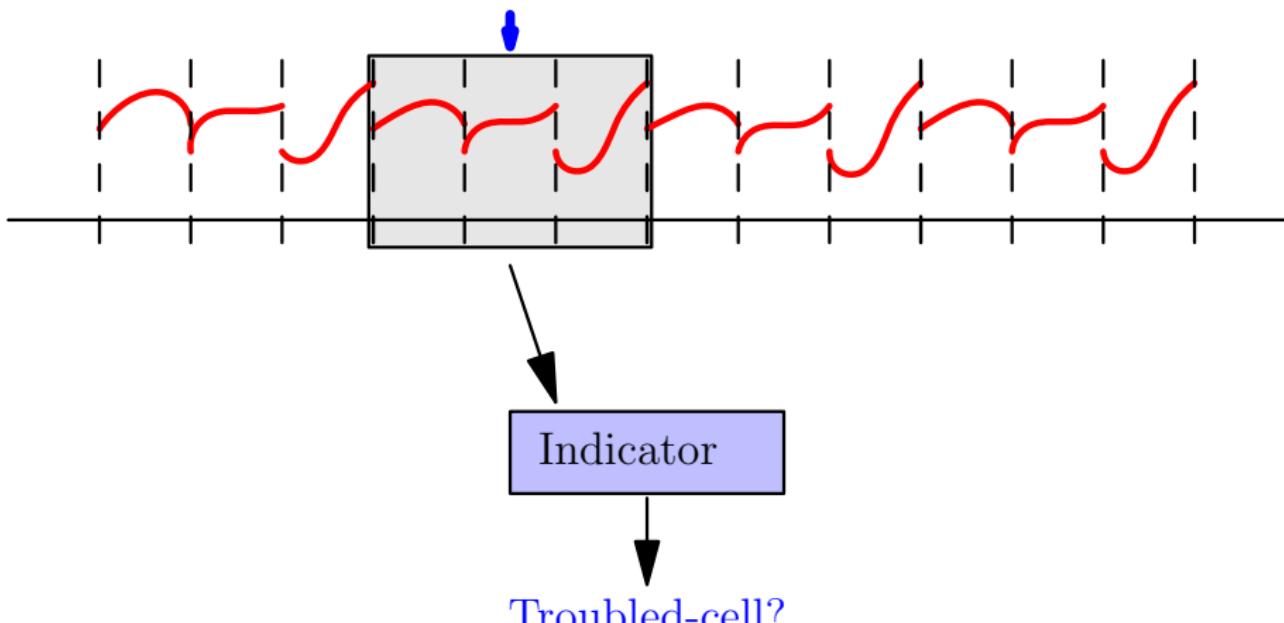
- ① Identify troubled-cells



# Limiting

Strategy for limiting

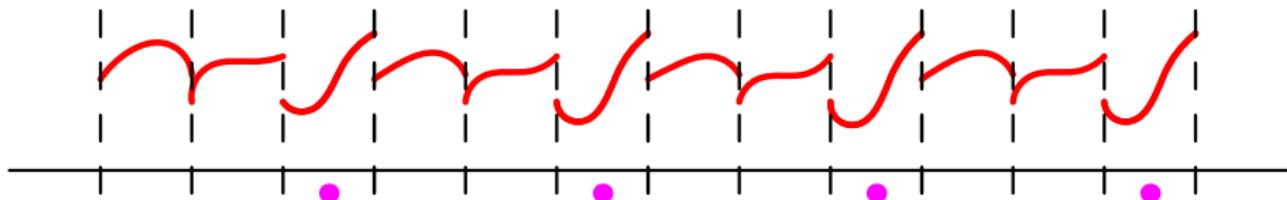
- ① Identify troubled-cells



# Limiting

Strategy for limiting

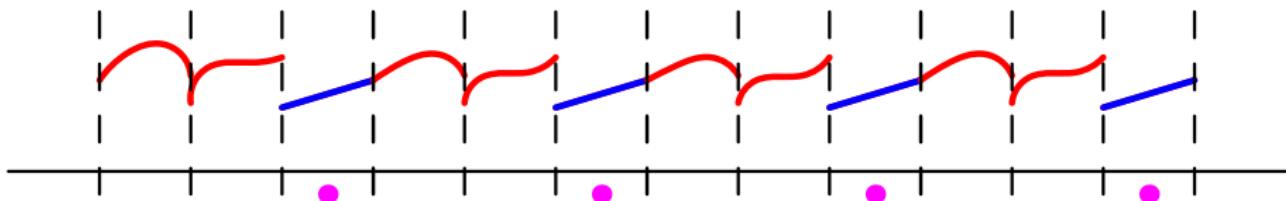
- ① Identify troubled-cells



# Limiting

Strategy for limiting

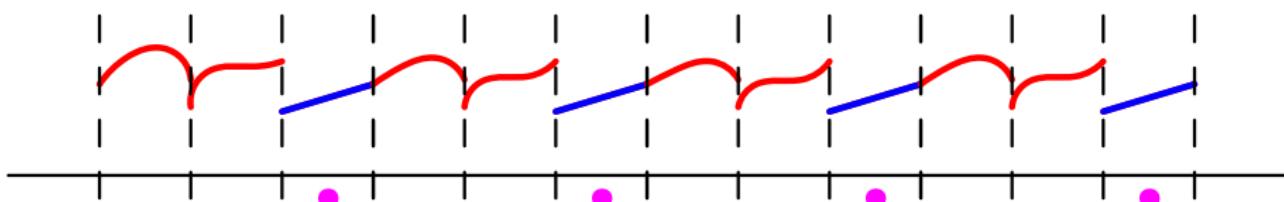
- ① Identify troubled-cells
- ② Limit solution in flagged cells



# Limiting

Strategy for limiting

- ① Identify troubled-cells
- ② Limit solution in flagged cells



Some issues:

- Problem-dependent parameters
- If insufficient cells marked → re-appearance of Gibbs oscillations
- If excessive cells marked
  - ▶ Unnecessary computational cost
  - ▶ Loss of accuracy for strong limiters

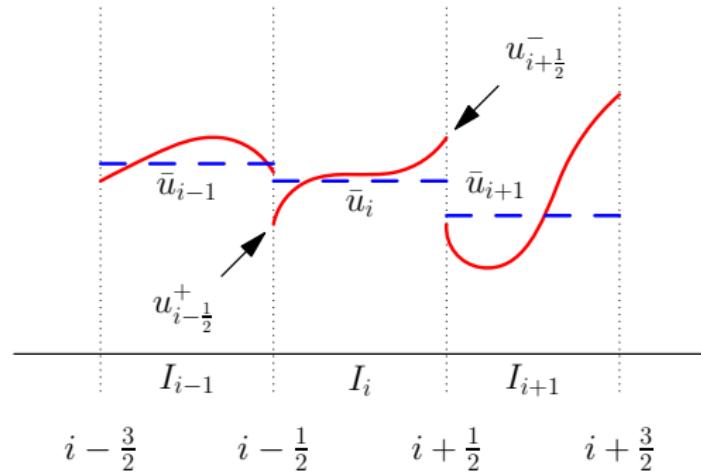
## Available troubled-cell indicators

- Minmod-based TVB limiter (Cockburn and Shu; Math. Comp. '98)
- Moment limiter (Biswas et al.; Appl. Numer. Math. '94)
- Modified moment limiter (Burbeau; JCP '01)
- Monotonicity preserving limiter (Suresh and Huynh; JCP '97)
- Modified MP limiter (Rider and Margolin; JCP '01)
- KXRCF indicator (Krivodonova et al.; App. Numer. Math. '04)
- Polynomial degree based limiter (Fu and Shu; JCP '17)
- Outlier detection using Tukey's boxplot method (Vuik and Ryan; J. Sci. Comp. '16)
- ...

## Available troubled-cell indicators

- Minmod-based TVB limiter (Cockburn and Shu; Math. Comp. '98)
- Moment limiter (Biswas et al.; Appl. Numer. Math. '94)
- Modified moment limiter (Burbeau; JCP '01)
- Monotonicity preserving limiter (Suresh and Huynh; JCP '97)
- Modified MP limiter (Rider and Margolin; JCP '01)
- KXRCF indicator (Krivodonova et al.; App. Numer. Math. '04)
- Polynomial degree based limiter (Fu and Shu; JCP '17)
- Outlier detection using Tukey's boxplot method (Vuik and Ryan; J. Sci. Comp. '16)
- ...

# TVB Indicator: Search for the elusive $M$



- For each cell  $I_i$ , get  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$ .
- Evaluate divided difference.
- Choose  $M \rightarrow$  problem dependent!!

**Objective:** Find a troubled-cell indicator which is:

- independent of problem-dependent parameters
- does not flag smooth extrema
- relatively inexpensive

# An MLP-based indicator

- Input  $\mathbf{X} = [\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-] \in \mathbb{R}^5$
- 5 Hidden Layers with width 256, 128, 64, 32, 16
- Leaky ReLU activation function with  $\nu = 10^{-3}$
- Softmax output function

$$\hat{Y}^{(k)} = \frac{e^{\hat{Y}^{(k)}}}{\sum_j e^{\hat{Y}^{(j)}}} \quad \in \quad [0, 1] \quad \longrightarrow \quad \text{probabilities/classification}$$

$$\text{Output } \hat{Y} = [\hat{Y}^{(0)}, \hat{Y}^{(1)}] \in [0, 1]^2$$

- Cost functional: L2 regularized cross-entropy

# Generating the training data

$u(x)$	Domain	Additional parameters varied	Good cells	Troubled cells
$\sin(4\pi x)$	$[0, 1]$	—	4470	0
$ax$	$[-1, 1]$	$a \in \mathbb{R}$	10000	0
$a x $	$[-1, 1]$	$a \in \mathbb{R}$	800	3200
$ul.(x < x_0) + ur.(x > x_0)$ (only troubled-cells selected)	$[-1, 1]$	$(u_l, u_r) \in [-1, 1]^2$ $x_0 \in [-0.76, 0.76]$	0	19800
			<b>15270</b>	<b>23000</b>

(a) Functions used to create  $\mathbb{T}$ .

