# *a priori* Neural Networks vs *a posteriori* MOOD loop
# A high accurate Finite Volume testing bed
## Preliminary results

Alexandre Bourriaud[1]
Raphaël Loubère[1] and Rodolphe Turpault[1]

[1]Institut de Mathématique de Bordeaux (IMB), Bordeaux, France
alexandre.bourriaud@math.u-bordeaux.fr

SHARK, May 20-24, 2019

# Summary of the talk

- Introduction
- Neural Network for dummies
- High accurate FV *a posteriori* MOOD schemes for systems of PDEs.
- MOOD weaknesses: R-DMP, implicit, massive parallelisation
- NN training for CFD in a FV MOOD context
- Numerical experiments on 1D advection equation
- Numerical experiments on 1D hydrodynamics
- Conclusions - Perspectives

# Introduction

## What do we need in a CFD high accurate FV/FE schemes?

- Identify bad/troubled cells (subcell limiter, MOOD, hierarchical limiters, etc.)
- Choose a stencil (WENO, CWENO)
- Determine the amount of dissipation (artificial viscosity, slope limiter, hierachical coefficients etc.)

## Why do we struggle with limiting/stabilizing a numerical scheme?

- (highly) Non-linear solution and non-linear methodology
- Free-parameters in the methodologies

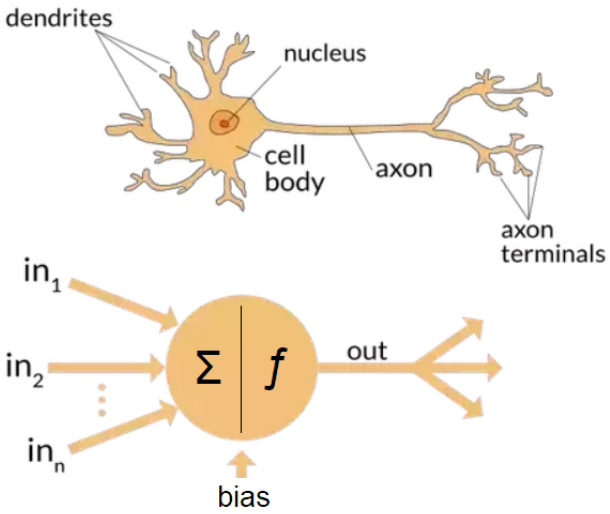
## Could we use Neural Networks for such CFD computations?

Can we improve a Finite Volume high-accurate scheme with a NN? More efficient, more accurate, faster, simpler, etc. ?
We will show an attempt to do so in a 1D FV MOOD scheme for advection and Euler equations

[1] A. Bourriaud, R. Loubère, R. Turpault, *a priori* Neural Networks vs *a posteriori* MOOD loop. A high accurate Finite Volume scheme testing bed. To be submitted in 09/2019
[2] D. Ray, J. Hesthaven, An artificial neural network as a troubled-cell indicator, JCP 367, 2018, Pages 166-191
[3] N. Discacciati, J.S. Hesthaven, D. Ray, Controlling oscillations in high-order Discontinuous Galerkin schemes using artificial viscosity tuned by neural networks, preprint 2019

# Neural Network for dummies

## Neuron/Perceptron




Inspired by neural structure in brain

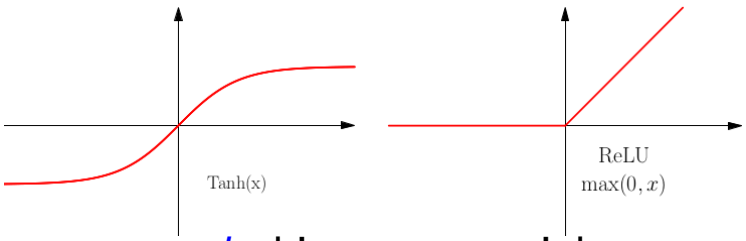
- In-put signals are transformed into out-put signal(s)
- An artificial perceptron/neuron $\delta$ mimics the behavior of a single neuron
- Perceptron: $\Sigma$=linear combinaison, $f$=non-linear mix

## Perceptron

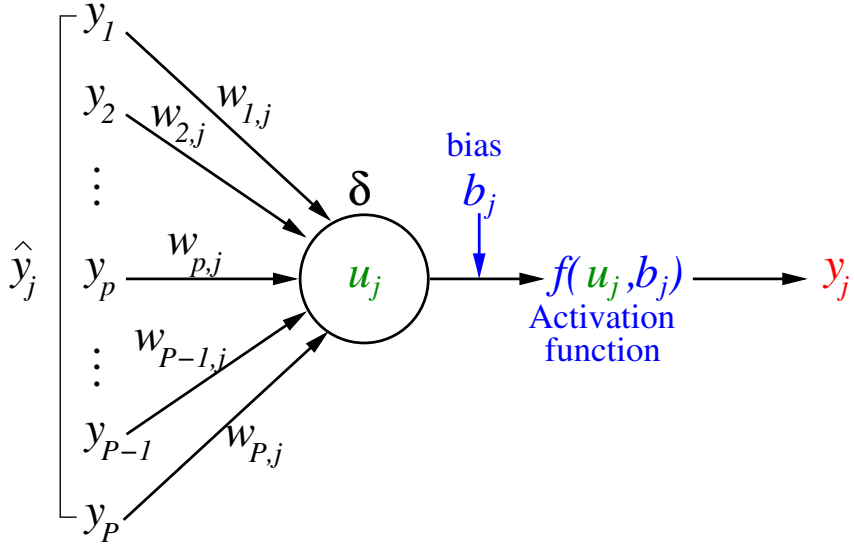
Input signals: $\widehat{y_j} = (y_1, \ldots, y_P)$ and output $y_j$

$$u_j = \sum_{p=1}^{P} w_{p,j}\, y_p, \quad y_j = f(u_j - b_j)$$

$f$: non-linear activation function,

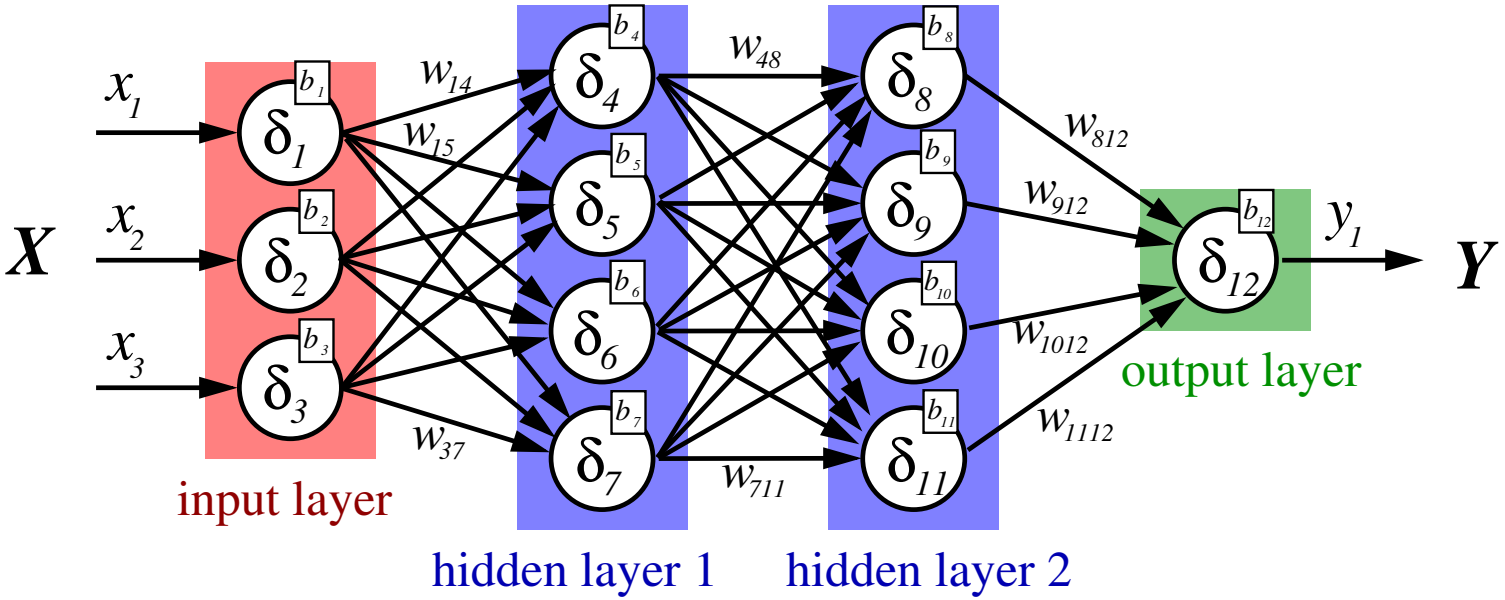



Unknowns $\Longrightarrow$ $b_j$ bias, $w_{p,j}$: weights



# Neural Network for dummies

## Multilayer perceptron network




Multi-connected set of perceptrons = Multi-Layer Perceptron (MLP) network

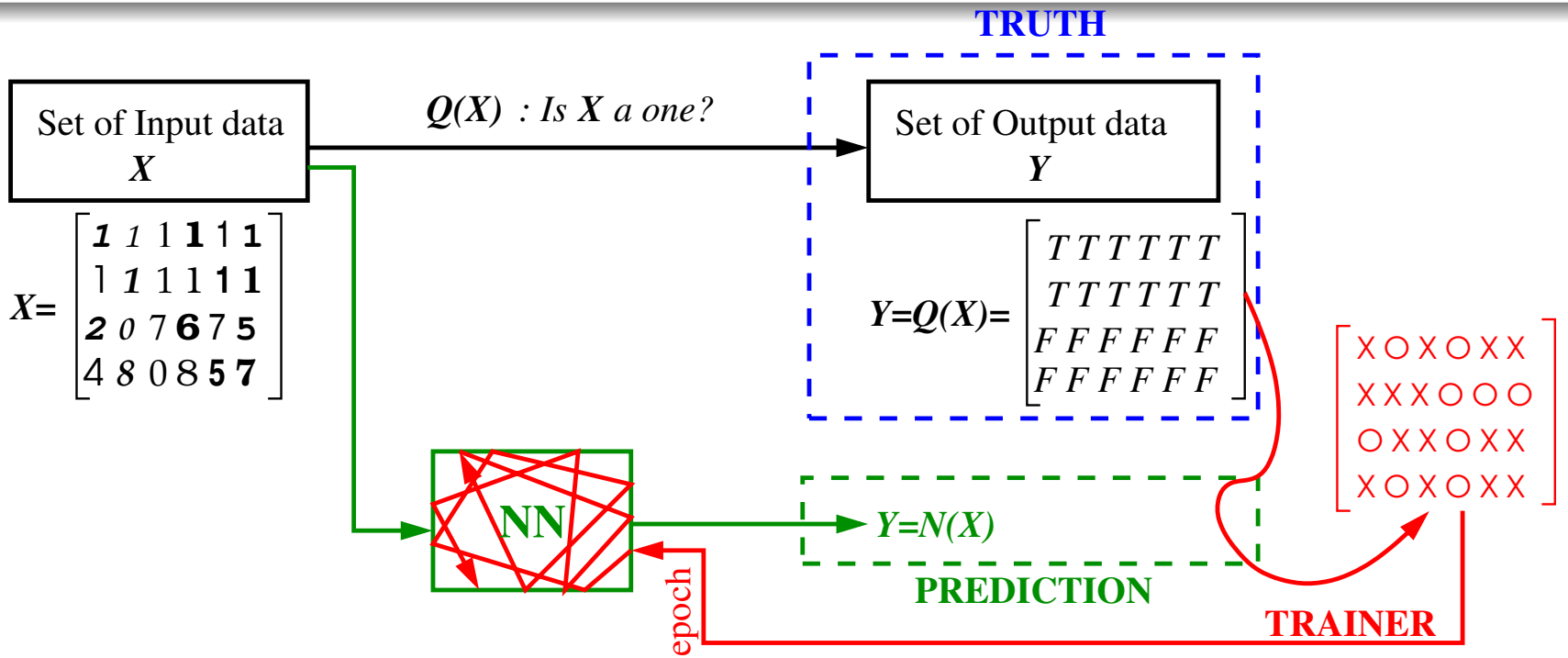
- Network architecture: $\ell = 1, \ldots, L$ layers of $P_\ell \geq 1$ perceptron(s) each
- Input/source layer is 1st, ouptut layer is last, in-btwn are hidden layers
- $\mathcal{N}$:=MLP=feed-forward network, weights and bias determined by a supervised learning

$$\text{Mapping}: \boldsymbol{X} = (x_1, x_2, \ldots, x_I) \xrightarrow{\mathcal{N}} \boldsymbol{Y} = (y_1, y_2, \ldots, y_O) = \mathcal{N}(\boldsymbol{X})$$
$$\text{Input} \rightarrow \text{Output}$$

# Neural Network for dummies

## NN must be trained (to fix its weights/bias)

A NN does exercice with a trainer which corrects/validates its predictions $\Longrightarrow$ "supervised training"
A false prediction is backwardly propagated into the NN for it to "learn" from its mistake.
Need for a training data-set, a verification data-set, a test data-set and a cost function.



# Neural Network for dummies

## Multilayer perceptron network: what for?

### A NN is employed nowadays in many applications

- voice, speech, image classification, google translate, automatic cars, go/chess players
- identifying faces, self-driving autonomous cars, langage driven image, etc.
- Bank, marketing, retail/sale, etc.

- Is good at detecting patterns, features. Is good at answering Yes or No.
- Is not good to predict real values.
- Can make mistakes. Can not predict things that it has never learned the existence of.

### Can we use this in a CFD community?

Already done in [2], [3], [4] and elsewhere for trouble cell indicator, artificial viscosity tuner, residual distribution scheme pickers, etc.
Today we are interested in using a NN in our High Accurate MOOD FV scheme

[2] D. Ray, J. Hesthaven, An artificial neural network as a troubled-cell indicator, JCP 367, 2018, Pages 166-191
[3] N. Discacciati, J.S. Hesthaven, D. Ray, Controlling oscillations in HO DG schemes using artificial viscosity tuned by neural networks, preprint 2019
[4] M.H. Veiga, R. Abgrall, Towards a general stabilisation method for conservation laws using a multilayer Perceptron neural network: 1D scalar and system of equations, preprint 2019, ¡hal-01856358¿

# High accurate FV MOOD scheme

## Basics

### Governing equations: hyperbolic system of PDEs (advection or Euler)

$$\frac{\partial \boldsymbol{W}}{\partial t} + \frac{\partial \mathrm{F}(\boldsymbol{W})}{\partial x} = 0, \qquad \boldsymbol{W} : \text{ conservative variables, } \mathrm{F}(\boldsymbol{W}) : \text{ flux vector.}$$

### Finite Volume discretization

Domain: $\Omega = [x_{\min}, x_{\max}]$, uniform mesh $\Omega_i = [x_{i-1/2}, x_{i+1/2}]$ for $i = 1, \ldots, M$. cell center: $x_i = \frac{1}{2}(x_{i-1/2} + x_{i+1/2})$ and $\Delta x = x_{i+1/2} - x_{i-1/2}$.
Time: $[0, T]$, cells $[t^n, t^{n+1}]$ of size $\Delta t \equiv \Delta t^{n+1/2} = t^{n+1} - t^n$.
Space/time dependent solution $\boldsymbol{W}(x,t)$ represent. over cell $\Omega_i$ at time $t^n$ by its mean value

$$\boldsymbol{W}_i^n = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} \boldsymbol{W}(x, t^n)\, dx.$$

FV scheme with HO reconstruction $\widetilde{\boldsymbol{W}}_i^n(x_{i+1/2})$ is given by

$$\begin{aligned} \boldsymbol{W}_i^{n+1} &= \boldsymbol{W}_i^n - \Delta t \left[\mathrm{F}(\boldsymbol{W}^n(x_{i+1/2})) - \mathrm{F}(\boldsymbol{W}^n(x_{i-1/2}))\right] \\ \mathrm{F}(\boldsymbol{W}^n(x_{i+1/2})) \equiv \mathcal{F}_{i+1/2}^n &= \mathcal{F}\left(\widetilde{\boldsymbol{W}}_i^n(x_{i+1/2}), \widetilde{\boldsymbol{W}}_{i+1}^n(x_{i+1/2})\right) \longleftarrow \text{Numerical flux} \end{aligned}$$

# High accurate FV MOOD scheme

## Polynomial reconstruction to reach $(d+1)$th order of accuracy in space

### $\mathbb{P}_d$ polynomial reconstruction using centred stencil in cell $I_i$

Search for $\widetilde{\boldsymbol{W}}_i^d \in \mathbb{P}_d(I_i)$ with same mean value in $I_i$

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \widetilde{\boldsymbol{W}}_i^d(x)\,dx = \boldsymbol{W}_i,$$

and best fit (LS sense) on a centred stencil of sufficient neighbor cells, $\mathcal{S}_i^d$,

$$J(\widetilde{\boldsymbol{W}}_i^d) = \sum_{k\in\mathcal{S}_i^d} \left| \left( \int_{x_{k-1/2}}^{x_{k+1/2}} \widetilde{\boldsymbol{W}}_i^d(x)\,dx \right) - \boldsymbol{W}_k, \right|^2 .$$

Involves solving a linear system only. Here we choose: $|\mathcal{S}_i^d| > 1.5d$.

### Reconstruction operator

$$\boldsymbol{R}_i^d \;:\; (\boldsymbol{W}_i^n, d, \mathcal{S}_i^d, (W_k^n)_{k\in\mathcal{S}_i^d}) \longrightarrow \widetilde{\boldsymbol{W}}_i^d \in \mathbb{P}_d(I_i).$$

Reconstruction of $d=0 \Rightarrow$ FV data, ie $\widetilde{\boldsymbol{W}}_i^{d=0}(x) = \boldsymbol{W}_i^n$ for all $x$ in $I_i$.

# High accurate FV MOOD scheme

## Time discretization

### SSPRK 3rd order scheme

$$\boldsymbol{W}_h^{n+1} = \boldsymbol{W}_h^n + \Delta t\, \mathcal{L}(\boldsymbol{W}_h^n),$$

where $\mathcal{L}$ is the spatial discrete operator associated to the current FV scheme.
ARK3 time discretization is an iterative scheme such that

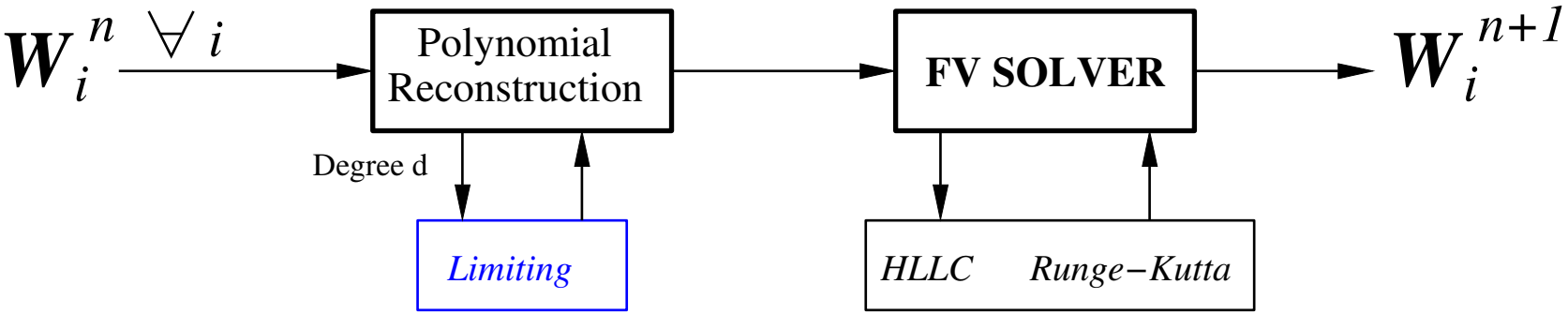
$$\begin{aligned}
\boldsymbol{W}_h^{(1)} &= \boldsymbol{W}_h^n + \Delta t\, \mathcal{L}(\boldsymbol{W}_h^n) && (1)\\
\boldsymbol{W}_h^{(2)} &= \boldsymbol{W}_h^{(1)} + \Delta t\, \mathcal{L}(\boldsymbol{W}_h^{(1)}) \quad \rightarrow \quad \boldsymbol{W}_h^{*} = \frac{3}{4}\boldsymbol{W}_h^n + \frac{1}{4}\boldsymbol{W}_h^{(2)} && (2)\\
\boldsymbol{W}_h^{(3)} &= \boldsymbol{W}_h^{(2)} + \Delta t\, \mathcal{L}(\boldsymbol{W}_h^{*}) \quad \rightarrow \quad \boldsymbol{W}_h^{n+1} = \frac{1}{3}\boldsymbol{W}_h^n + \frac{2}{3}\boldsymbol{W}_h^{(3)}. && (3)
\end{aligned}$$

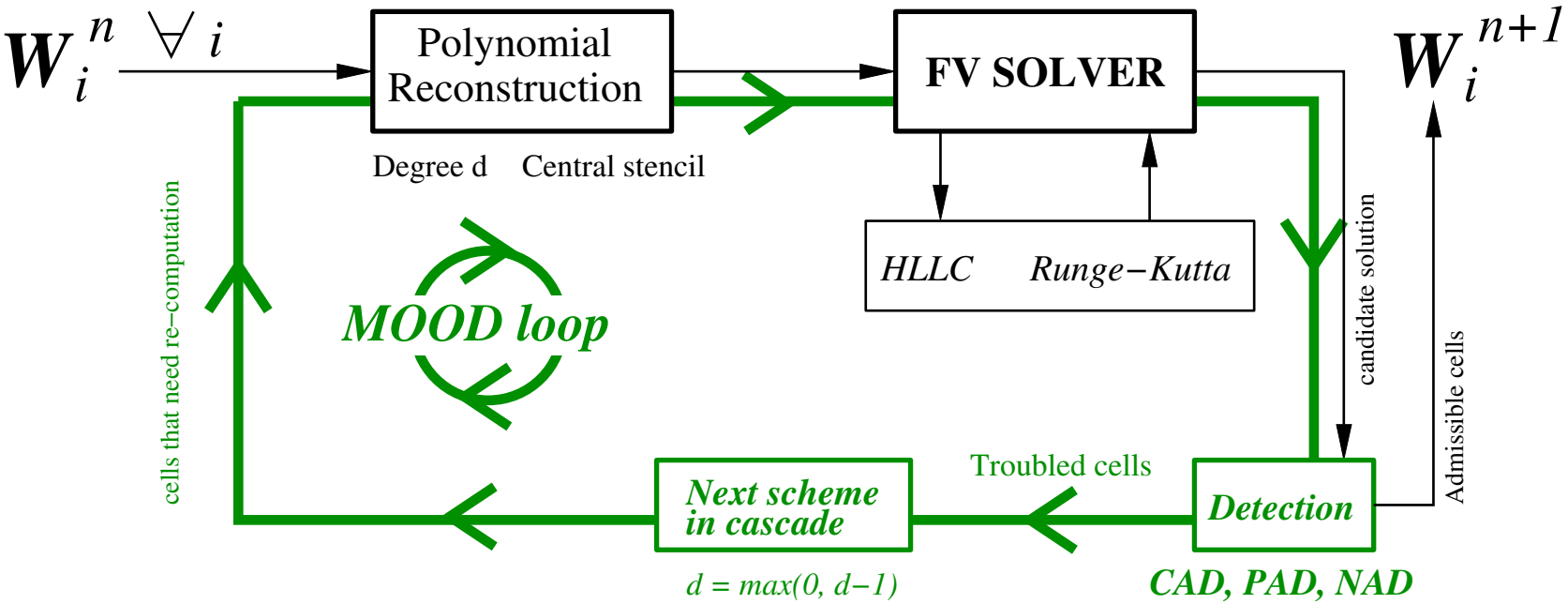
### Reaching $> 3$ order of accuracy in time

SSPRK3 introduces a third-order error $\Rightarrow$ formally third-order accurate
However, we may set $\Delta t \leq \Delta x^{r/3}$ where $r$ is the spatial order of accuracy such that the time error matches with spatial error leading to a formal $r$th order accurate space/time scheme.

# High accurate FV MOOD scheme

## *a posteriori* MOOD stabilization



# High accurate FV MOOD scheme

## *a posteriori* MOOD stabilization

### MOOD tools

- Detection criteria answer the question "Is the candidate solution valid in cell $I_i$?"
- Parachute scheme must ensure solution acceptability (ex: 1st order FV scheme)
- Cascade of schemes to be tested (ex: $d = 3 \rightarrow 2 \rightarrow 1 \rightarrow 0$) and MOOD loop

### What can we improve?

Parachute, Cascade and Loop and the variables to check/limit are somewhat fixed by the developer and/or user.
But the detection criteria are not entirely mathematically justified. Here we can act!

# High accurate FV MOOD scheme

## *a posteriori* detection criteria

**Computational - CAD** the numerical solution should pass the Computer Admissible Detection criteria. No NaN (Not-a-Number), Inf (Infinite), etc.

Ensure that the code has not already crashed

**Physical - PAD** the num. sol. must verify the physical properties given of the PDEs model. The PAD criteria are adapted to the physics.
Advection equation $\Rightarrow$ solution in-between the initial bounds
Euler $\Rightarrow \rho_i^{n+1,*} > 0$ and internal energy $\varepsilon_i^{n+1,*} > 0$

Ensure that the code will not crashed in the next time step

**Numerical - NAD** the numerical solution should be Numerically admissible $\longleftarrow$ Fuzzy concept
Essentially non-oscillatory (ENO): usually rely on a Relaxed Discrete Maximum Principle (RDMP) on the conservative variables for variable $A$

$$-\delta + \min_{k \in \mathcal{S}_i^d}(A_k^n) \leq A_i^{n+1,*} \leq \max_{k \in \mathcal{S}_i^d}(A_k^n) + \delta,$$

the relaxing parameter $\delta$ is fixed to $\delta = \min\left(10^{-4}, 10^{-3}|M_i^d - m_i^d|\right)$.
Alternatively a $u2$ smooth extrema indicator can be used in replacement of $\delta$.

Ensure that the code is essentially non-oscillatory

# High accurate FV MOOD scheme

## MOOD weaknesses

**1. NAD criteria.** the RDMP criteria is not firmly based on mathematical concepts and some parameters still need to be tuned. Due to no definition of what is an oscillation and how to charaterise it (even *a posteriori* ). Smooth new extrema should not be limited, non-smooth ones should.
**2. Massive parallelisation.** tendency to slow down massively parallel simulation. The cells do not endure the same amount of work depending on their physical local situation (may be re-computed several times (up to the parachute scheme)), while some cells are updated once by the highest accurate scheme.
**2. Implicitness.** Explicit nature of the numerical method as it stands. The dissipation is added localy in space in order to act there (and not everywhere as for an implicit scheme).

Most of those drawbacks are due to the *a posteriori* MOOD re-update

If a NN could *a priori* predict the polynomial degree in each cell, then the need for *a posteriori* detection could be mitigated (ideally removed).
However the *a posteriori* loop must remain to ensure the CAD, PAD criteria.

Let us try to use a NN to do this but in a simplified context:
$d_i \leq 3$, 1D, advection/Euler, for a uniform grid.

# NN training for a 1D FV MOOD context

Goal: make an educated guess of the $d_i = 0, 1, 2$ or 3 in cell $I_i$ *a priori* at $t^n$

Input: mean values in stencil $\mathcal{S}_i^{d=3}$: $\boldsymbol{X}_i = (\boldsymbol{W}_{i-3}^n, \boldsymbol{W}_{i-2}^n, \boldsymbol{W}_{i-1}^n, \boldsymbol{W}_i^n, \boldsymbol{W}_{i+1}^n, \boldsymbol{W}_{i+2}^n, \boldsymbol{W}_{i+3}^n)$
Output: probability that MOOD scheme uses $d_i = 0, 1, 2$ or 3 for the current time-step

$$\boldsymbol{Y}_i = P = (P_0, P_1, P_2, P_3), \quad \text{with} \quad 0 \leq P_d \leq 1, \ \sum_{d=0}^{3} P_d = 1 \Longrightarrow d_i = \left\{ d \,\text{s.t}\, P_d = \max_{k=0,1,2,3}(P_k) \right\}$$

## Architecture of NN

Layers $\ell$: 1 Input, $H \geq 1$ Hidden and 1 Output layers: $3 \leq \ell \leq L = H + 2$ layers in total
Perceptrons: 7 in Input, 4 in Ouput, 10 in Hidden layers
Activation fcts: $f(x) =$ sigmoid and $f(x) = x$ (output $\ell$)
Learning: Levenberg-Marquadt back-propagation algorithm [5,6,7] = network training that updates weight and bias values according to L-M optimization.

## Effective construction by Matlab, simuation in F90

`trainlm` function with 200 good predictions to validate the NN over a max of 5000.
The NN is stored and passed into the F90 MOOD scheme and used *a priori* to anticipate $d_i$.


[5] Marquardt, D., "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," SIAM Journal on Applied Mathematics, Vol. 11, No. 2, June 1963, pp. 431–441.
[6] Hagan, M.T., and M. Menhaj, "Training feed-forward networks with the Marquardt algorithm," IEEE Transactions on Neural Networks, Vol. 5, No. 6, 1999, pp. 989–993, 1994.
[7]Hagan, M.T., H.B. Demuth, and M.H. Beale, Neural Network Design, Boston, MA: PWS Publishing, 1996.

# NN training for a 1D FV MOOD context

## Advection equation

### Initializing the NN

Choose the architecture ($L$ layers, $P_\ell$ perceptrons/layer). All weights and biases are set to 1.