$u(x)$	Domain	Additional parameters varied	Good cells	Troubled cells
$\sum_{p=1}^5 \sin(p\pi x)$	$[0, 2]$	—	3740	0
$\sin(2\pi x) \cos(3\pi x) \sin(4\pi x)$	$[0, 2]$	—	3740	0
$\sin(\pi x) + e^x$	$[-1, 1]$	—	3740	0
$ul.(x < x_0) + ur.(x > x_0)$ (only troubled-cells selected)	$[-1, 1]$	$(u_l, u_r) \in [-20, 20]^2$ $x_0 \in [-0.76, 0.76]$	0	13060
			<b>11220</b>	<b>13060</b>

(b) Functions used to create  $\mathbb{V}$ .

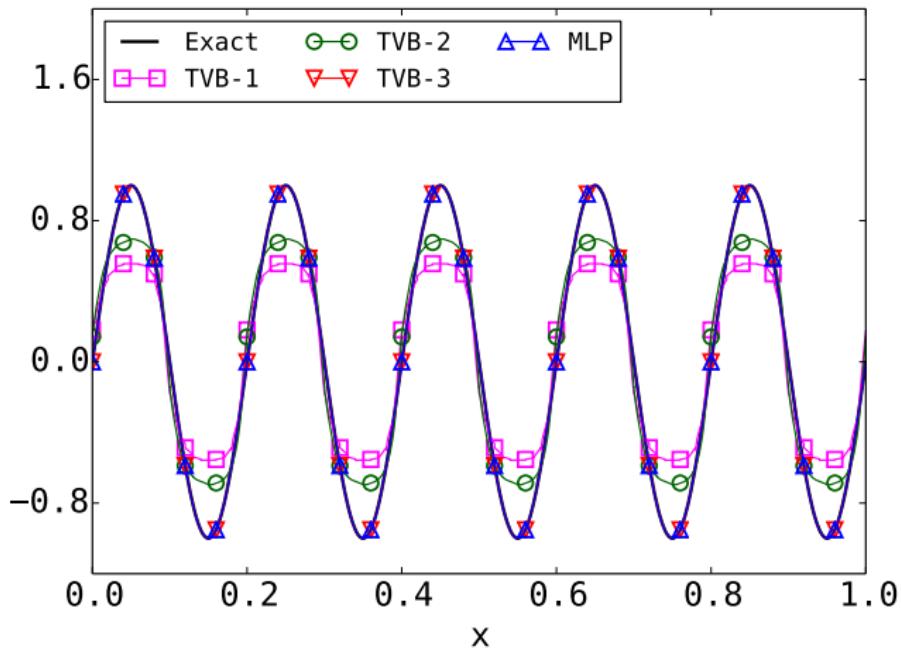
Parameters varied:

- Mesh size  $h$
- Approximating polynomial degree  $r$

# Linear advection: $u_t + u_x = 0$

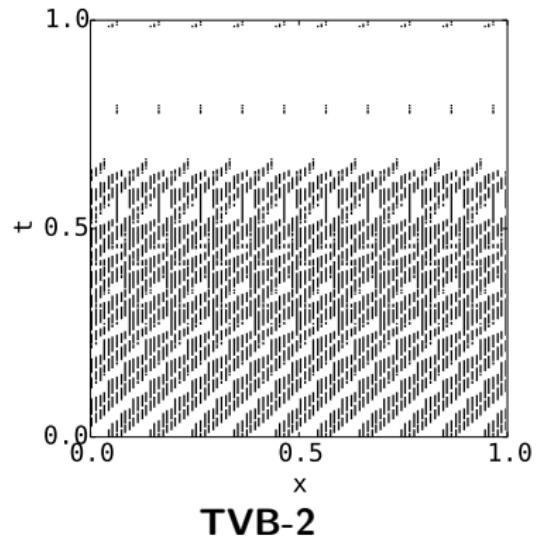
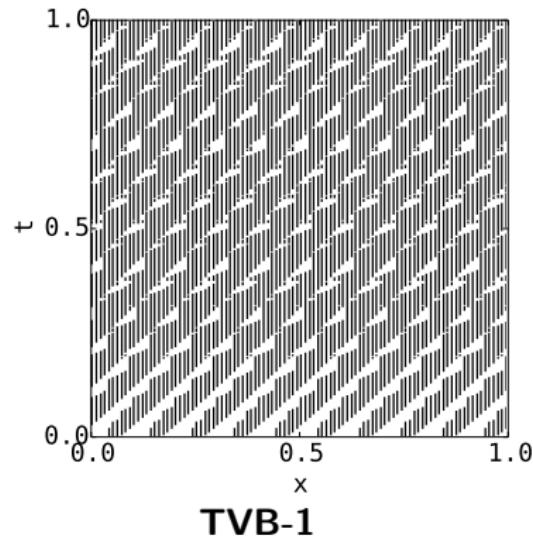
$$u_0(x) = \sin(10\pi x), \quad x \in [0, 1], \quad T_f = 1, \quad N = 100, \quad r = 4$$

TVB-1 → M=10  
TVB-2 → M=100  
TVB-3 → M=1000



Linear advection:  $u_t + u_x = 0$

$$u_0(x) = \sin(10\pi x), \quad x \in [0, 1], \quad T_f = 1, \quad N = 100, \quad r = 4$$

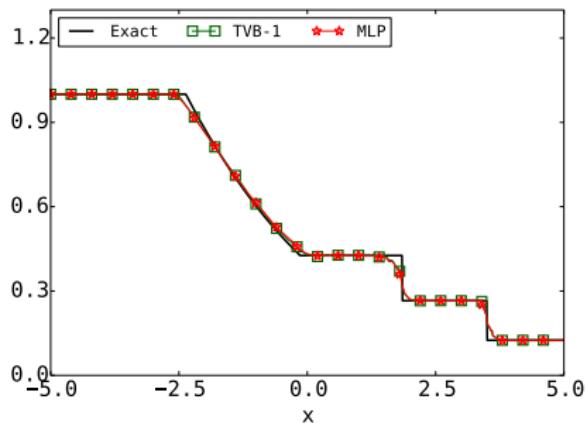
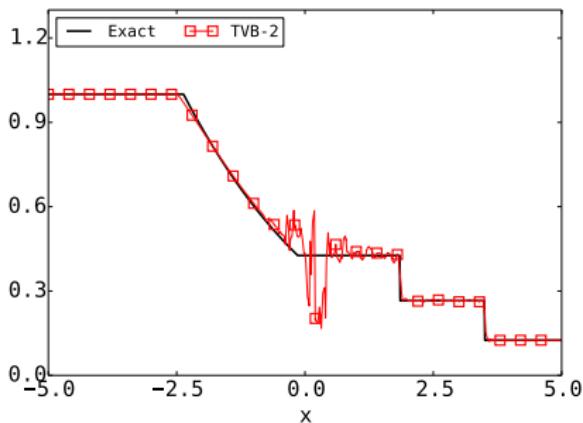


MLP and TVB-3 do not flag any cell

# Euler equations: Sod shock tube

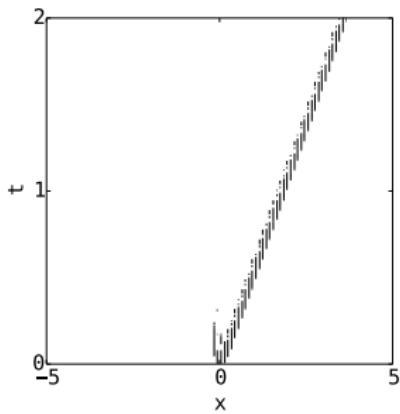
$$(\rho, u, p) = \begin{cases} (1, 0, 1) & \text{if } x < 0 \\ (0.125, 0, 0.1) & \text{if } x > 0 \end{cases}, \quad x \in [-1, 1]$$

$$T_f = 2, \quad N = 100, \quad r = 4$$

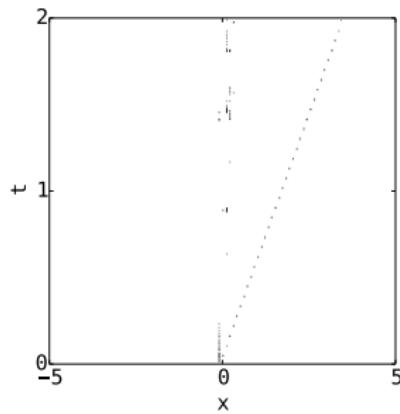


Loss of positivity with TVB-3

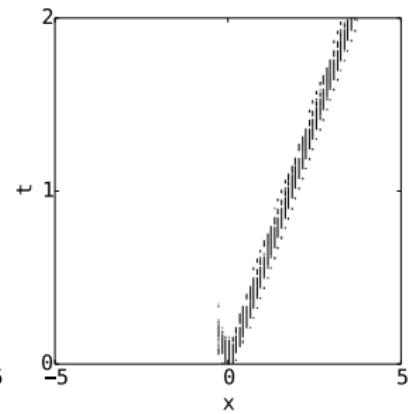
# Euler equations: Sod shock tube



TVB-1



TVB-2

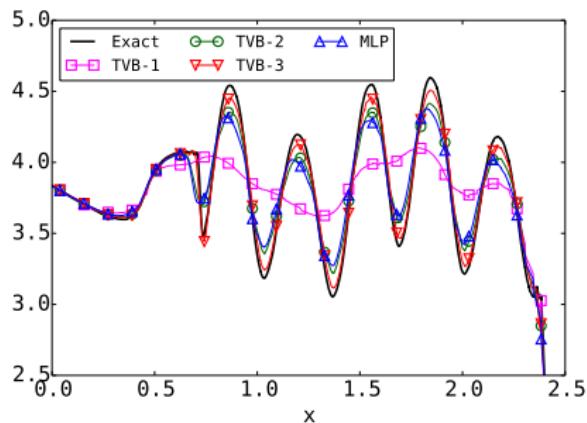
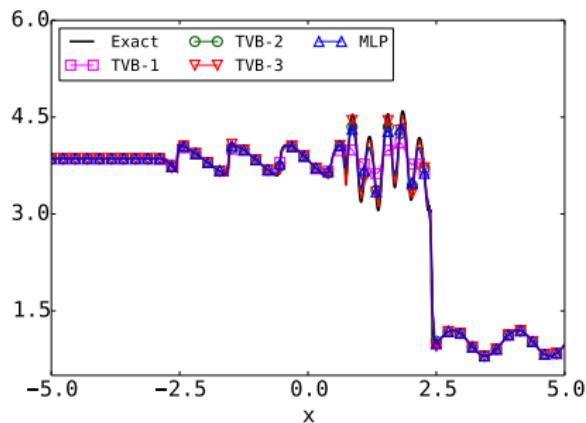