### Training

Profile function at $t^n$: evolution of a mix of sin and Heaviside functions.
$\boldsymbol{X}_i^n$ are the mean normalised values in the vicinity of cell $I_i$
1) Simulation $[t^n, t^{n+1}]$ with *a posteriori* MOOD scheme: for $\boldsymbol{X}_i$, get $\boldsymbol{Y}_i^n = \text{MOOD}(\boldsymbol{X}_i^n) = d_i^n$
2) (in parallel) Simulation with *a priori* NN scheme: get $\widehat{\boldsymbol{Y}}_i^n = \text{NN}(\boldsymbol{X}_i^n) = \widehat{d}_i^n$
3) The prediction is backwardly propagated into the NN (accordingly updates weights/biases)

### Training data-set

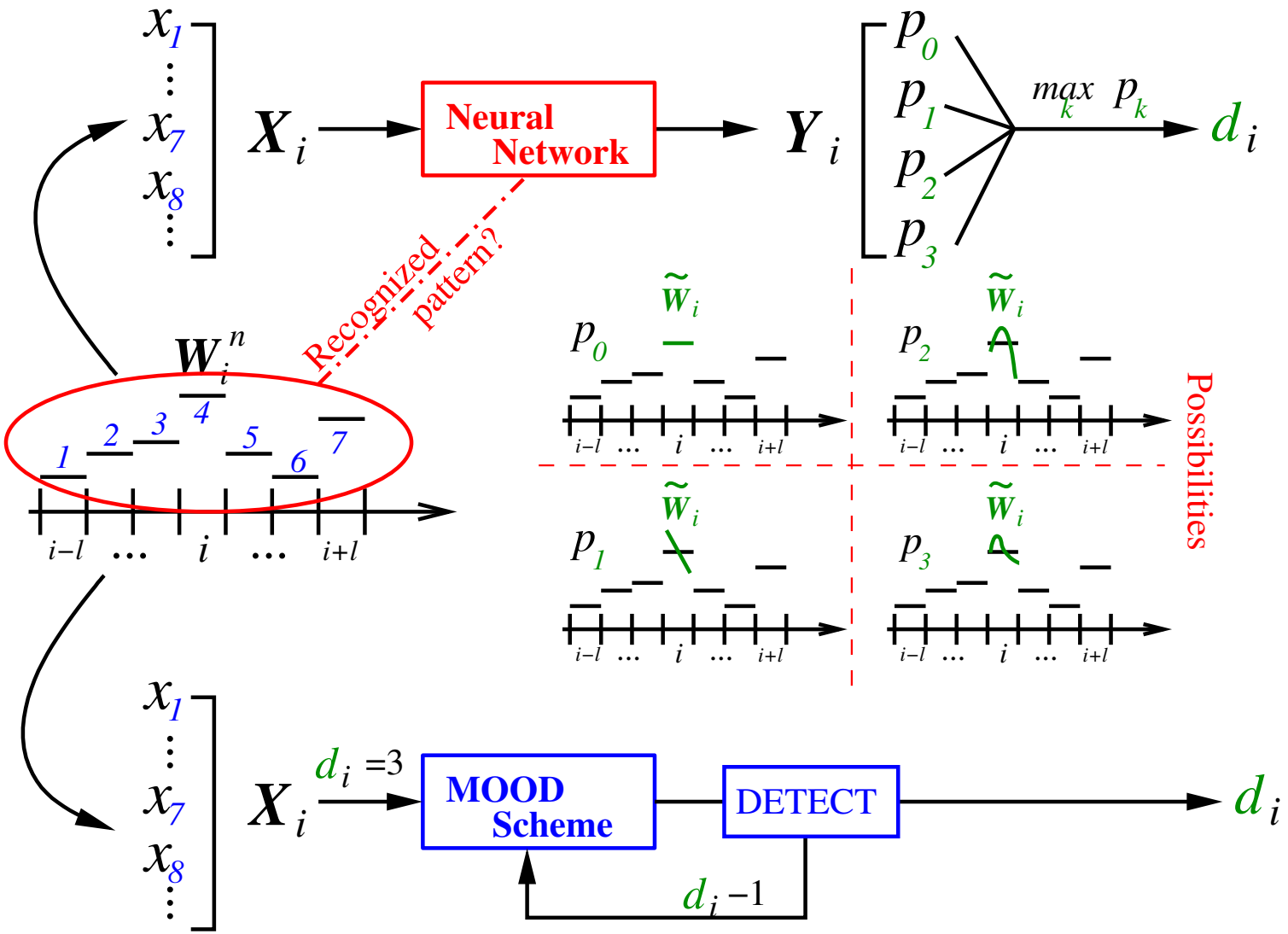
Set of pairs $(\boldsymbol{X}_i^n, \boldsymbol{Y}_i^n)$ at time $t^n$. For $N_c$ cell and $N_t$ time iterations $\Longrightarrow$ $N_c \times N_t$ data.

### Simulations made starting from unknowns data $G(x)$ at $t = 0$

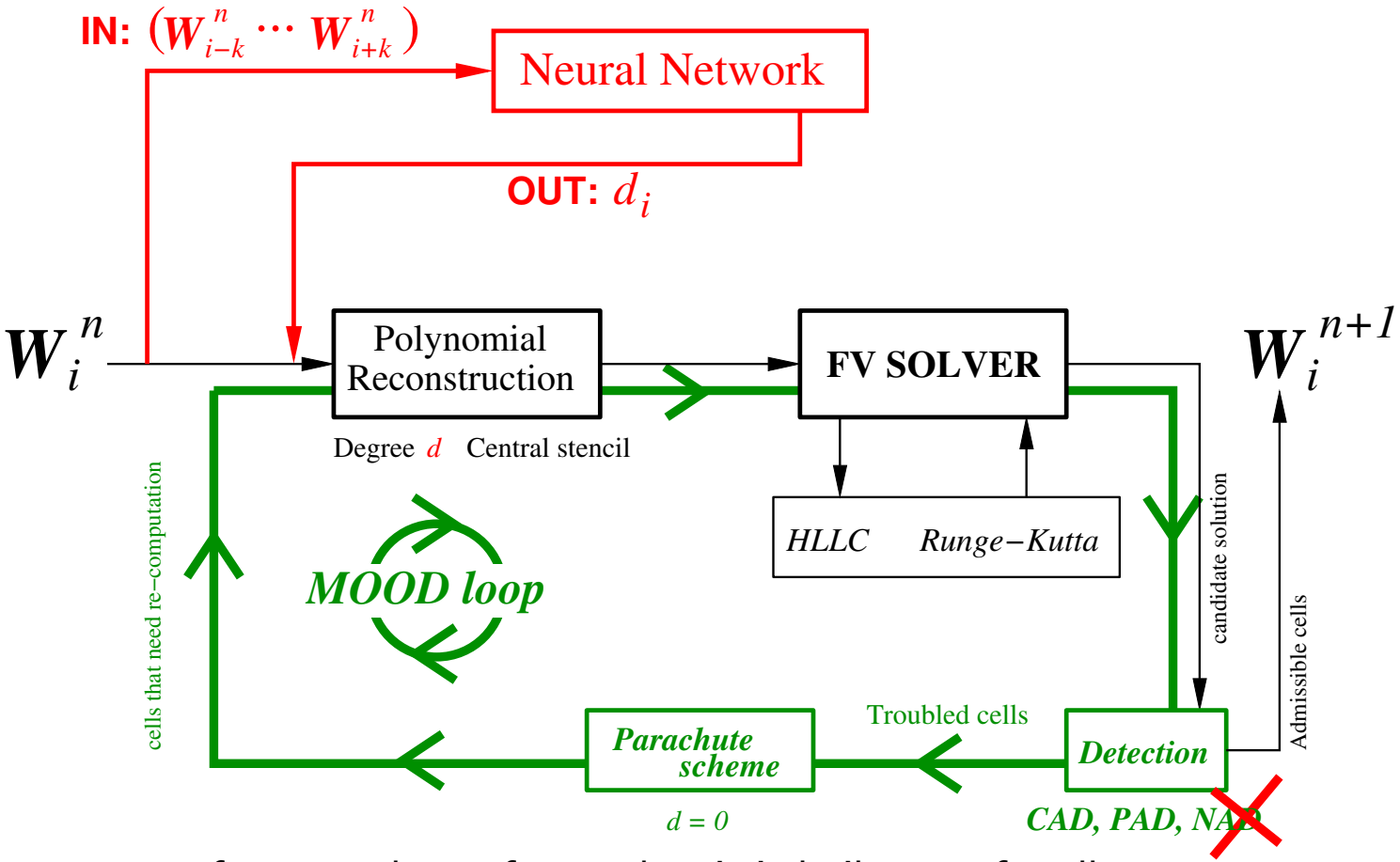
$G=$ 4 smooth and irregular shaped profiles: Gaussian, square, triangular shaped and continuous profile  Image of initial data in pdf or png format

# NN training for a 1D FV MOOD context

## How does it look like in practice?



# *a priori* NN and *a posteriori* MOOD scheme




- NN may come from another software, but it is built once for all
- NN calls: each cell at each time-step. NN evaluation cost depends on its architecture
- Ideally: NN prediction is 99% correct, MOOD loop only for CAD, PAD $\rightarrow$ almost never on
- $10-20\%$ of NAD troubled cell re-updates by MOOD loop are anticipated by the NN

# Numerical experiences

## Methodology of testing

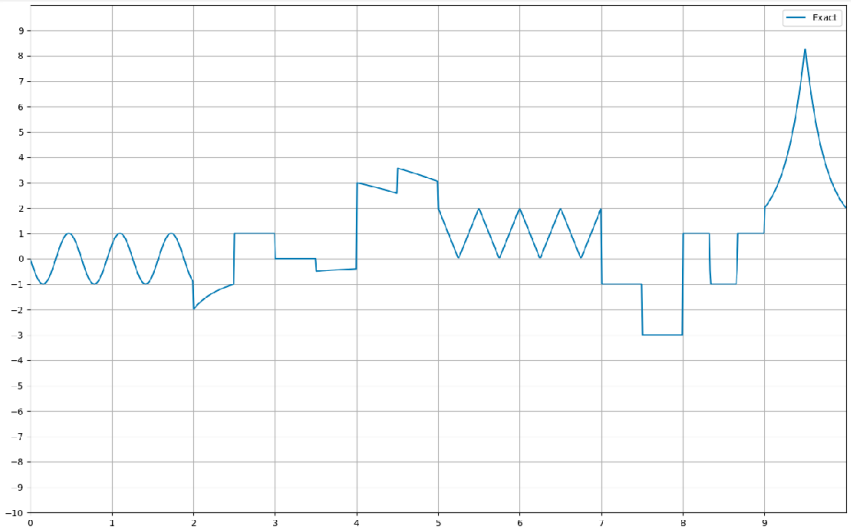
### NN architecture, training

$L = 3 + 2$ layers, hidden layers with 5 perceptrons, input with 7, ouput with 4.
Profil $G(x)$ (conpound wave) to train. NN validated with 200 success predictions.
Max degree 3, min degree 0.

### Methodology of testing — Compare the quality of numerical solutions

| Low Accurate/Robust | Accurate/ENO | High accurate/Oscillatory |
|---|---|---|
| 1st order God FV scheme | *a posteriori* MOOD<br>*a priori* NN ??? | Unlimited FV scheme |

Goal: NN reproduces MOOD results, measure of accuracy (eye-norm), cost, history of $d_i$



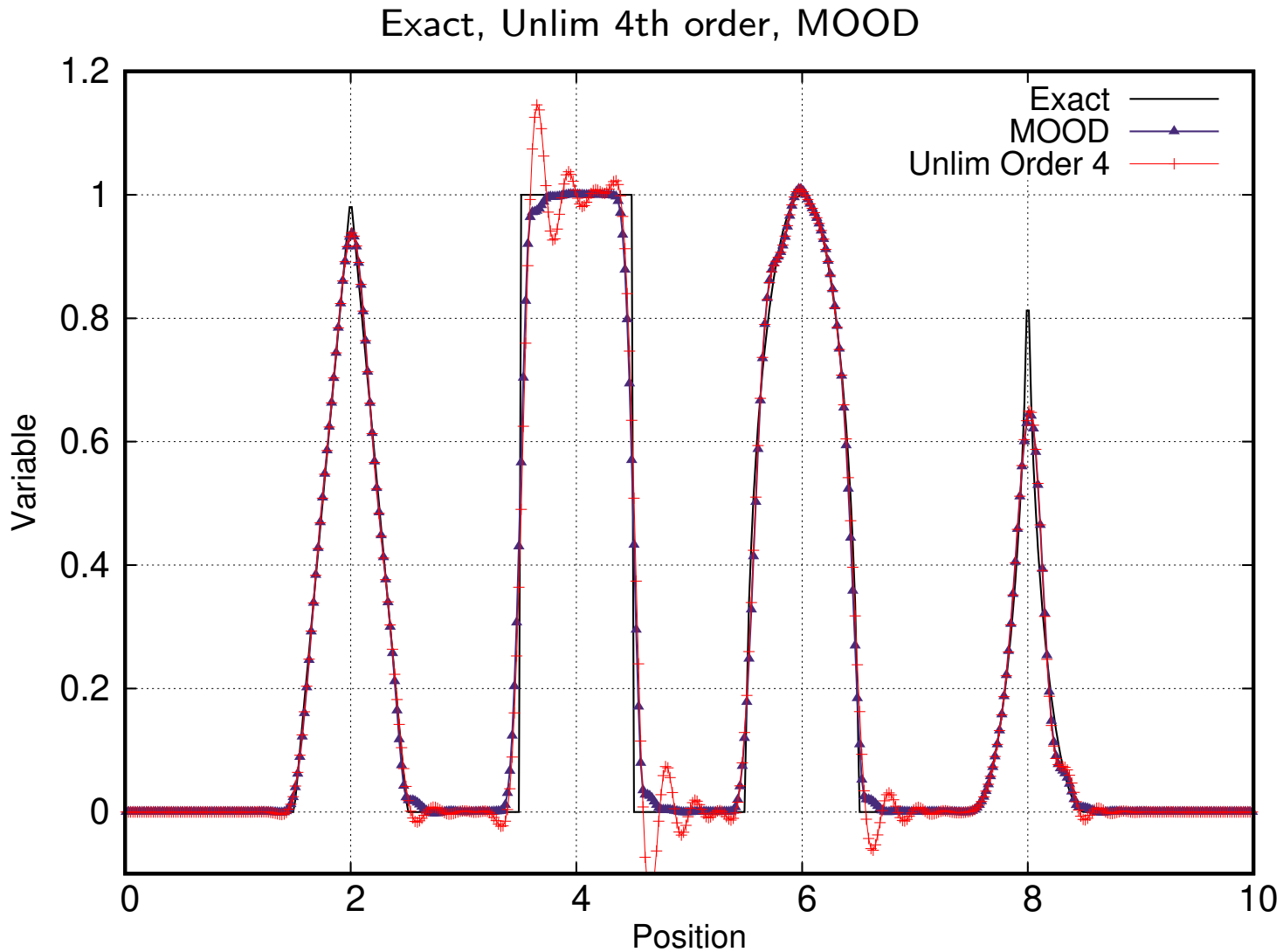

Simulation starting from an un-seen initial compound waves $G(x)$

Mix of sines, Heaviside, plateaus

# Numerical experiences I

## Advection equation with constant velocity $u = 1$

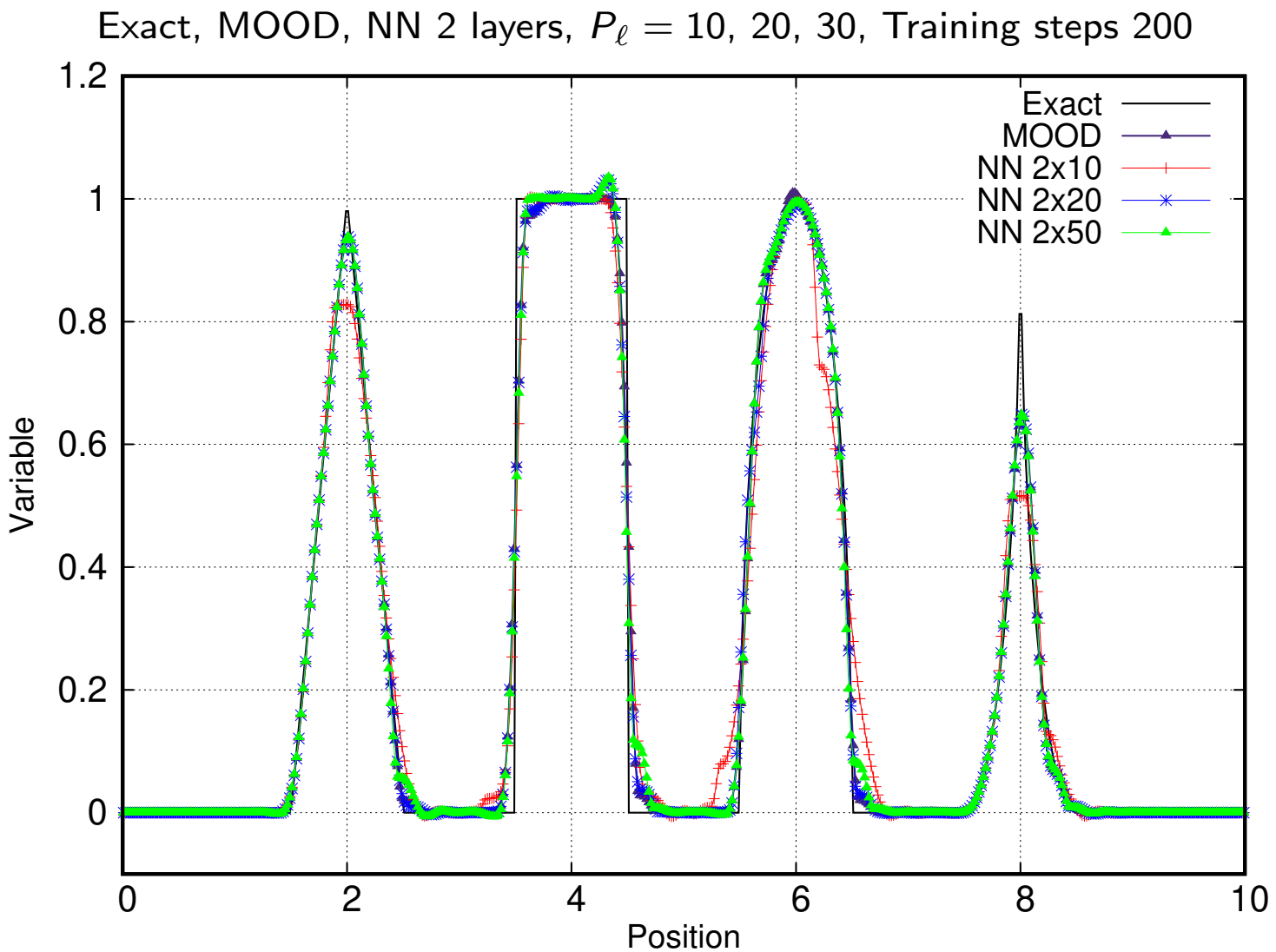
Five rotations of the initial profile, 500 cells, $d_i = 3, 2, 1, 0$, RK3



# Numerical experiences I

## Advection equation with constant velocity $u = 1$

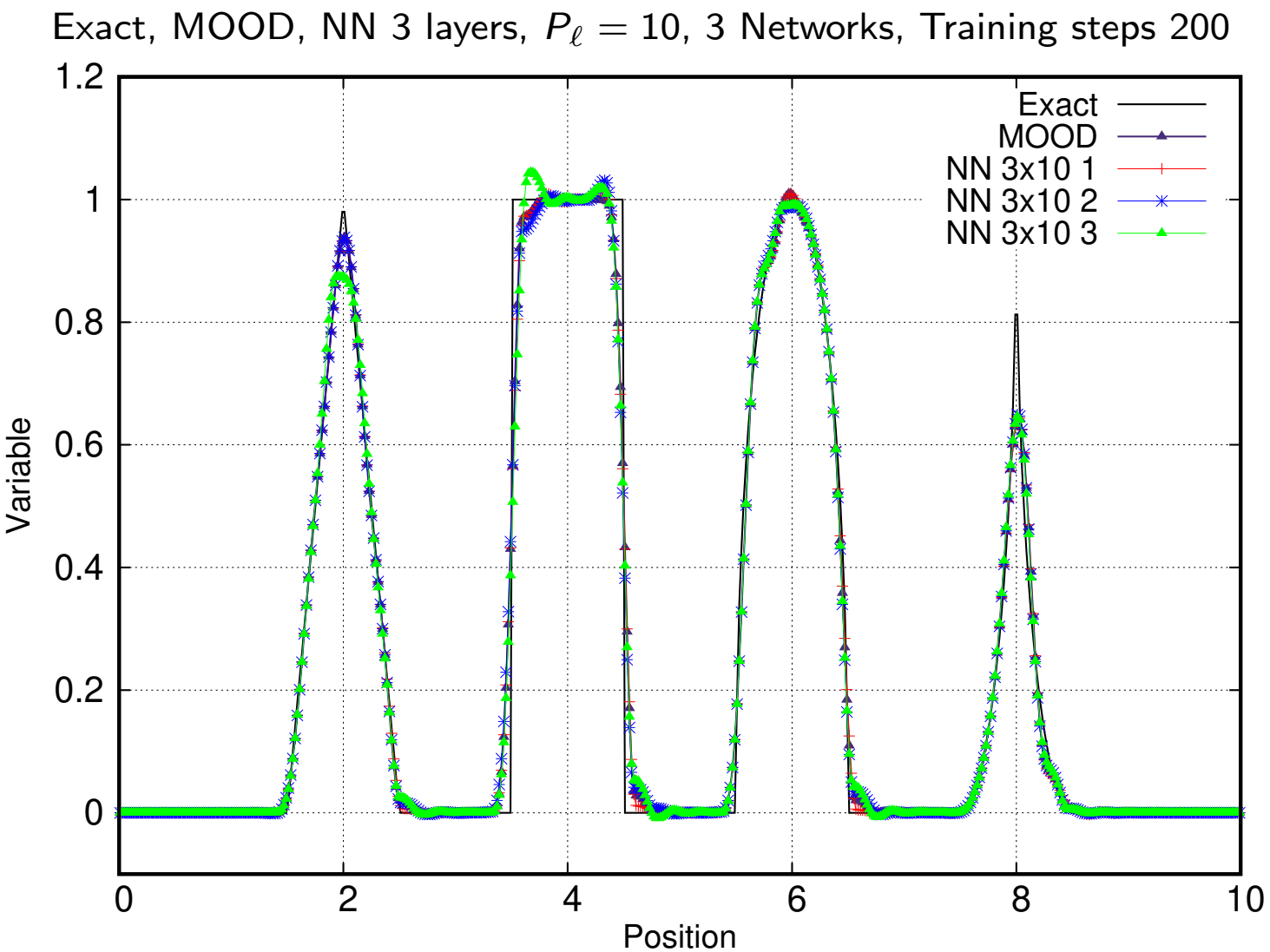
Five rotations of the initial profile, 500 cells, $d_i = 3, 2, 1, 0$, RK3



# Numerical experiences I

## Advection equation with constant velocity $u = 1$

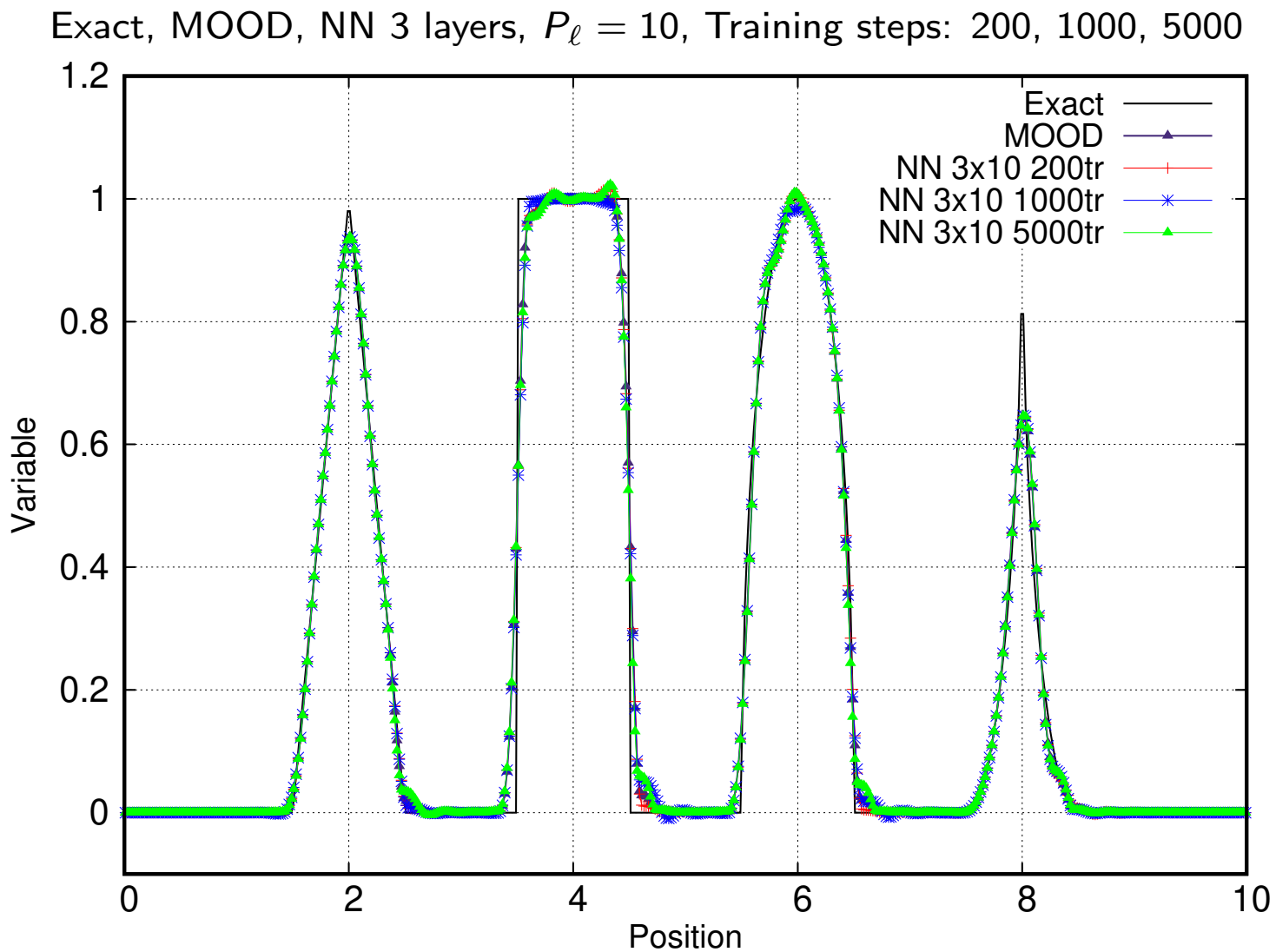
Five rotations of the initial profile, 500 cells, $d_i = 3, 2, 1, 0$, RK3



# Numerical experiences I

## Advection equation with constant velocity $u = 1$

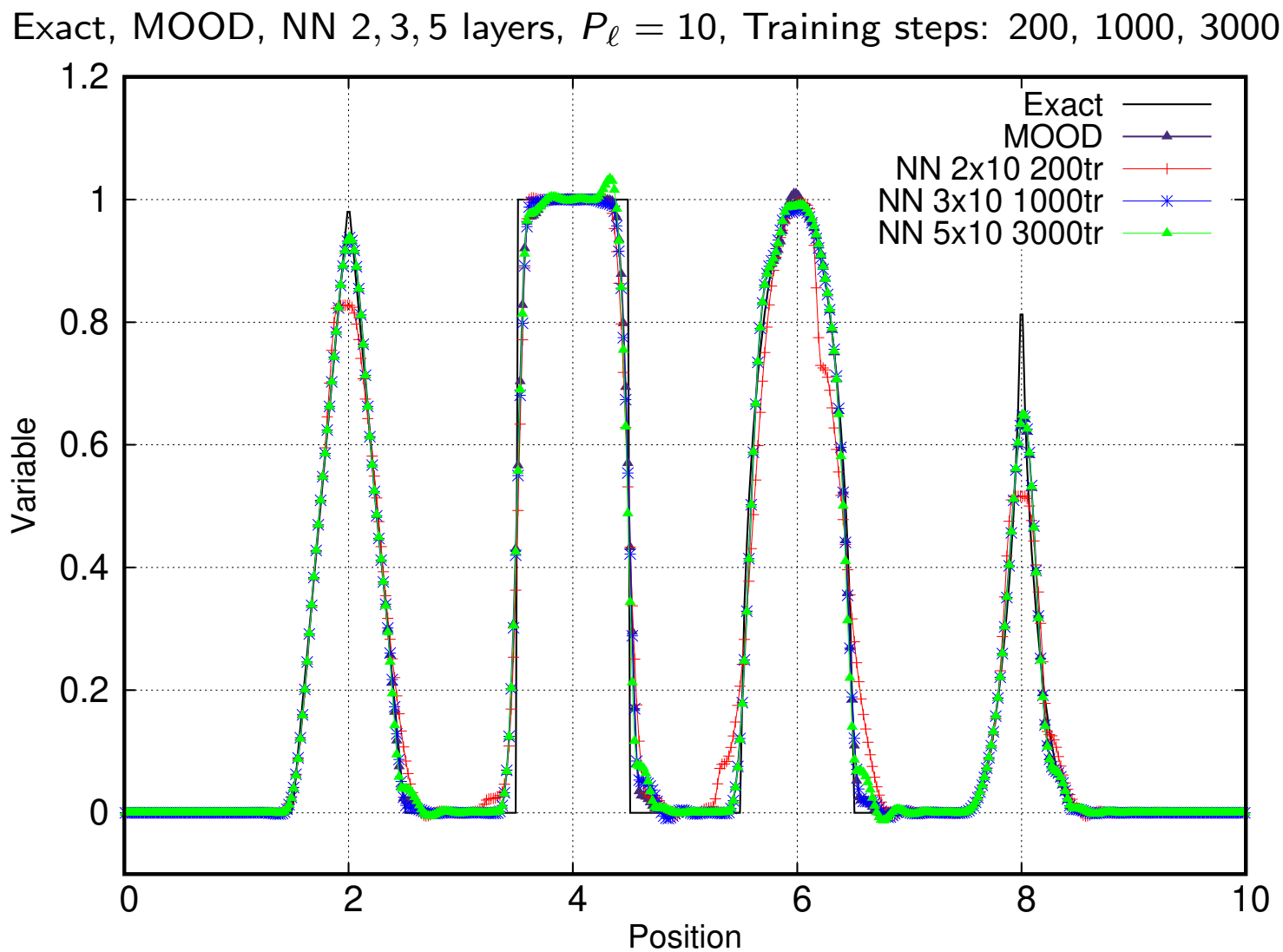
Five rotations of the initial profile, 500 cells, $d_i = 3, 2, 1, 0$, RK3



# Numerical experiences I

## Advection equation with constant velocity $u = 1$

Five rotations of the initial profile, 500 cells, $d_i = 3, 2, 1, 0$, RK3



# Numerical experiences

## Hydrodynamics system of PDEs

### Scheme and NN architecture

NN validated with 200 success predictions over 5000.
Polynomial degree for the density $\rho$ is predicted by NN, and all conservative variables adopt this degree (like in MOOD). Max polynomial degree 3, min degree 0
Test 2 architectures:
arch1= $L = 3 + 2$ layers, hidden layers with 20 perceptrons, input with 7, ouput with 4.
arch2= $L = 3 + 2$ layers, hidden layers with $50, 20, 10$ perceptrons, input with 7, ouput with 4.