MLP

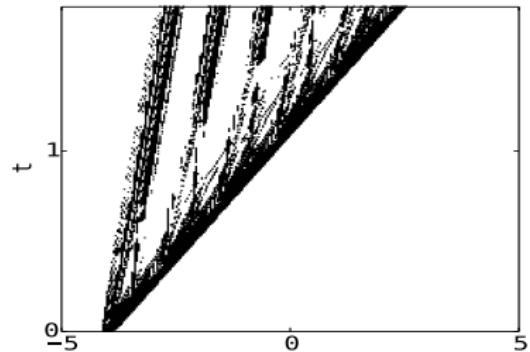
# Euler equations: Shock-entropy problem

$$(\rho, u, p) = \begin{cases} (3.857143, 2.629369, 10.33333) & \text{if } x < -4 \\ (1 + 0.2 \sin(5x), 0, 1) & \text{if } x > -4 \end{cases}$$

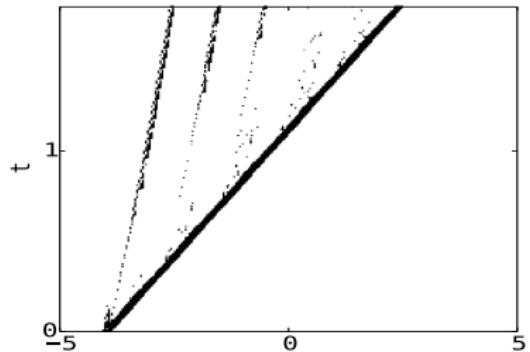
$$T_f = 1.8, \quad N = 256, \quad r = 4$$



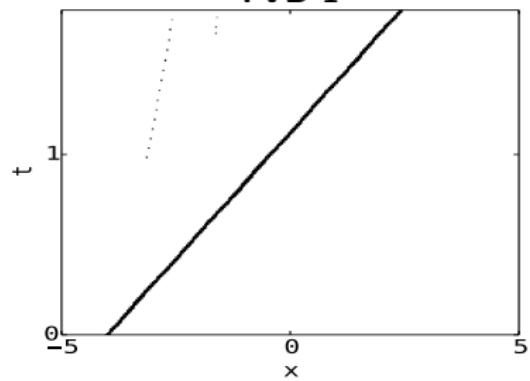
# Euler equations: Shock-entropy problem



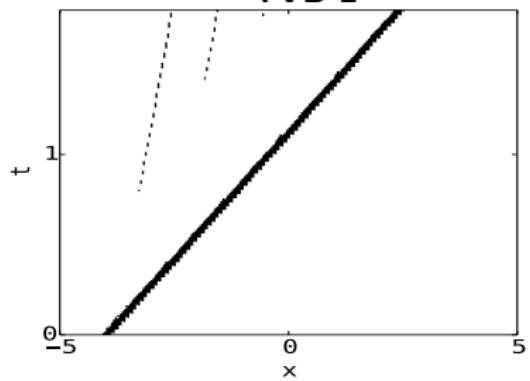
TVB-1



TVB-2



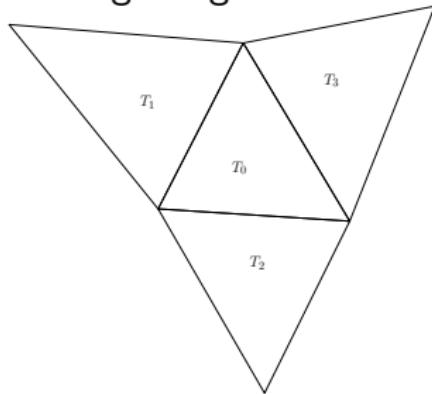
TVB-3



MLP

## Extension to 2D

We consider unstructured triangular grids



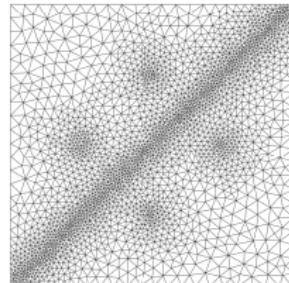
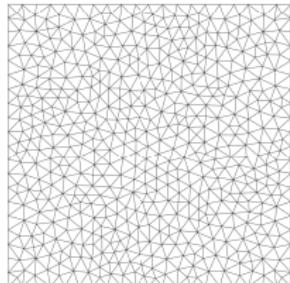
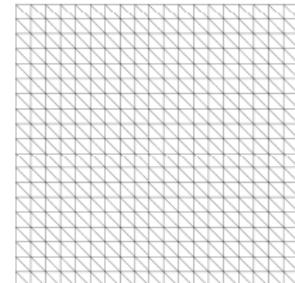
Training data constructed by:

- Interpolating functions to patch of 4 triangles.
- Extracting linear components, with  $\mathbf{X} \in \mathbb{R}^{12}$ .
- Label cell as troubled-cell if discontinuity present within circumscribed circle.

# Extension to 2D

## Training functions

$\mathbf{u}(\mathbf{x})$	Parameters	Good cells	Troubled cells
$ax + by$	$a, b \in \mathcal{U}[-10, 10]$	34,776	0
$\sum_{k=1}^{N_f} a_k \sin(k\pi x) + b_k \cos(k\pi y)$	$a_k, b_k \in \mathcal{U}[-1, 1], \quad N_f = 1, 2, 3$	202,980	0
$\exp(a_1[(x - a_2)^2 + (y - a_3)^2]) + \exp(a_4[(x - a_5)^2 + (y - a_6)^2])$	$a_i \in \mathcal{U}[-1, 1]$	135,320	0
4 values $u_1, u_2, u_3, u_4$ in 4 sections created by the lines $y - y_0 = m(x - x_0)$ and $y - y_0 = -1/m(x - x_0)$ (only troubled-cells selected)	$u_i \in \mathcal{U}[-1, 1], \quad m \in \mathcal{U}[0, 20],$ $x_0, y_0 \in \mathcal{U}[-0.5, 0.5]$	0	337,976
$a (y - y_0) - m(x - x_0) $ (only troubled-cells selected)	$a \in \mathcal{U}[-100, 100], \quad m \in \mathcal{U}[-1, 1],$ $x_0, y_0 \in \mathcal{U}[-0.5, 0.5]$	0	60,182
		<b>373,076</b>	<b>398,158</b>



MLP with:

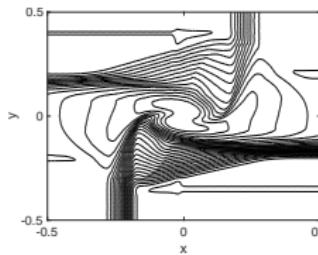
- 5 hidden layer
- 20 neurons each

# 2D Linear advection: $u_t + u_x + u_y = 0$

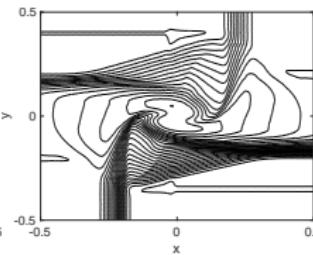
Indicator	Mesh	$p = 1$		$p = 2$		$p = 3$	
		Max. % cells flagged	Avg. % cells flagged	Max. % cells flagged	Avg. % cells flagged	Max. % cells flagged	Avg. % cells flagged
TVB-1	S-100	31.71	29.51	32.10	28.57	33.78	29.91
	S-150	20.46	18.90	24.85	21.98	27.01	24.00
	U-0.01	29.67	28.33	33.15	31.23	32.29	30.35
	U-0.005	19.43	18.46	28.20	26.54	28.39	26.79
TVB-2	S-100	28.06	26.40	28.20	25.11	30.80	26.90
	S-150	17.35	16.41	22.10	19.73	25.51	22.30
	U-0.01	25.00	23.85	28.85	26.97	27.83	25.71
	U-0.005	16.30	15.40	25.44	23.87	25.32	23.72
TVB-3	S-100	23.86	22.66	24.48	22.13	26.34	23.63
	S-150	14.77	13.95	19.08	17.44	21.54	19.65
	U-0.01	20.88	19.99	25.34	23.46	24.12	22.27
	U-0.005	13.47	12.67	23.36	21.66	23.20	21.50
MLP	S-100	1.33	0.74	1.04	0.23	0.92	0.33
	S-150	0.24	0.12	0.23	0.02	0.18	0.01
	U-0.01	0.71	0.56	0.81	0.60	1.08	0.76
	U-0.005	0.13	0.11	0.19	0.13	0.18	0.13

# 2D Euler equations

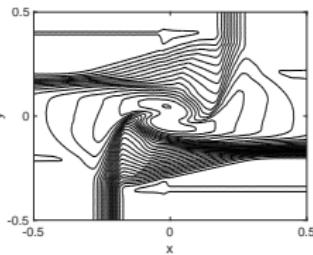
2D Riemann problem (config. 6),  $r=3$



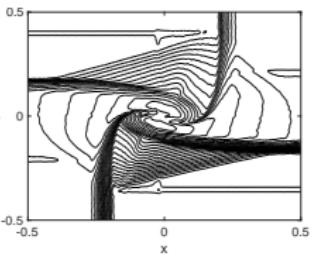
TVB-1



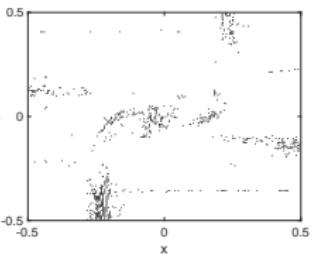
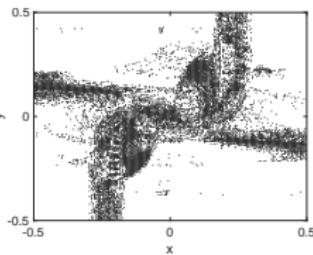
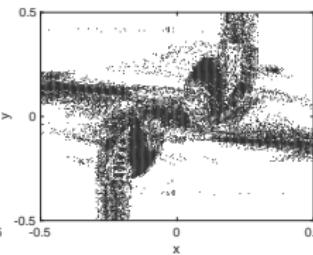
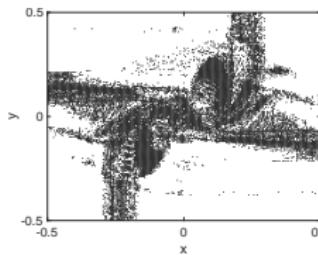
TVB-2