### Training situations: Adv=advection profile, Rie=set of Riemann problem simple waves, or Mix

Adv=$H(x)$ (mix sin, Heaviside) for $\rho$, $u = 0$, $p = \rho^\gamma$,

Rie=

| | $\rho_L$ | $u_L$ | $p_L$ | $\rho_R$ | $u_R$ | $p_R$ | |
|---|---|---|---|---|---|---|---|
| RP1 | 2 | 0 | 2 | 1 | 0 | 1 | $\rightarrow$ RP4=RP1 sym |
| RP2 | 2 | -1 | 1 | 2 | 1 | 1 | $\rightarrow$ RP5=RP2 sym |
| RP3 | 1 | 0 | 2 | 1 | 0 | 1 | $\rightarrow$ RP6=RP3 sym |

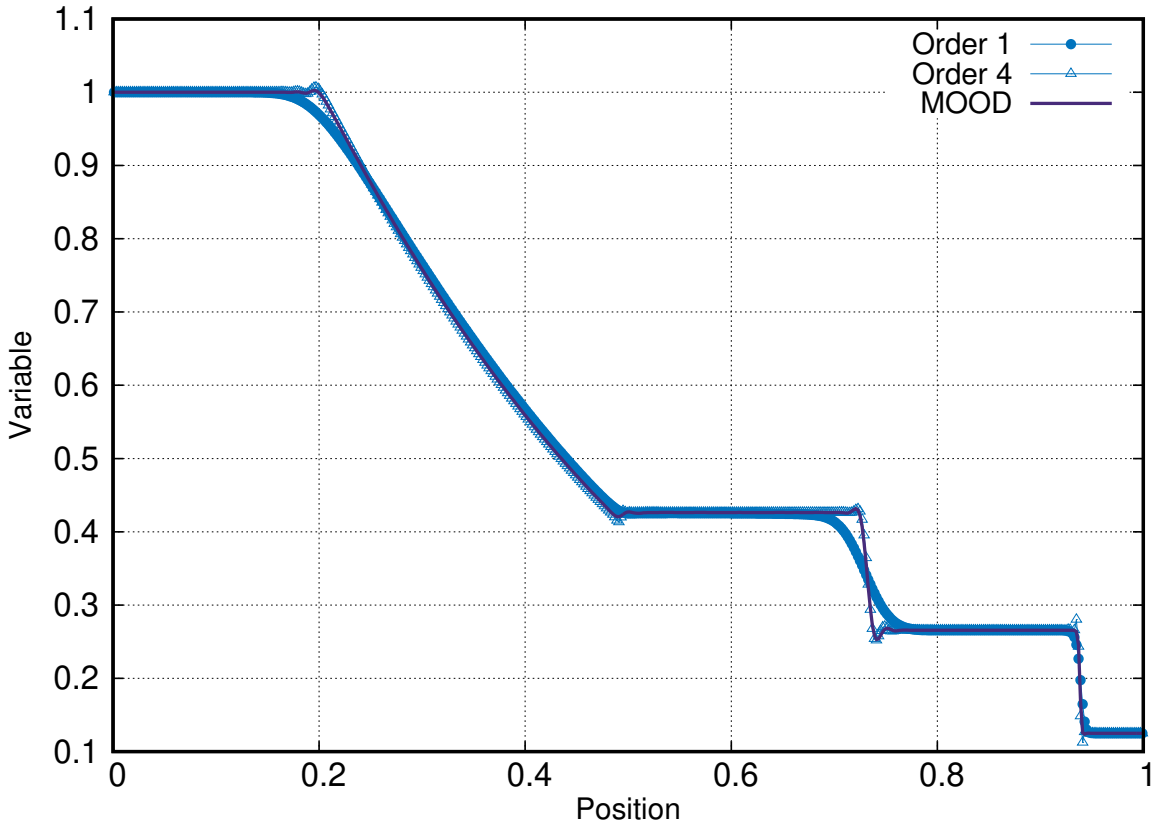
### Simulations and diagnostics

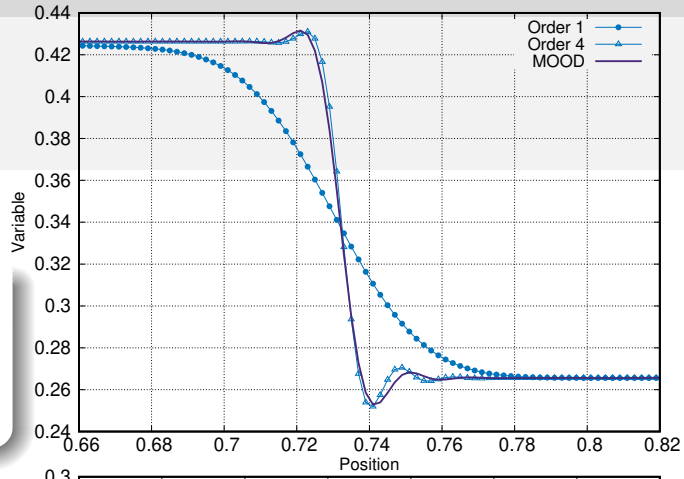
Sod, Blastwave (Collela-Woodward) test problems. Diagnostics: NN solution $\simeq$ MOOD solution?

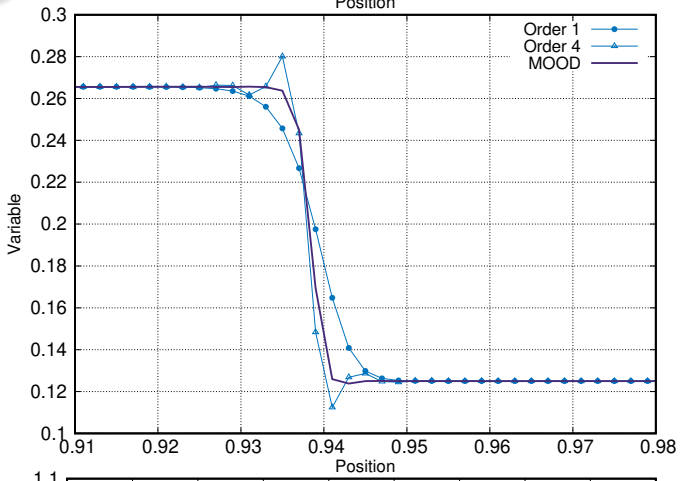
# Numerical experiences

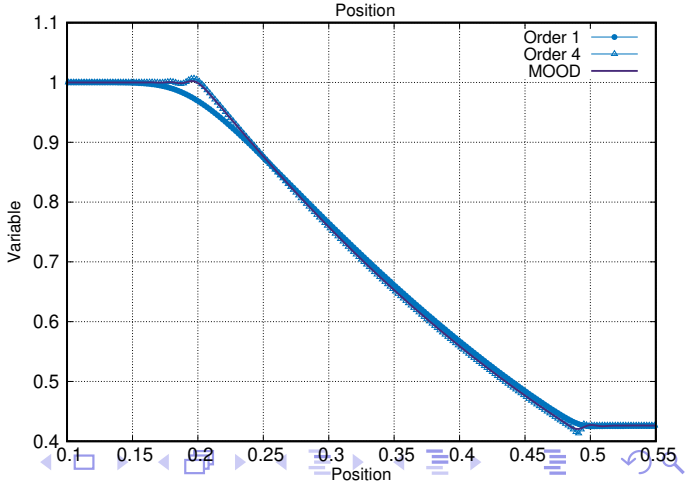
## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution



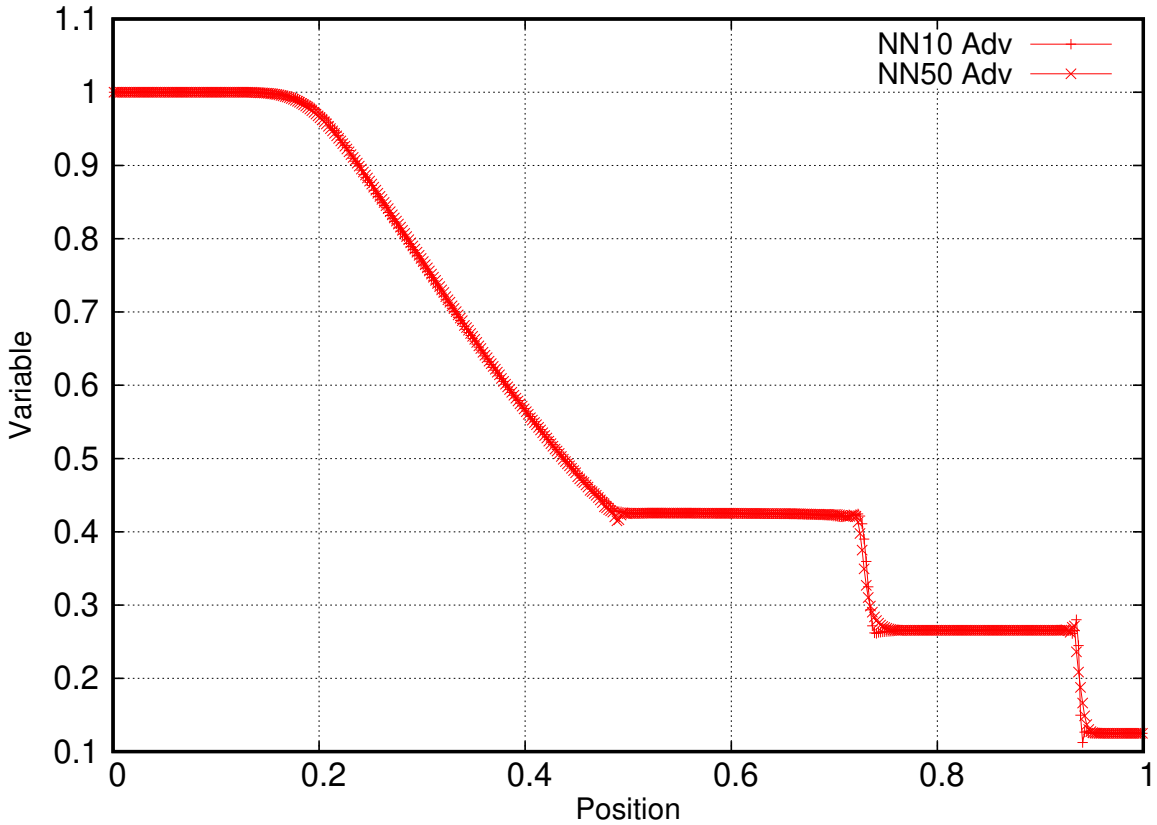



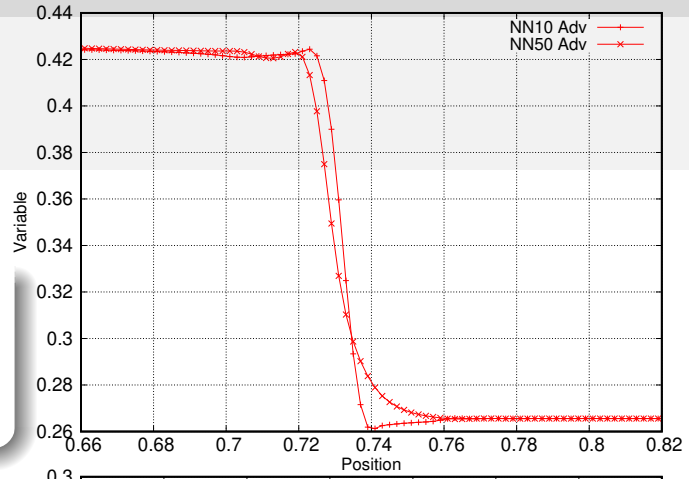



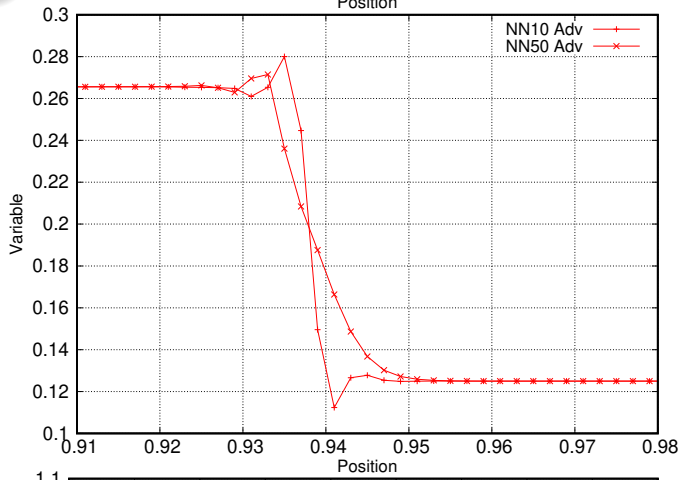
# Numerical experiences

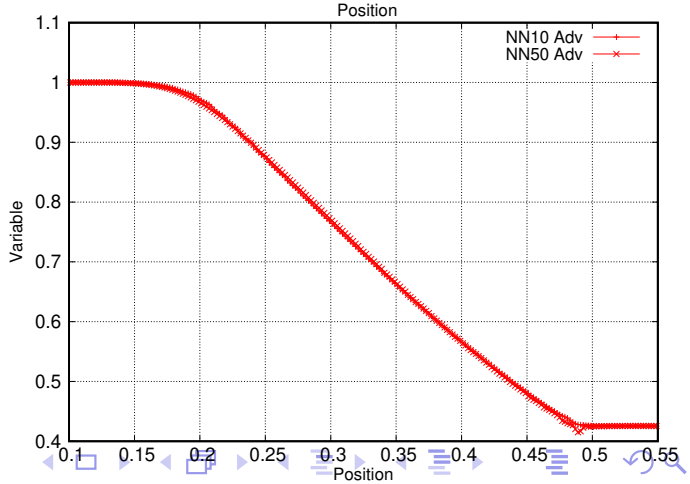
## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution



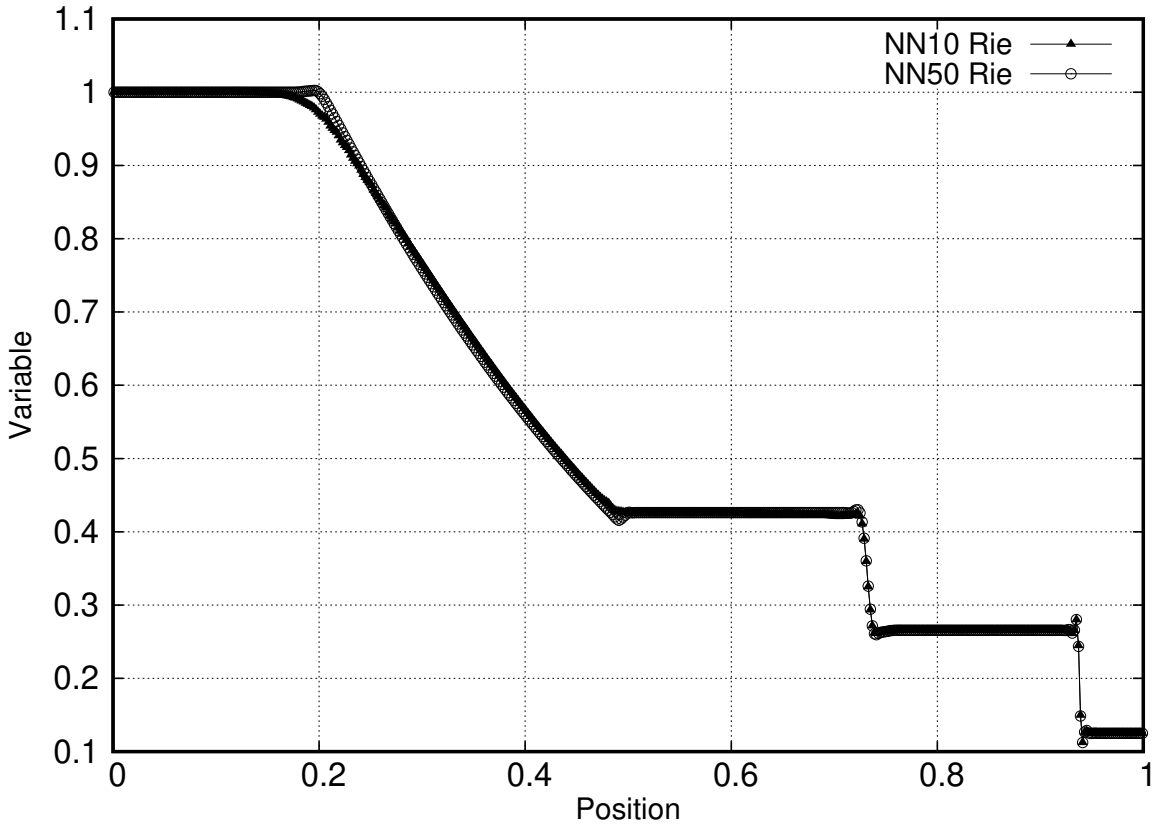



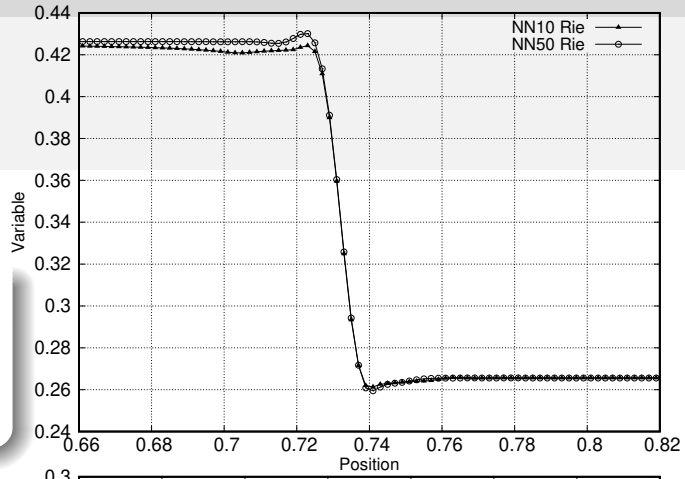



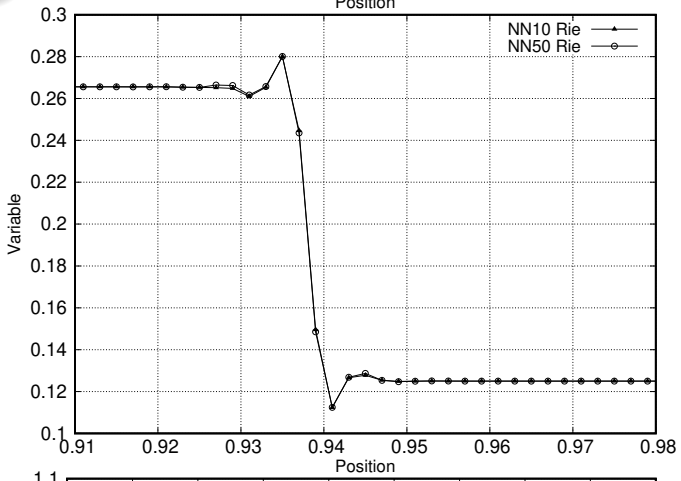
# Numerical experiences

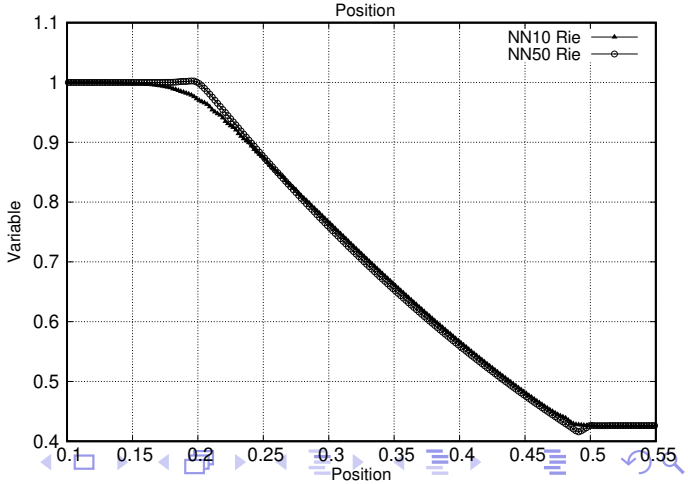
## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution



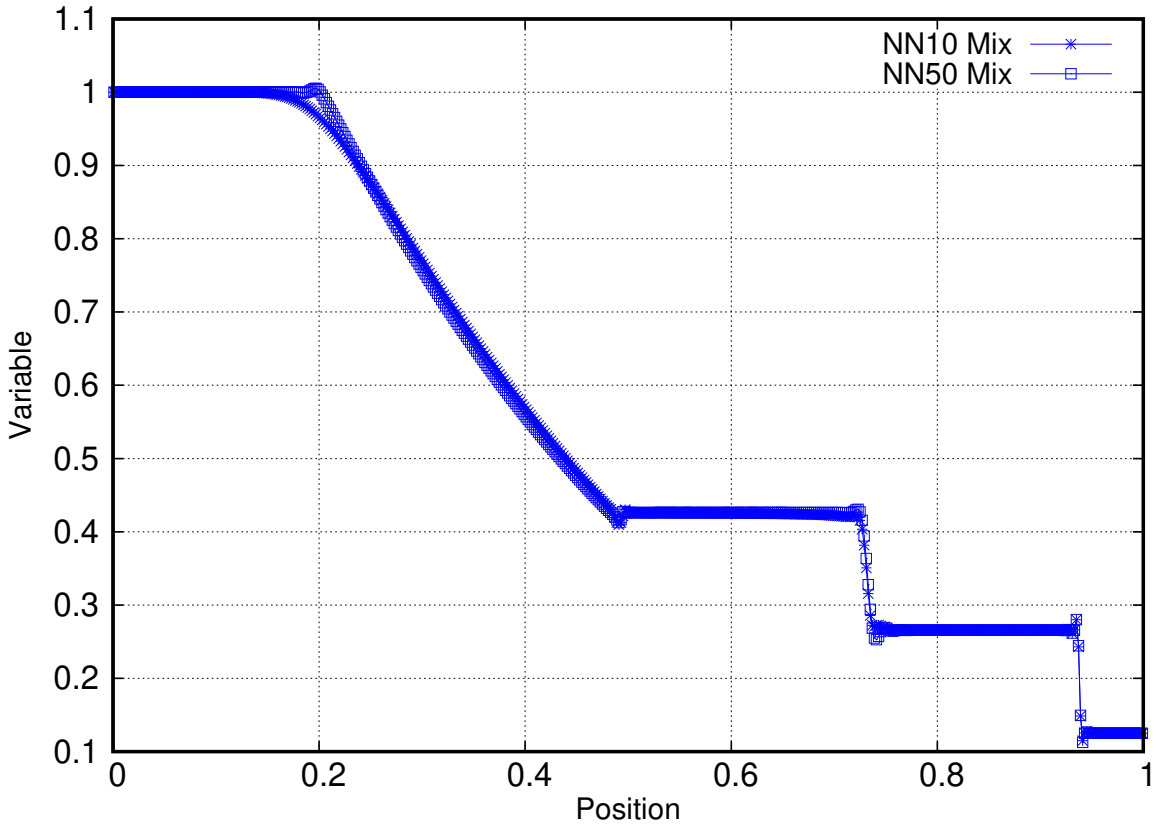



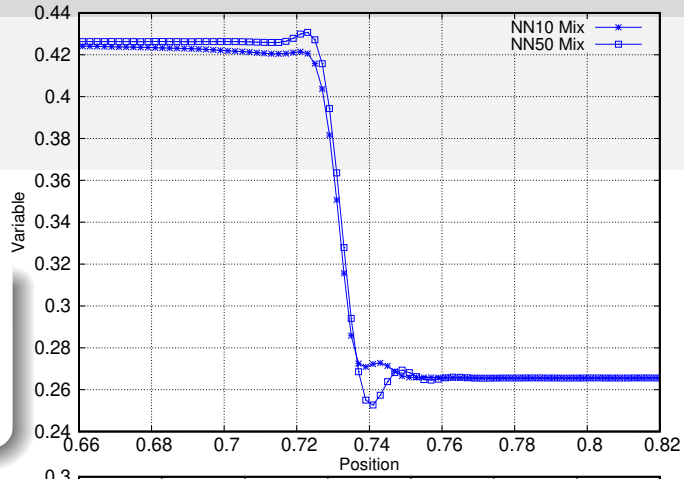



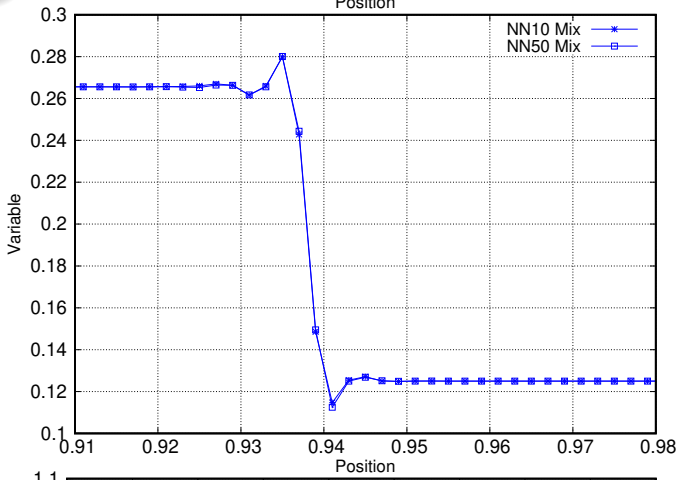
# Numerical experiences

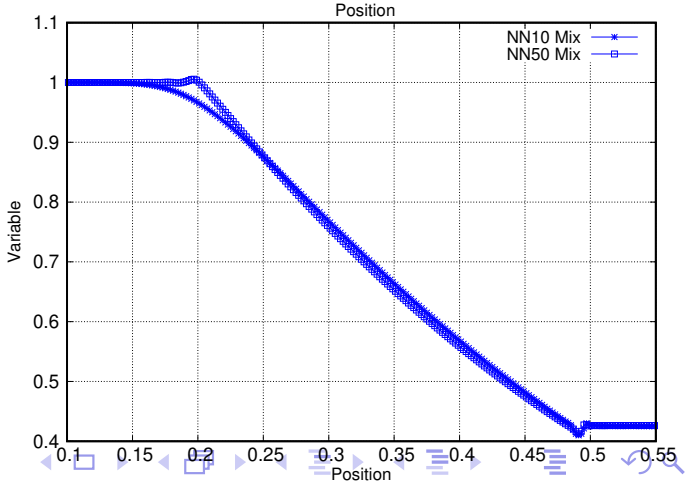
## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution



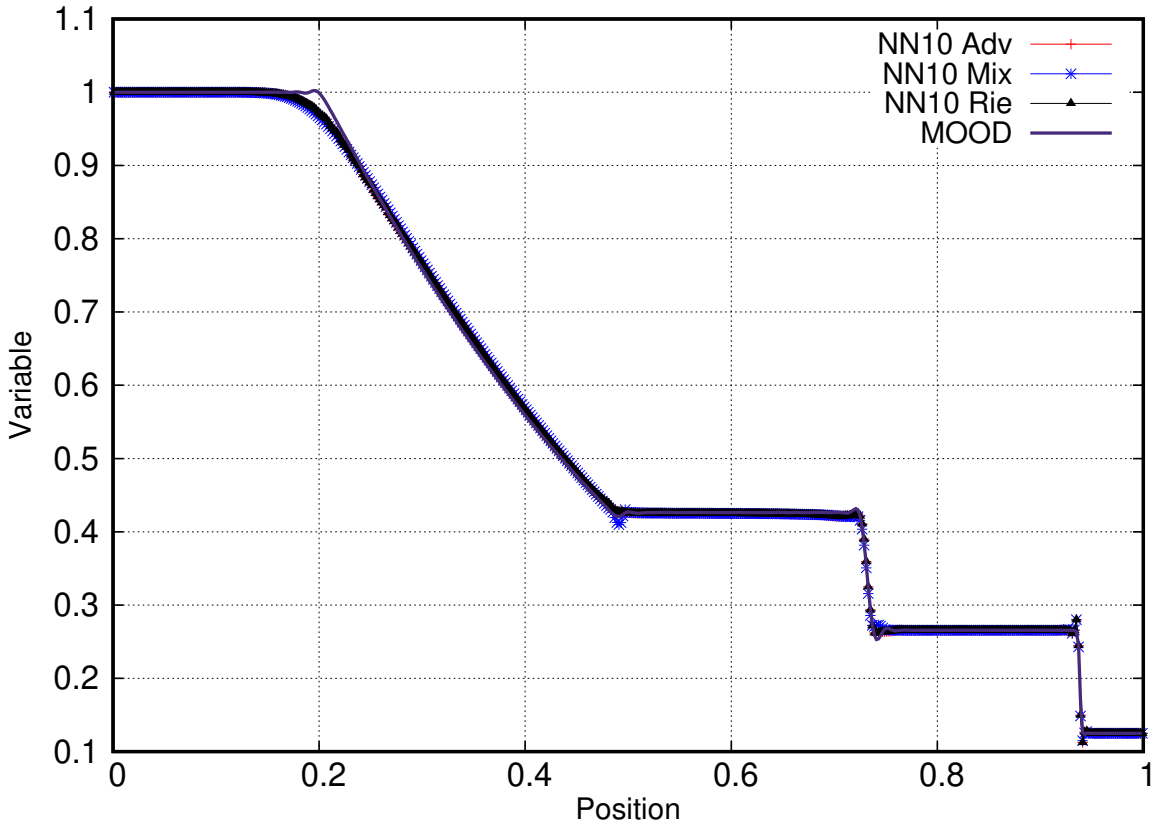



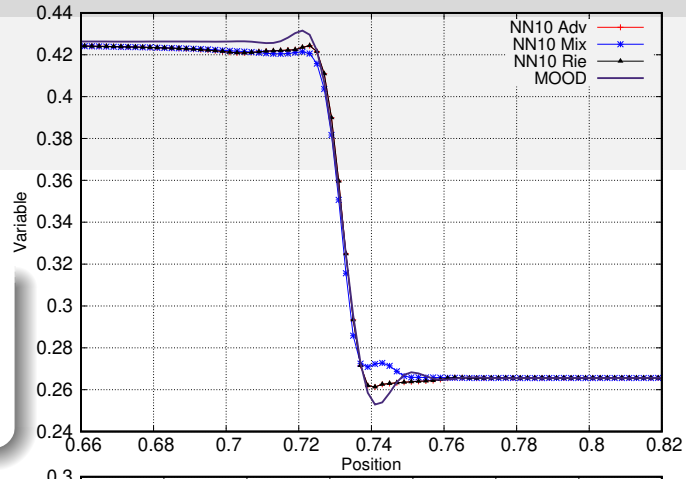



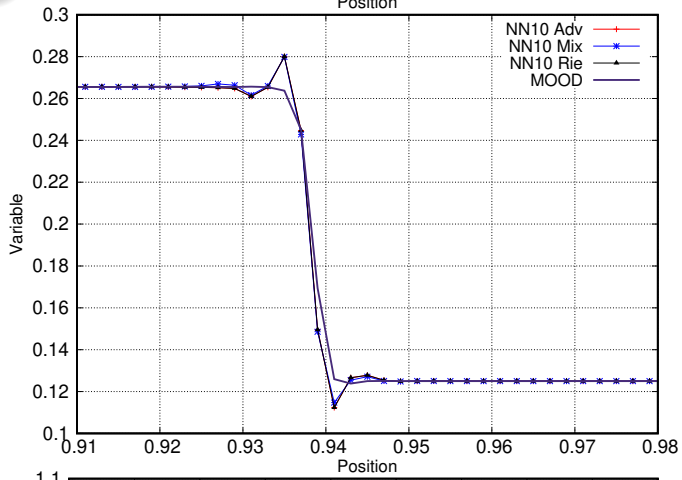
# Numerical experiences

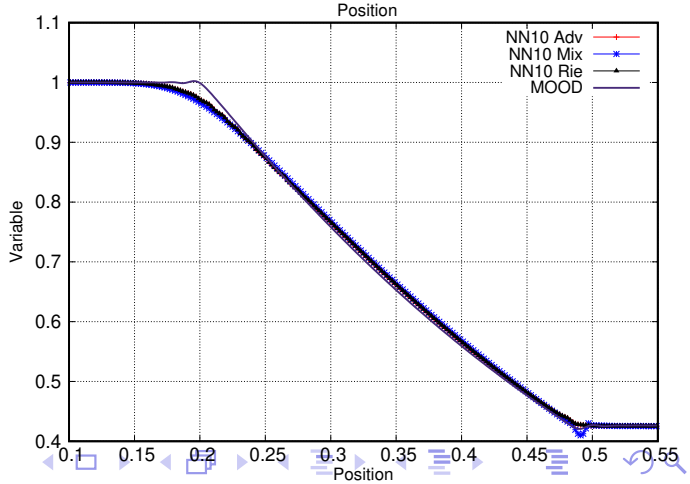
## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution



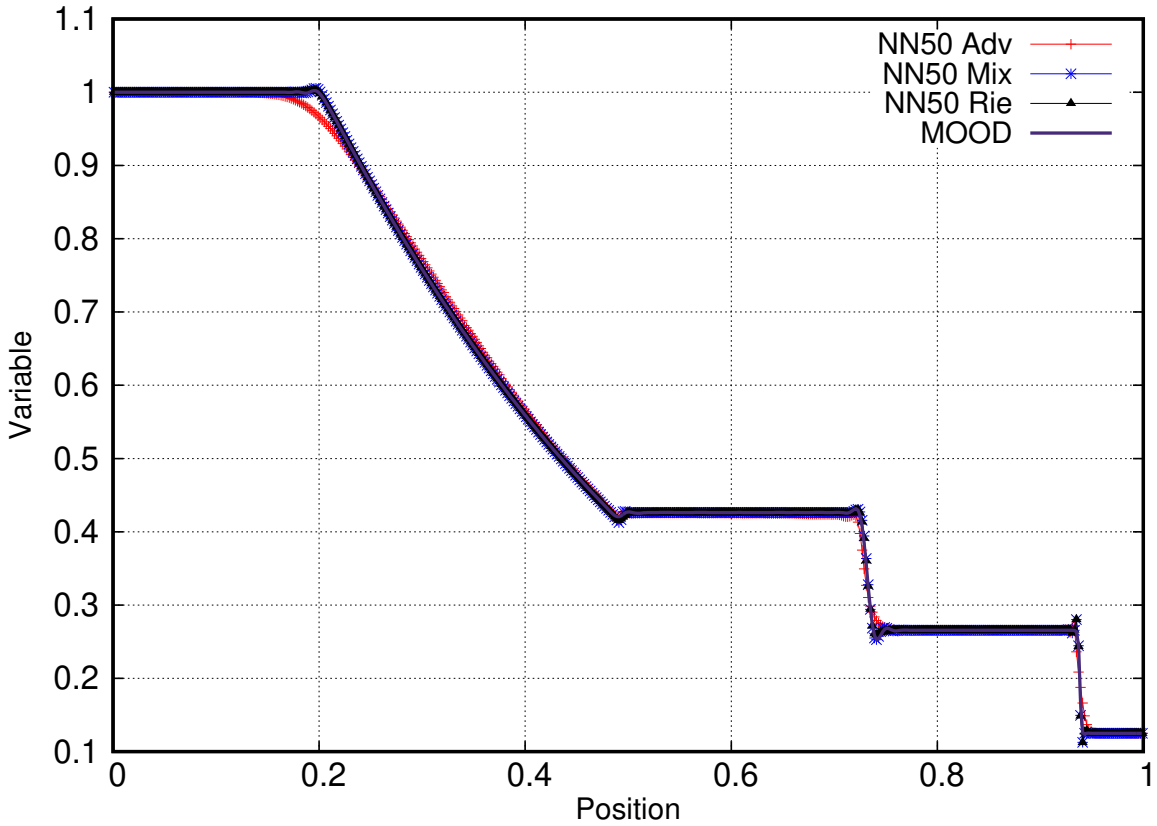



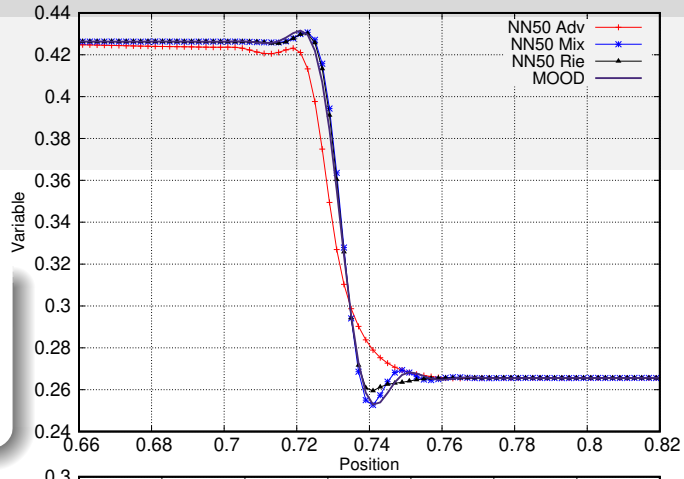



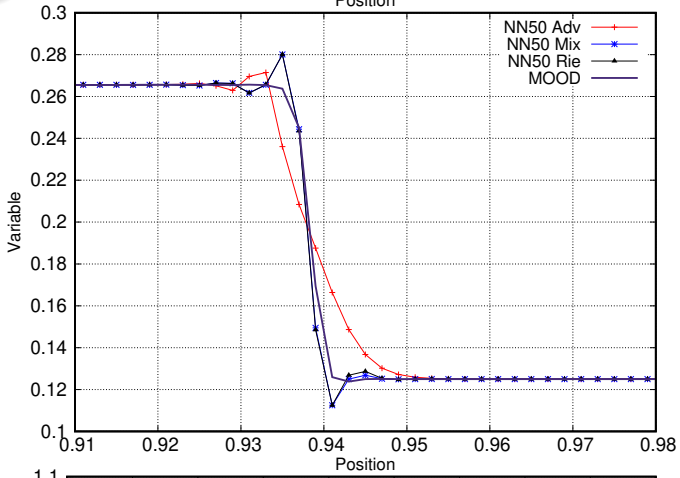
# Numerical experiences

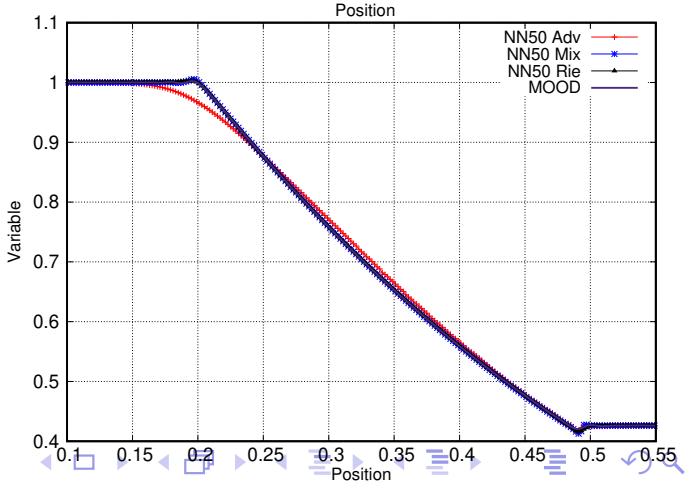
## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution

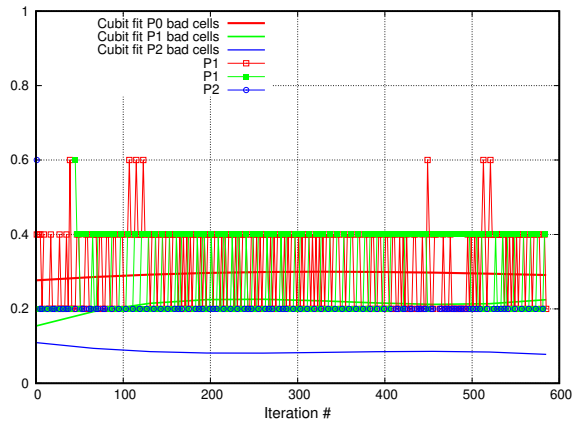


## MOOD bad cells - Sod

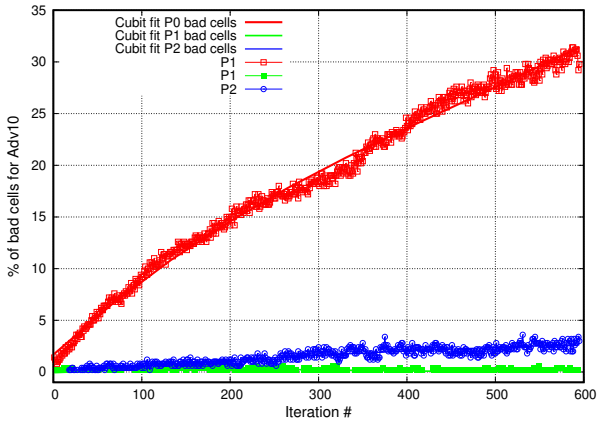


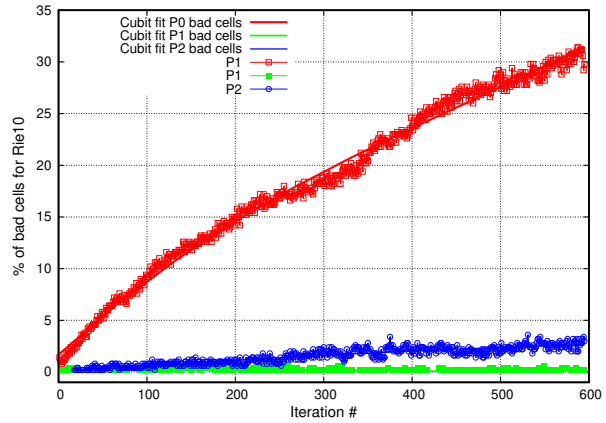

## Advection fct

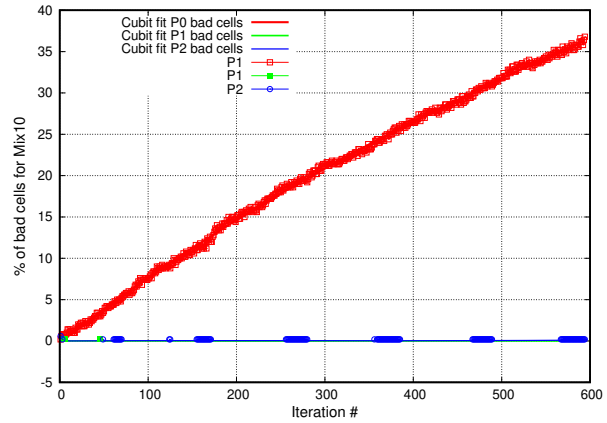
## Riemann problems

## Mix fct

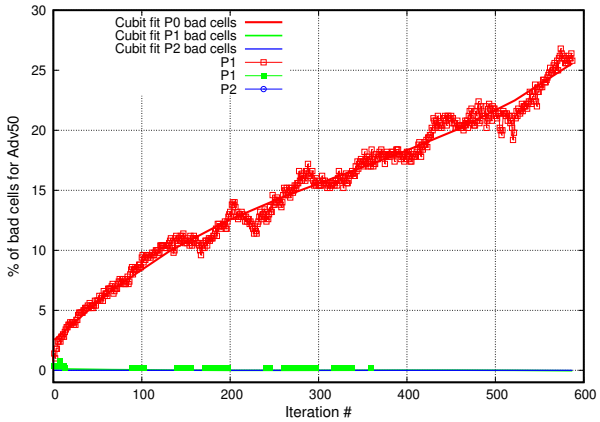
NN 10 × 10 × 10

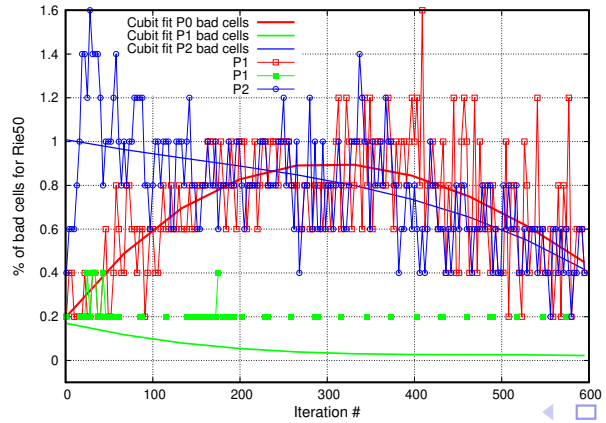


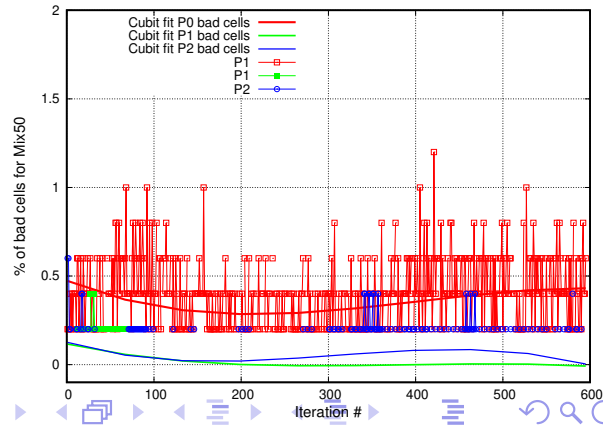




NN 50 × 20 × 10



# Numerical experiences

## Sod problem: simple waves

$(\rho, u, p)_L = (1, 0, 1)$, $(\rho, u, p)_R = (0.125, 0, 1)$, 500 uniform cells, $T = 0.1$, exact solution