TVB-3

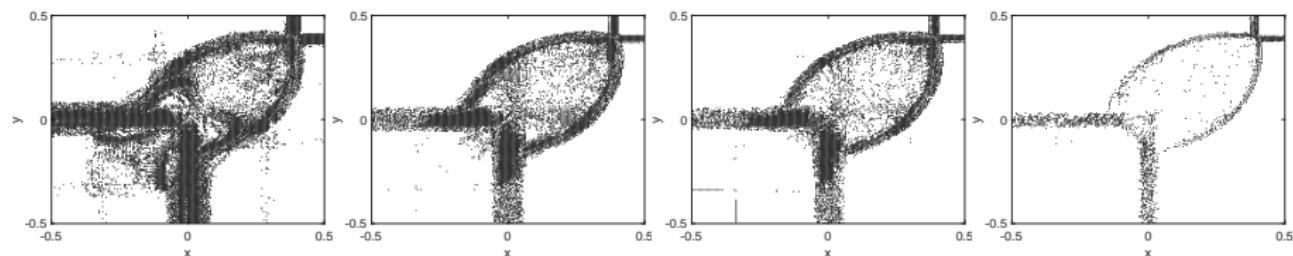
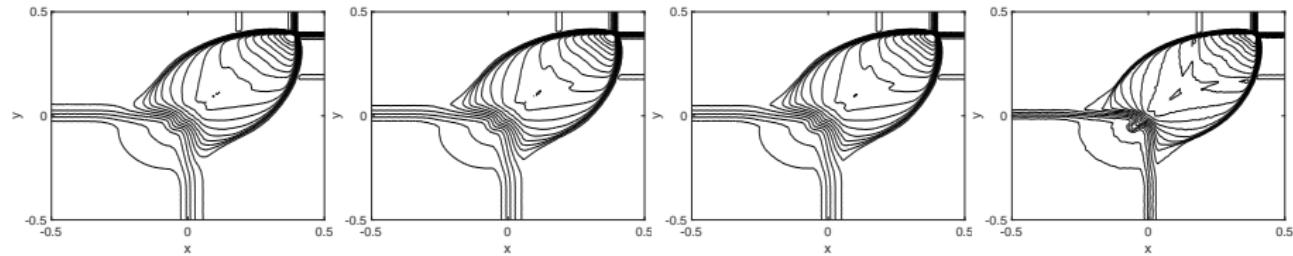


MLP



# 2D Euler equations

2D Riemann problem (config. 12),  $r=3$



*Detecting troubled-cells on two-dimensional unstructured grids using a neural network* by D. Ray and J. S. Hesthaven  
(submitted, 2018)

## Part II: ANNs predicting artificial viscosity

## Adding artificial viscosity

Now consider the problem

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}) = \nabla \cdot \mathbf{g}, \quad \mathbf{g} = \mu \mathbf{w}, \quad \mathbf{w} = \nabla \mathbf{u}$$

Viscosity depends on  $\mathbf{u}$  and locally controls oscillations.

$$\mu \sim h |\mathbf{f}'(\mathbf{u})|$$

## Adding artificial viscosity

Now consider the problem

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}) = \nabla \cdot \mathbf{g}, \quad \mathbf{g} = \mu \mathbf{w}, \quad \mathbf{w} = \nabla \mathbf{u}$$

Viscosity depends on  $\mathbf{u}$  and locally controls oscillations.

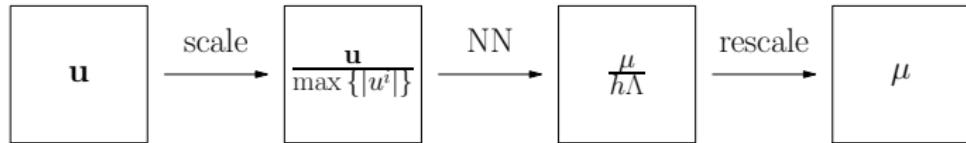
$$\mu \sim h |\mathbf{f}'(\mathbf{u})|$$

Several artificial viscosity models exist

- Derivative-based model (DB)
- Highest Modal Decay (MDH)
- Averaged Modal Decay (MDA)
- Entropy Viscosity (EV)

Dependent on problem specific parameters

# Training the 1D network

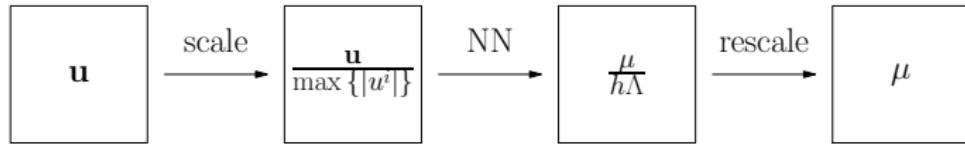


$$\Lambda = \max |\mathbf{f}'(\mathbf{u})|$$

Network architecture:

- 5 hidden layers of width 10 each
- Leaky ReLU activation
- Softplus output function, i.e.,  $f(x) = \log(1 + e^x)$
- Mean Squared Error cost function

# Training the 1D network



$$\Lambda = \max |\mathbf{f}'(\mathbf{u})|$$

Network architecture:

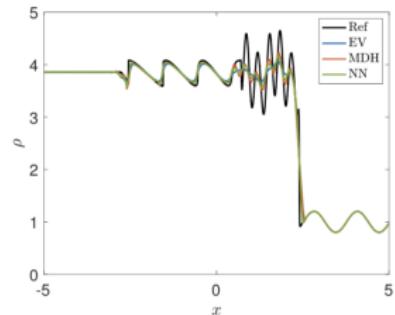
- 5 hidden layers of width 10 each
- Leaky ReLU activation
- Softplus output function, i.e.,  $f(x) = \log(1 + e^x)$
- Mean Squared Error cost function

Training/validation sets:

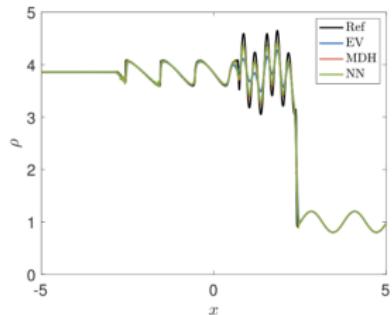
- Using numerical solution of conservation laws (only linear adv. and Burgers)
- Target viscosity: viscosity corresponding to "best" model
- Different network for each degree  $r$

# 1D Euler equations: Shock-Entropy

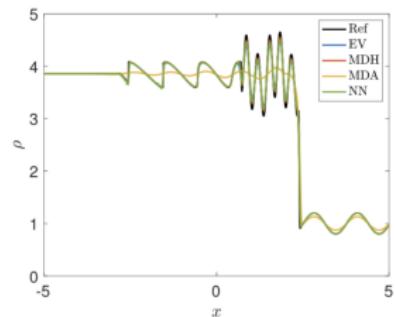
$$x \in [-5, 5], \quad T_f = 1.8, \quad h = 10/200$$



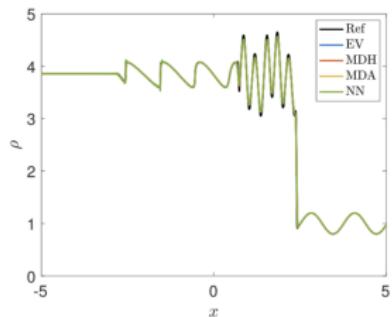
(a) Overall result,  $m = 1$ .



(b) Overall result,  $m = 2$ .



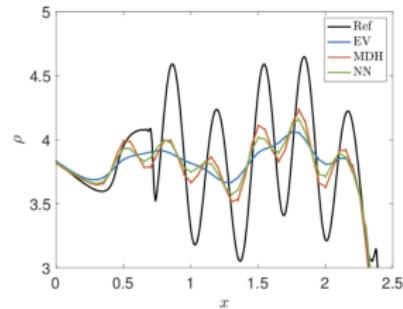
(c) Overall result,  $m = 3$ .



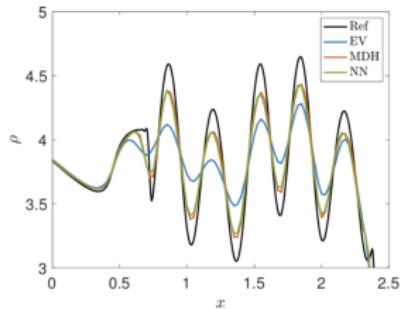
(d) Overall result,  $m = 4$ .

# 1D Euler equations: Shock-Entropy

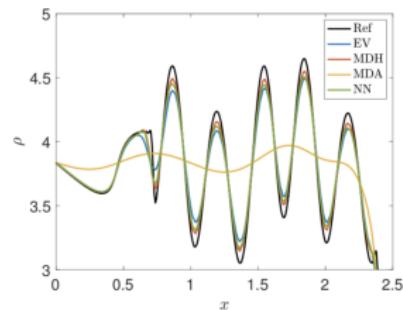
$$x \in [-5, 5], \quad T_f = 1.8, \quad h = 10/200$$



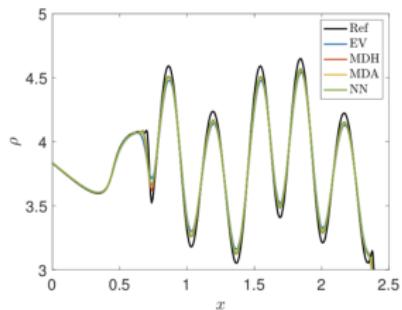
(a)  $m = 1$ .



(b)  $m = 2$ .



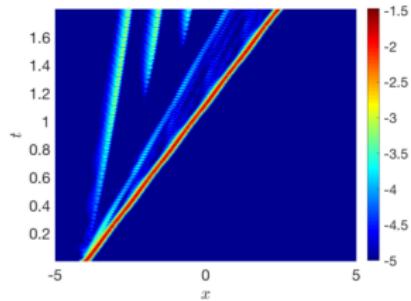
(c)  $m = 3$ .



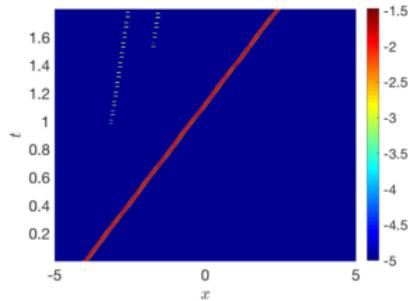
(d)  $m = 4$ .

# 1D Euler equations: Shock-Entropy

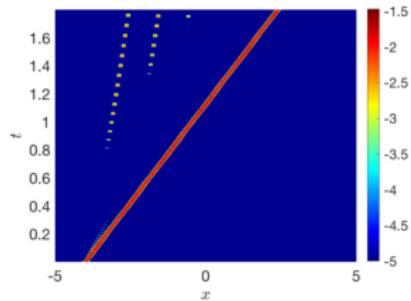
$$x \in [-5, 5], \quad T_f = 1.8, \quad h = 10/200$$



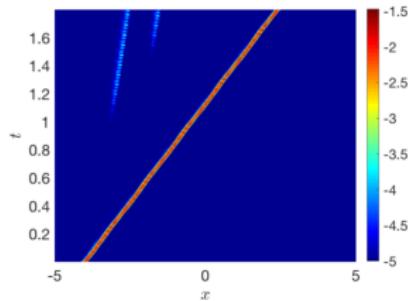
(a) EV



(b) MDH



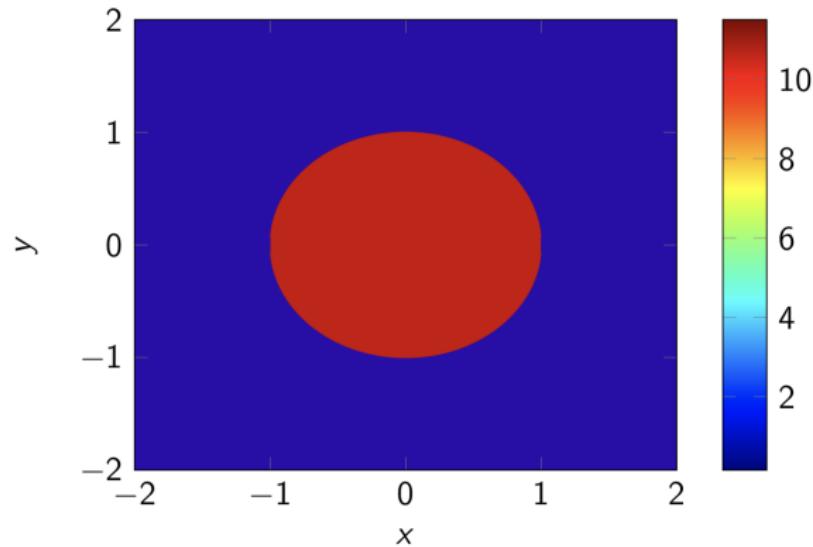
(c) MDA



(d) NN

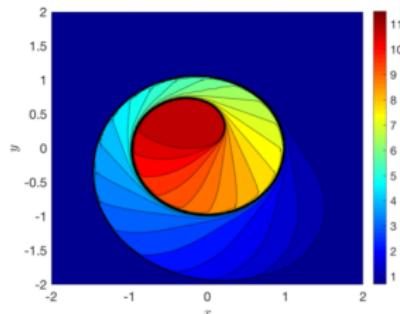
## 2D KPP equations: Rotating wave

$$\mathbf{f}(u) = (\sin u, \cos u), \quad T_f = 1, \quad h = 4\sqrt{2}/120, \quad r = 4$$

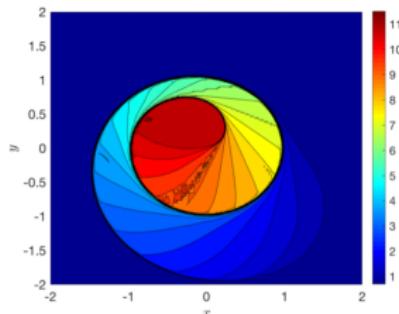


# 2D KPP equations: Rotating wave

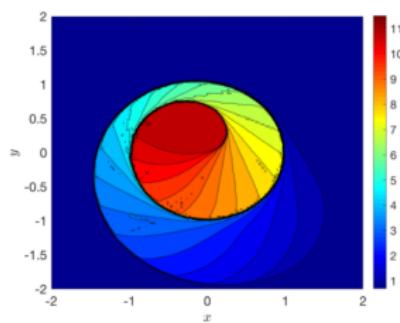
$$\mathbf{f}(u) = (\sin u, \cos u), \quad T_f = 1, \quad h = 4\sqrt{2}/120, \quad r = 4$$



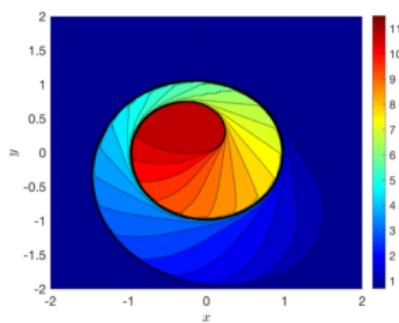
(a) EV



(b) MDH

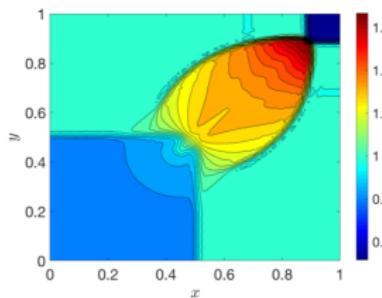


(c) MDA

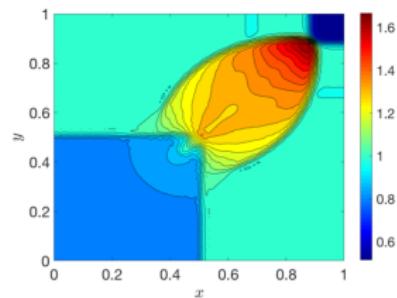


(d) NN

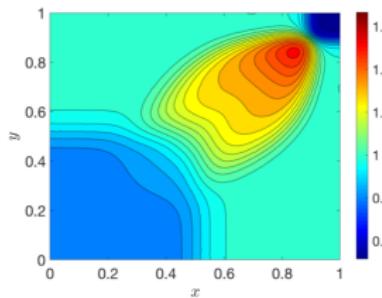
# 2D Euler equations: Riemann Problem 12, r=3



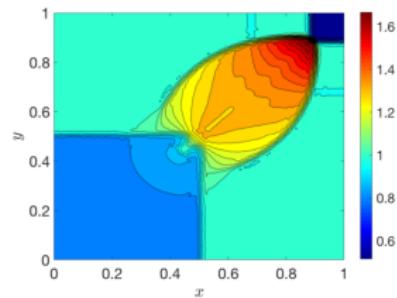
(a) EV



(b) MDH

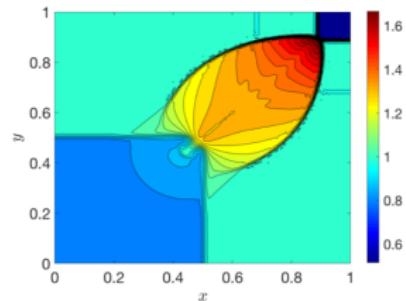


(c) MDA

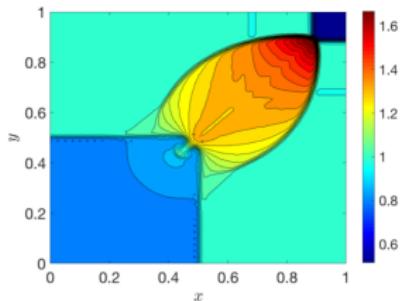


(d) NN

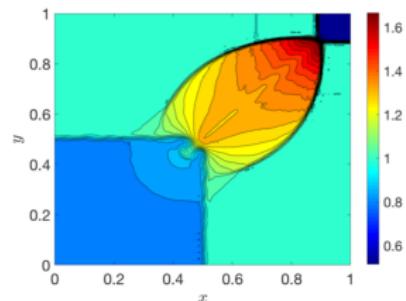
# 2D Euler equations: Riemann Problem 12, r=4



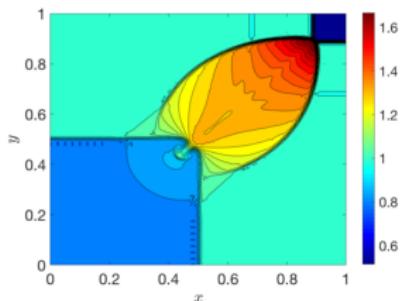
(a) EV



(b) MDH



(c) MDA



(d) NN

# Summarizing

- When trained properly, ANN's work well for classification and regression
- Only needs to be trained once offline
- Parameter free
- Improves computational cost
- Can be easily extended to 3D (in principle)

# Summarizing

- When trained properly, ANN's work well for classification and regression
- Only needs to be trained once offline
- Parameter free
- Improves computational cost
- Can be easily extended to 3D (in principle)

## References:

- *An artificial neural network as a troubled-cell indicator* by R, Hesthaven; J. Comp. Phy., vol. 367(15), 2018.
- *Detecting troubled-cells on two-dimensional unstructured grids using a neural network* by R, Hesthaven; (submitted).
- *A data-driven shock capturing approach for discontinuous Galerkin methods* by Yu, Hesthaven, Yan; (submitted)
- *Controlling oscillations in high-order Discontinuous Galerkin schemes using artificial viscosity tuned by neural networks* by Discacciati, Hesthaven, Ray; (submitted).

# Summarizing

- When trained properly, ANN's work well for classification and regression
- Only needs to be trained once offline
- Parameter free
- Improves computational cost
- Can be easily extended to 3D (in principle)

## References:

- *An artificial neural network as a troubled-cell indicator* by R, Hesthaven; J. Comp. Phy., vol. 367(15), 2018.
- *Detecting troubled-cells on two-dimensional unstructured grids using a neural network* by R, Hesthaven; (submitted).
- *A data-driven shock capturing approach for discontinuous Galerkin methods* by Yu, Hesthaven, Yan; (submitted)
- *Controlling oscillations in high-order Discontinuous Galerkin schemes using artificial viscosity tuned by neural networks* by Discacciati, Hesthaven, Ray; (submitted).

Thank you

# RKDG and limiting

---

## RKDG solver

---

```
1: Initialize U[0]
2: n = 1
3: while t .lte. Tf do
4:     U[n] = U[n-1]
5:     for r = 1 to 3 do
6:         L = FindRHS(U[n])
7:         U[n] = RK_update(U[n-1], U[n], L, r)
8:         U[n] = Limit(U[n])
9:     end for
10:    n++, t+=dt
11: end while
```

---

# RKDG and limiting

---

## RKDG solver

---

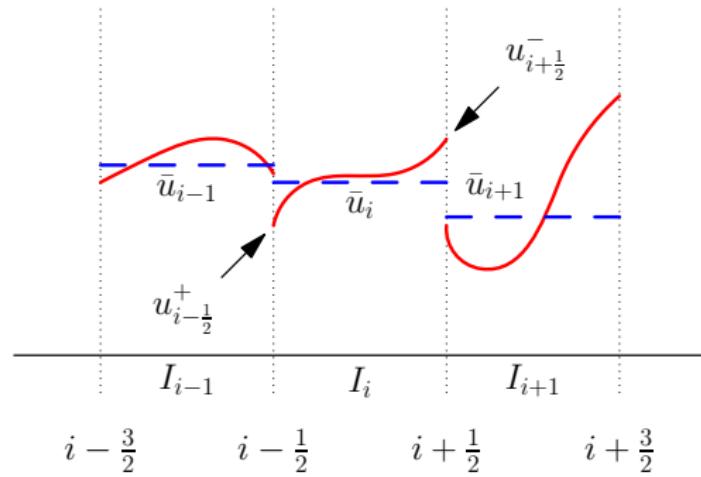
```
1: Initialize U[0]
2: n = 1
3: while t .lte. Tf do
4:     U[n] = U[n-1]
5:     for r = 1 to 3 do
6:         L = FindRHS(U[n])
7:         U[n] = RK_update(U[n-1], U[n], L, r)
8:         U[n] = Limit(U[n])
9:     end for
10:    n++, t+=dt
11: end while
```

---

Bottleneck step!!

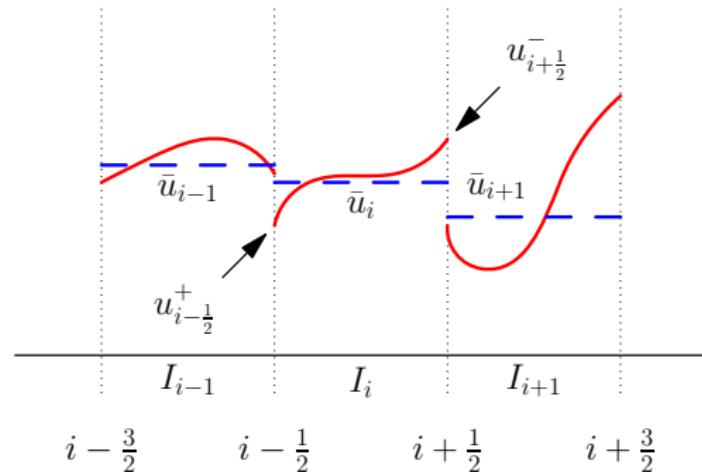
# TVB Limiter (Qiu and Shu, 2005)

**Identification:** For each cell  $I_i$ , get  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$



# TVB Limiter (Qiu and Shu, 2005)

**Identification:** For each cell  $I_i$ , get  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$

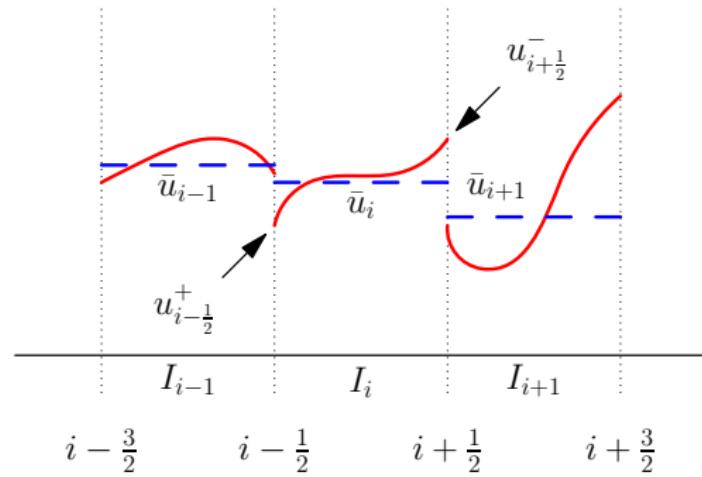


Evaluate 4 differences

$$\begin{aligned}\Delta^- u_i &= \bar{u}_i - \bar{u}_{i-1}, & \Delta^+ u_i &= \bar{u}_{i+1} - \bar{u}_i, \\ \check{u}_i &= \bar{u}_i - u_{i-\frac{1}{2}}^+, & \hat{u}_i &= u_{i+\frac{1}{2}}^- - \bar{u}_i\end{aligned}$$

# TVB Limiter (Qiu and Shu, 2005)

**Identification:** For each cell  $I_i$ , get  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$



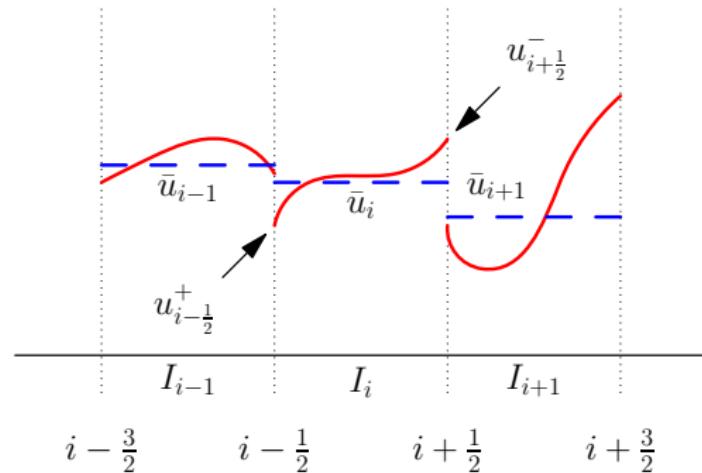
Modify interface values

$$\tilde{u}_{i-\frac{1}{2}}^+ = \bar{u}_i + \mathcal{F}(\check{u}_i, \Delta^- u_i, \Delta^+ u_i)$$

$$\tilde{u}_{i+\frac{1}{2}}^- = \bar{u}_i - \mathcal{F}(\hat{u}_i, \Delta^- u_i, \Delta^+ u_i)$$

## TVB Limiter (Qiu and Shu, 2005)

**Identification:** For each cell  $I_i$ , get  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$



Flag  $I_i$  as troubled-cell if

$$\tilde{u}_{i-\frac{1}{2}}^+ \neq u_{i-\frac{1}{2}}^+ \quad \text{or} \quad \tilde{u}_{i+\frac{1}{2}}^- \neq u_{i+\frac{1}{2}}^-$$

# Search for the elusive $M$

We consider the following limiter-based indicators  $\mathcal{F}$ :

- Minmod limiter:

$$\mathcal{F}^{\text{mm}}(a, b, c) = \begin{cases} s \cdot \min(|a|, |b|, |c|), & \text{if } s = \text{sign}(a) = \text{sign}(b) = \text{sign}(c) \\ 0, & \text{otherwise} \end{cases}$$

**Disadvantage:** Flags cell with smooth extrema

# Search for the elusive $M$

We consider the following limiter-based indicators  $\mathcal{F}$ :

- Minmod limiter:

$$\mathcal{F}^{\text{mm}}(a, b, c) = \begin{cases} s \cdot \min(|a|, |b|, |c|), & \text{if } s = \text{sign}(a) = \text{sign}(b) = \text{sign}(c) \\ 0, & \text{otherwise} \end{cases}$$

**Disadvantage:** Flags cell with smooth extrema

- TVB limiter: Depends on  $h$  and tunable parameter  $M$

$$\mathcal{F}^{\text{tvb}}(a, b, c, h, M) = \begin{cases} a, & \text{if } |a| \leq Mh^2 \\ \mathcal{F}^{\text{mm}}(a, b, c), & \text{otherwise} \end{cases}$$

$M$  is proportional to second derivative at smooth extreme

**Disadvantage:**  $M$  is problem dependent

## Limiting the solution (Qiu and Shu, 2005)

**Limited reconstruction:** In troubled cells:

- Project  $u_h$  to  $\mathbb{P}_1$

$$u_h = \bar{u}_i + \left( \frac{x - x_i}{\frac{1}{2}\Delta x_i} \right) s_i + \text{H.O.T.}$$

## Limiting the solution (Qiu and Shu, 2005)

**Limited reconstruction:** In troubled cells:

- Project  $u_h$  to  $\mathbb{P}_1$

$$\tilde{u}_h = \Pi^1 u_h = \bar{u}_i + \left( \frac{x - x_i}{\frac{1}{2} \Delta x_i} \right) s_i$$

# Limiting the solution (Qiu and Shu, 2005)

**Limited reconstruction:** In troubled cells:

- Project  $u_h$  to  $\mathbb{P}_1$

$$\tilde{u}_h = \Pi^1 u_h = \bar{u}_i + \left( \frac{x - x_i}{\frac{1}{2} \Delta x_i} \right) s_i$$

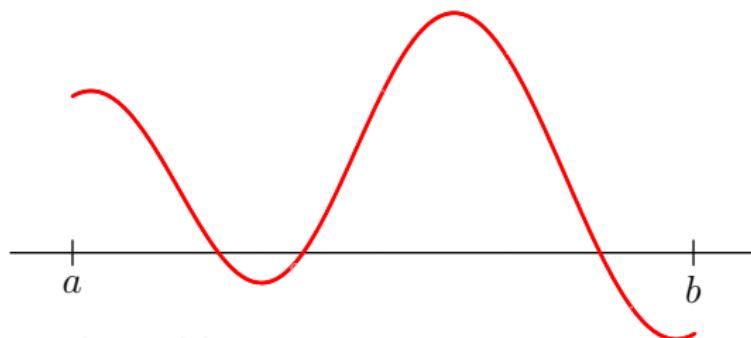
- Limit slope

$$\tilde{u}_h^{(m)} = \bar{u}_i + \left( \frac{x - x_i}{\frac{1}{2} \Delta x_i} \right) \tilde{s}_i$$

where

$$\tilde{s}_i = Q(s_i, \bar{u}_i - \bar{u}_{i-1}, \bar{u}_{i+1} - \bar{u}_i)$$

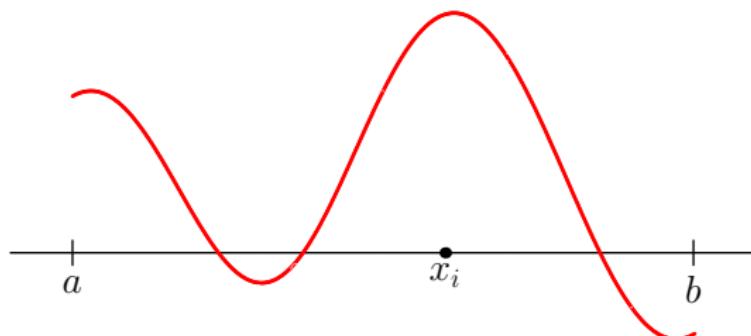
## Generating the training data



Data sampling is achieved by

- Choose a known function  $u(x)$

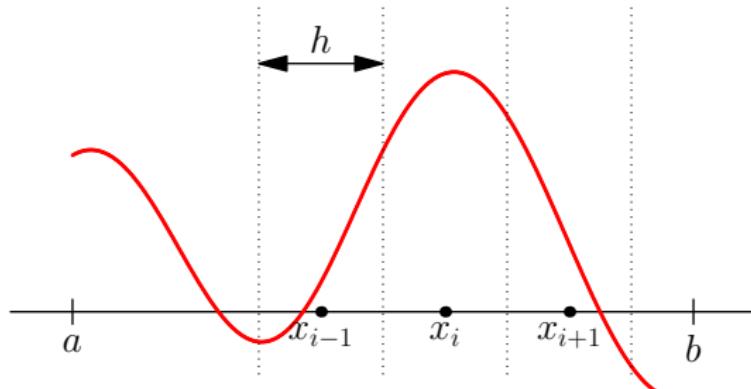
## Generating the training data



Data sampling is achieved by

- Choose a known function  $u(x)$
- Pick a point  $x_i$

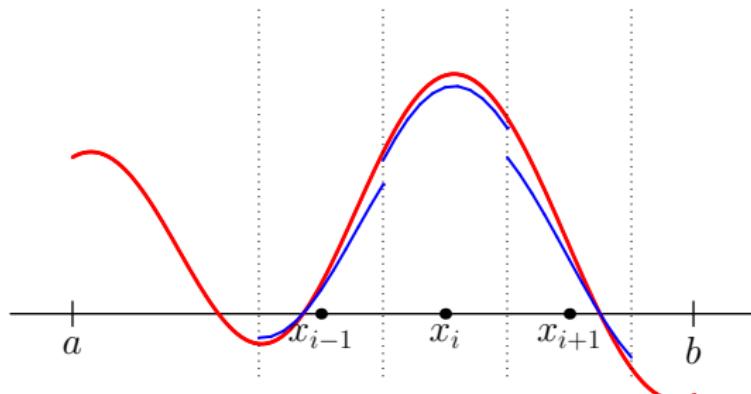
# Generating the training data



Data sampling is achieved by

- Choose a known function  $u(x)$
- Pick a point  $x_i$
- Pick a cell size  $h$  and make stencil

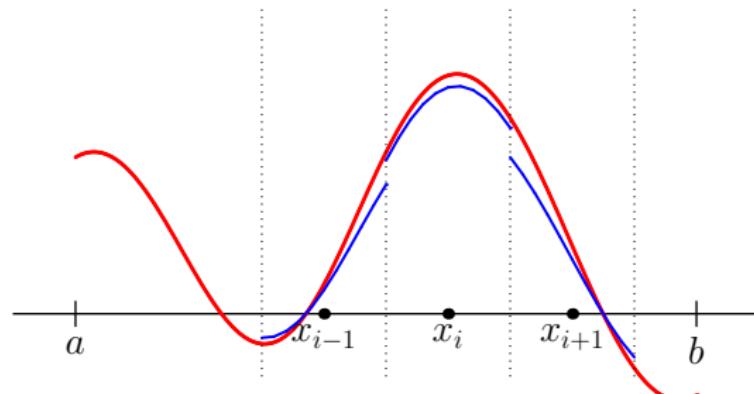
# Generating the training data



Data sampling is achieved by

- Choose a known function  $u(x)$
- Pick a point  $x_i$
- Pick a cell size  $h$  and make stencil
- Pick a degree  $r$  and approximate

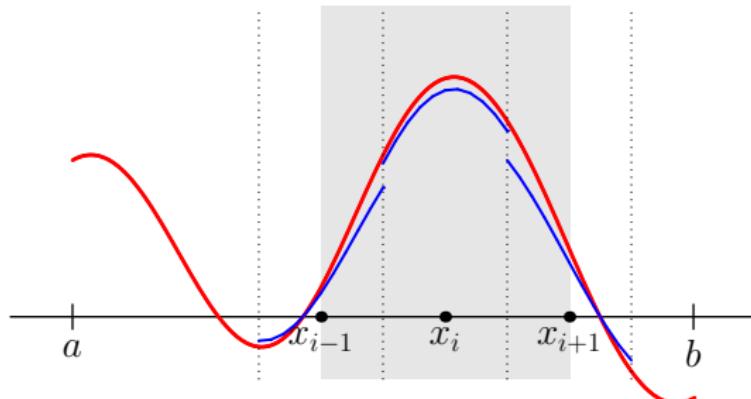
# Generating the training data



Data sampling is achieved by

- Choose a known function  $u(x)$
- Pick a point  $x_i$
- Pick a cell size  $h$  and make stencil
- Pick a degree  $r$  and approximate
- Extract needed data  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$

# Generating the training data

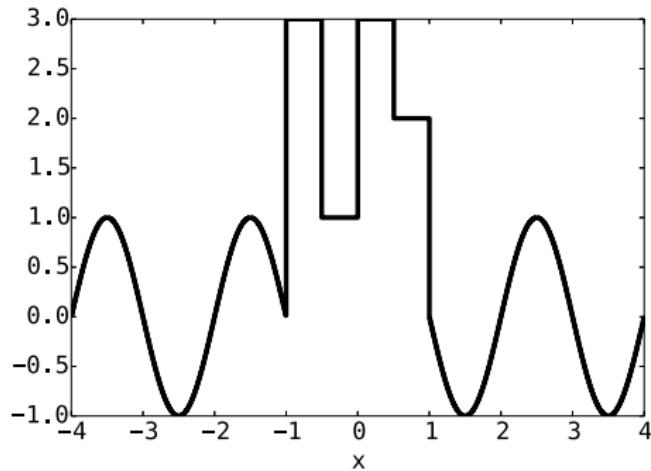


Data sampling is achieved by

- Choose a known function  $u(x)$
- Pick a point  $x_i$
- Pick a cell size  $h$  and make stencil
- Pick a degree  $r$  and approximate
- Extract needed data  $[\bar{u}_{i-1}, \bar{u}_i, \bar{u}_{i+1}, u_{i-\frac{1}{2}}^+, u_{i+\frac{1}{2}}^-]$
- Flag cell if discontinuity in  $[x_{i-\frac{1}{2}} - h/2, x_{i+\frac{1}{2}} + h/2]$

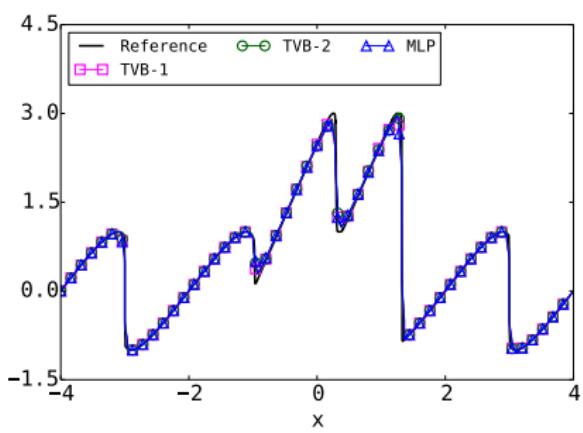
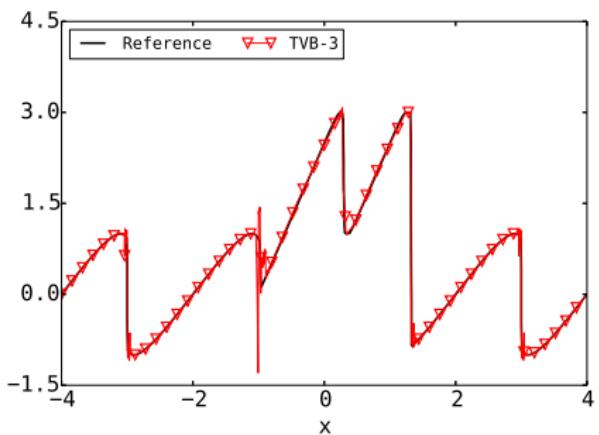
# Burgers equation: $u_t + (u^2/2)_x = 0$

$$x \in [-4, 4], \quad T_f = 0.4, \quad N = 200, \quad r = 4$$

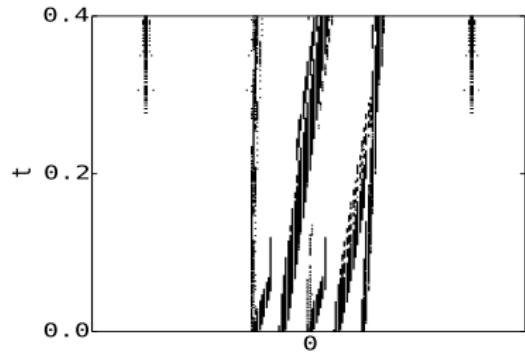


# Burgers equation: $u_t + (u^2/2)_x = 0$

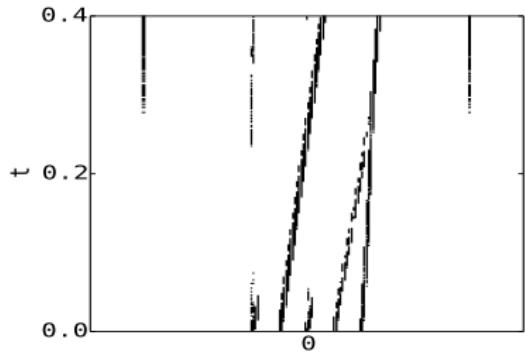
$$x \in [-4, 4], \quad T_f = 0.4, \quad N = 200, \quad r = 4$$



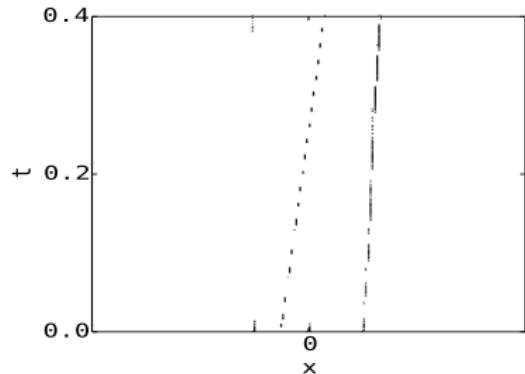
$$\text{Burgers equation: } u_t + (u^2/2)_x = 0$$



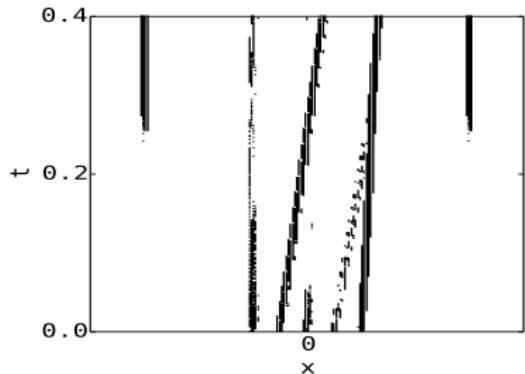
TVB-1



TVB-2



TVB-3

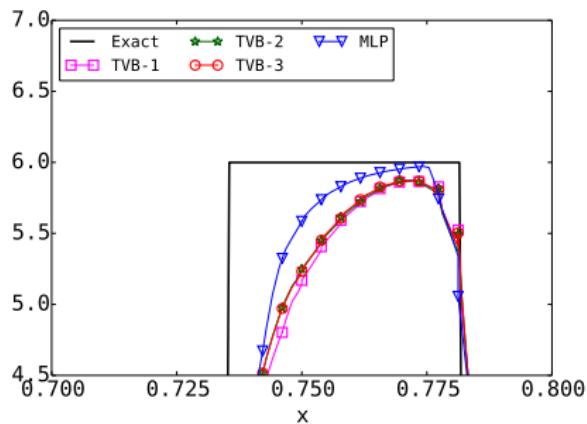
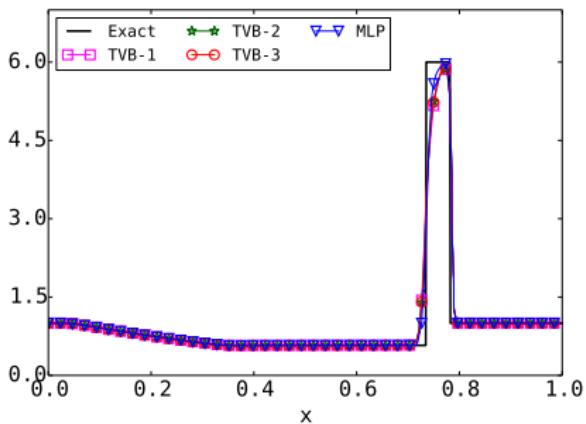


MLP

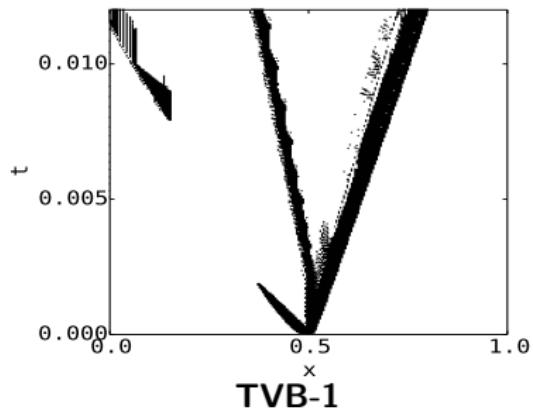
## Euler equations: Left half of blast-wave

$$(\rho, u, p) = \begin{cases} (1, 0, 1000) & \text{if } x < 0.5 \\ (1, 0, 0.01) & \text{if } x > 0.5 \end{cases}, \quad x \in [0, 1],$$

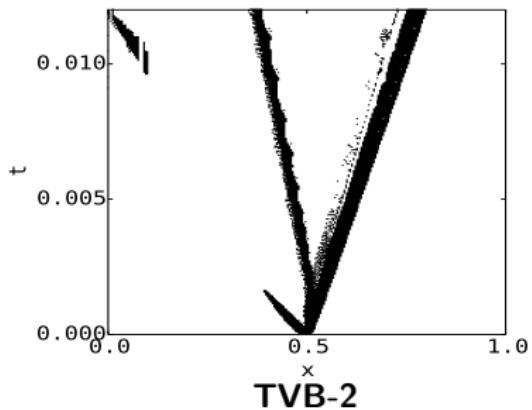
$$T_f = 0.012, \quad N = 256, \quad r = 4$$



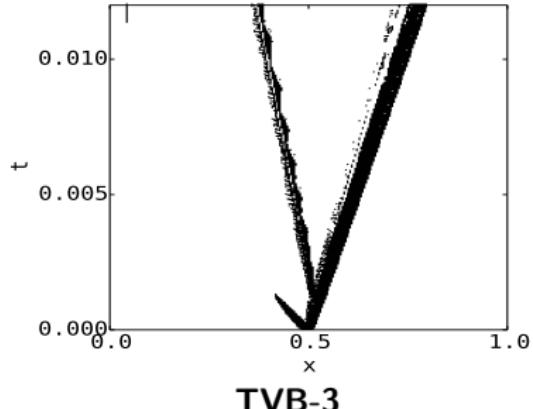
## Euler equations: Left half of blast-wave



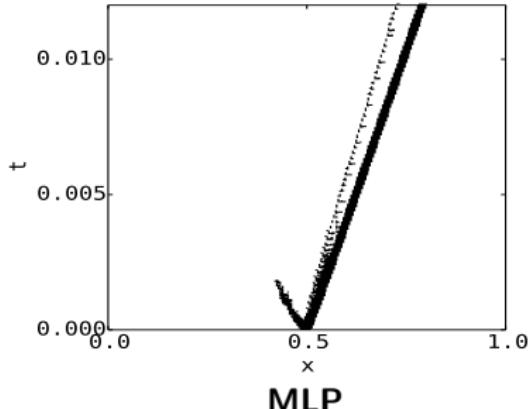
TVB-1



TVB-2



TVB-3

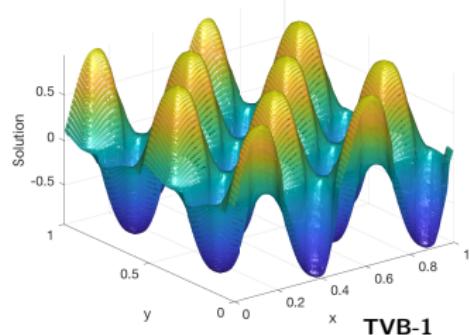


MLP

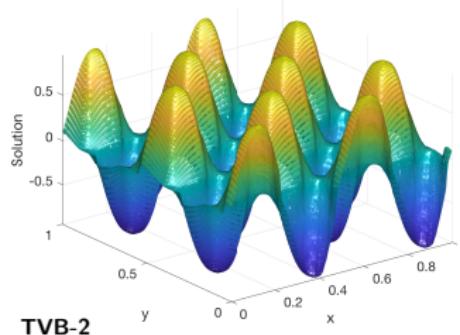
## 2D Linear advection: $u_t + u_x + u_y = 0$

$$u_0(x) = \sin(4\pi x) \sin(4\pi y), \quad r = 3$$

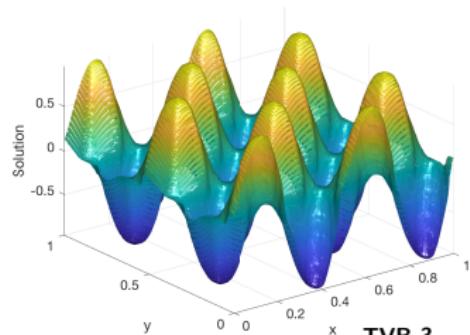
Mesh S-150:



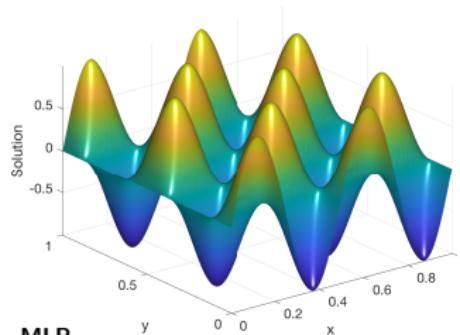
TVB-1



TVB-2



TVB-3

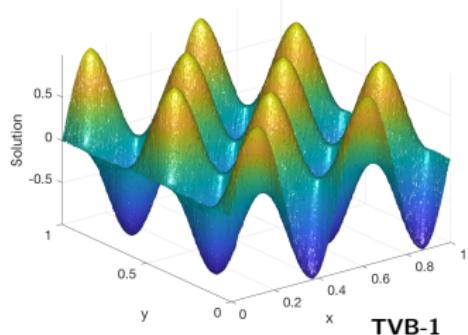


MLP

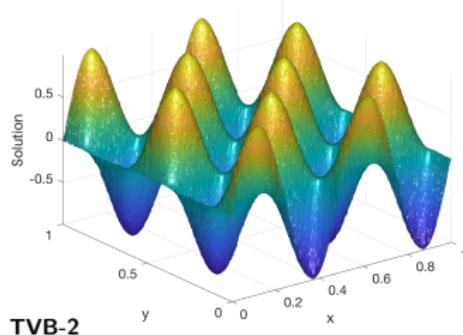
## 2D Linear advection: $u_t + u_x + u_y = 0$

$$u_0(x) = \sin(4\pi x) \sin(4\pi y), \quad r = 3$$

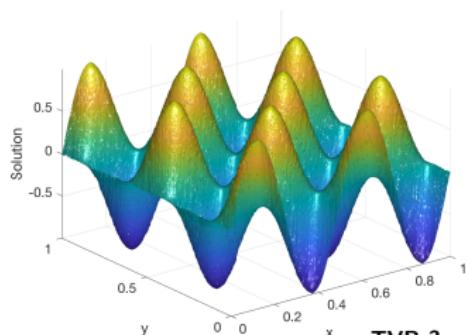
Mesh U-0.005:



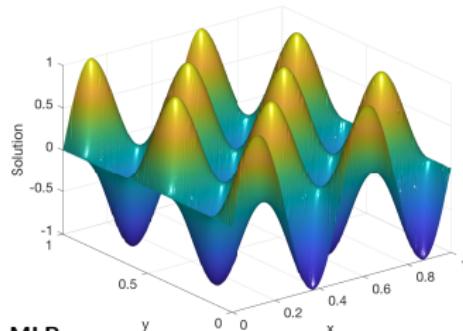
TVB-1



TVB-2



TVB-3



MLP

# 2D Euler equations

2D Riemann problem (config. 4),  $r=3$