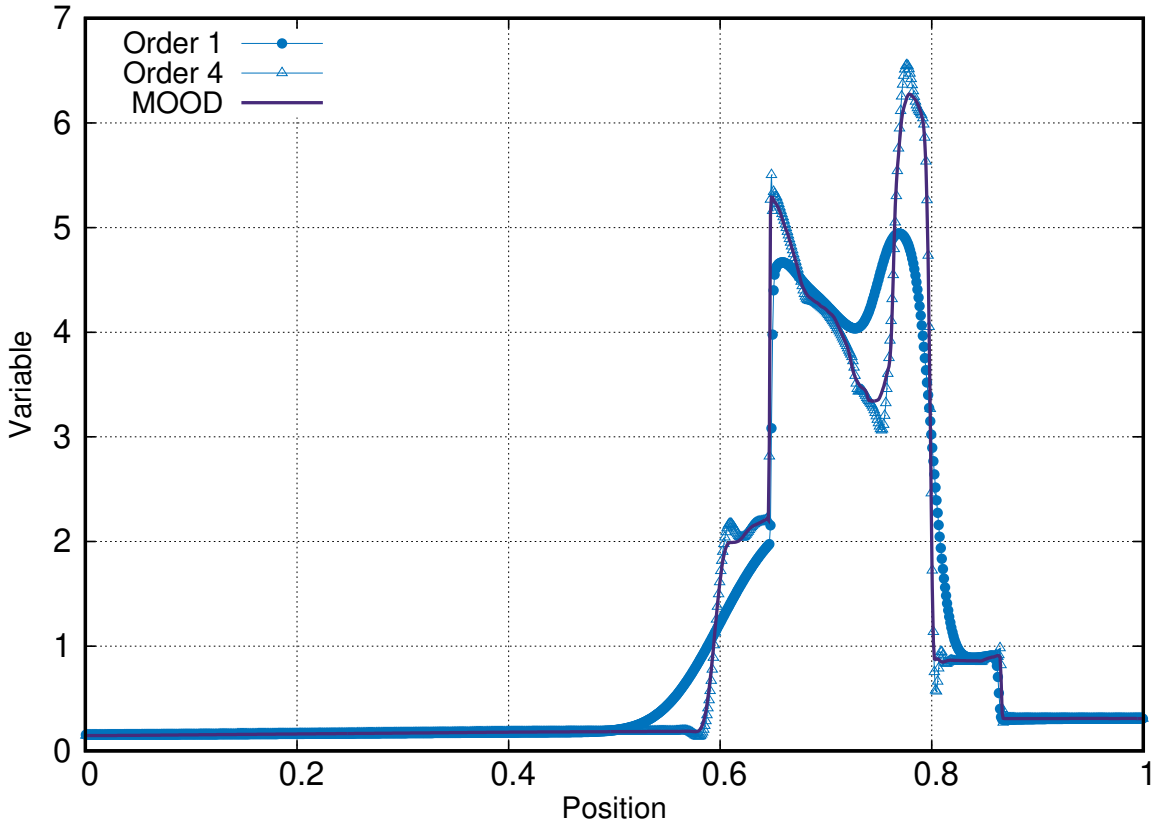
## Observations

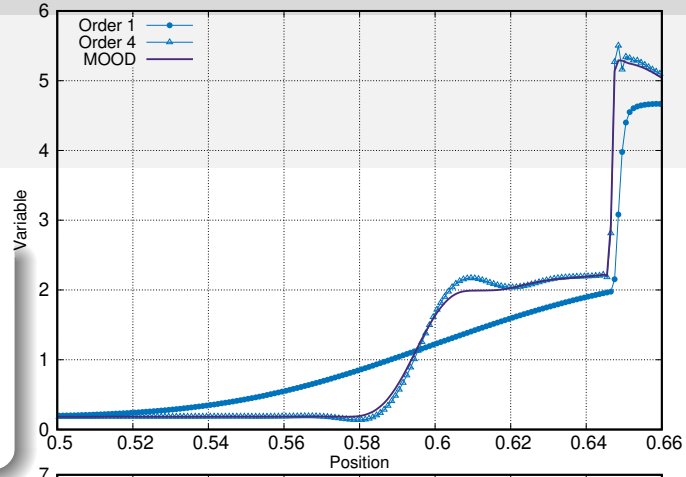
1- Difficulties to perform like MOOD, to reduce the oscillations at the shock and at the same time not spoiling the rarefaction/contact.
2- Architecture: large network (50) captures the rarefaction head, not the small one but is monotonic
3- Training: advection profile seems sufficient provided a large network is employed (apart from rarefaction head)
4- Often the NN catches the unlimited 4th order of 1st order behavior.

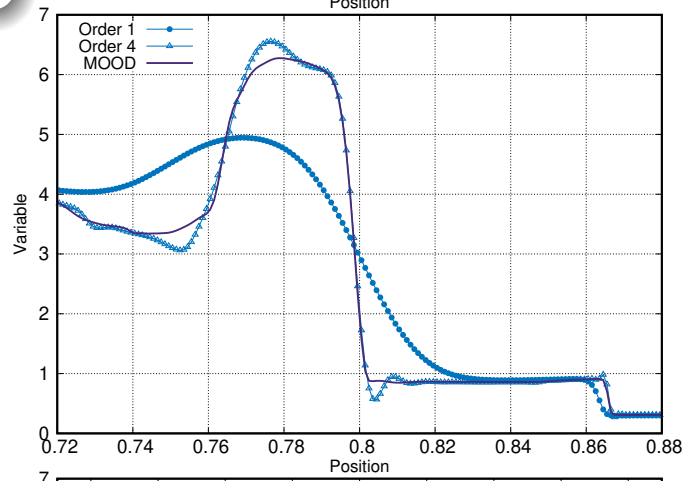
# Numerical experiences

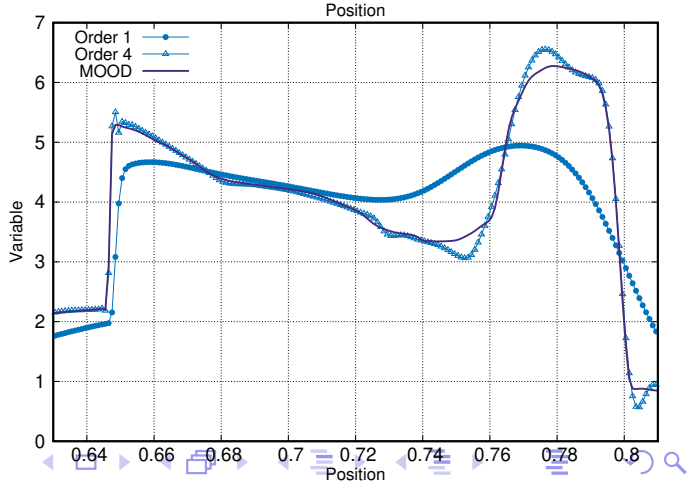
## Blastwave problem: interaction of waves

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution



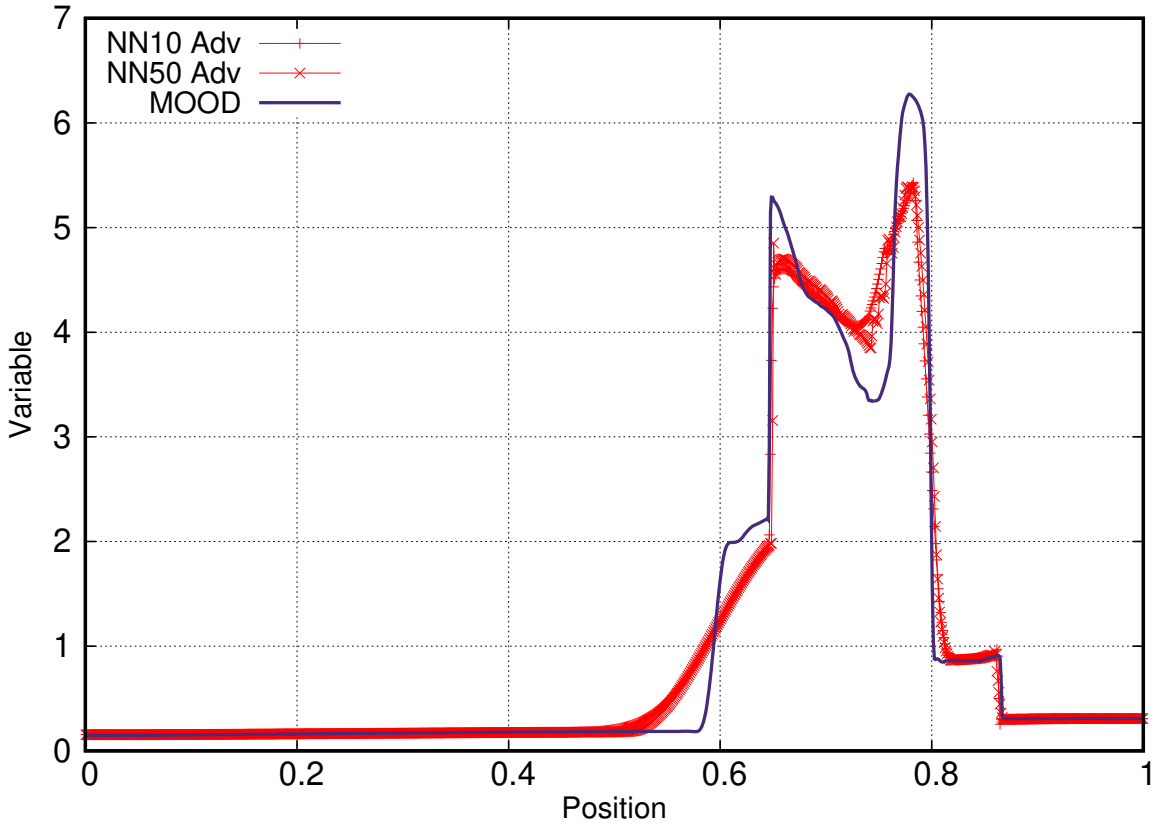



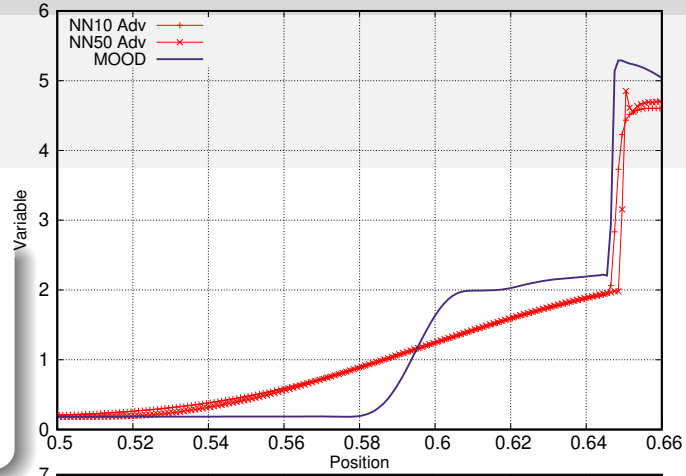



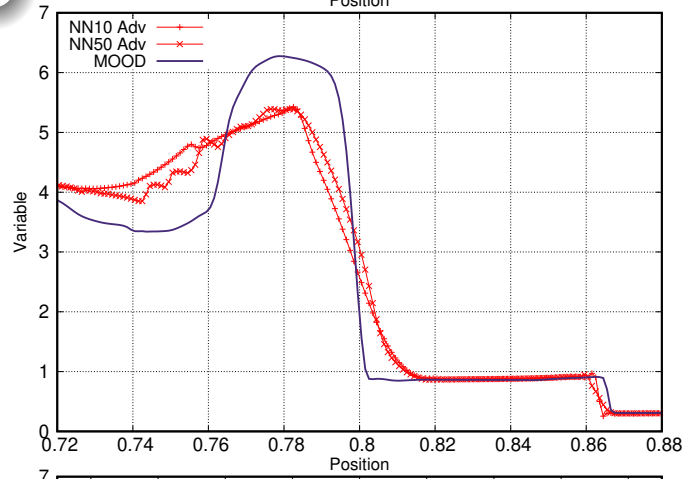
# Numerical experiences

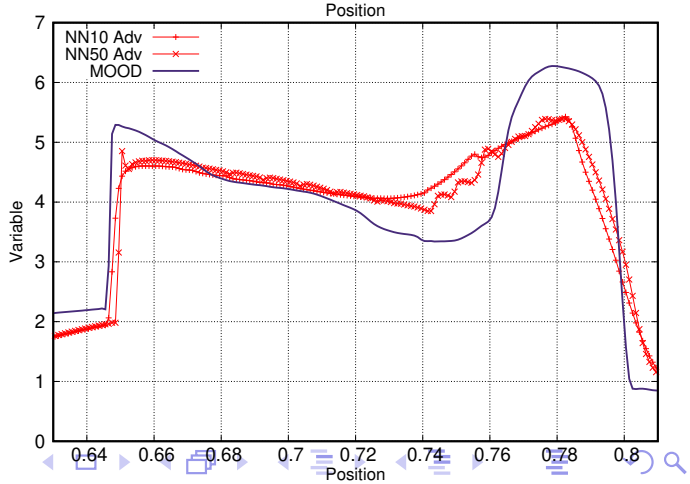
## Blastwave problem: interaction of waves

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution



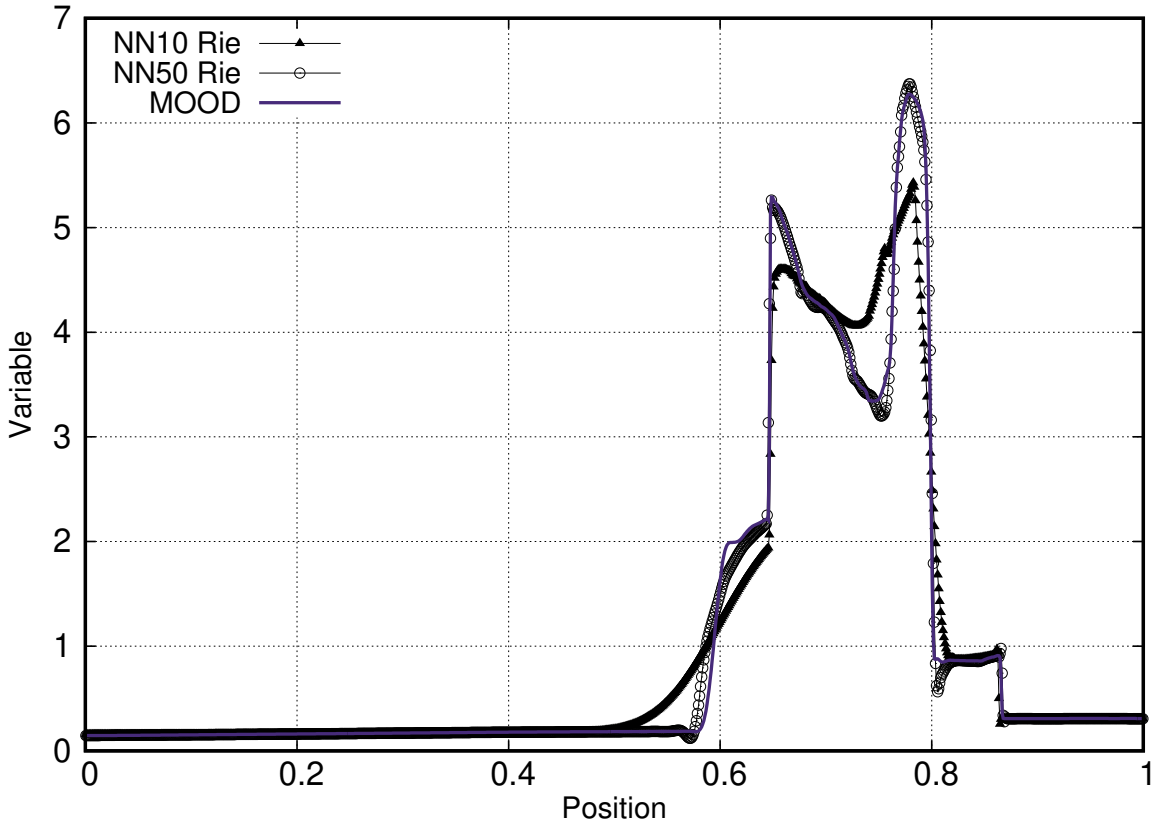



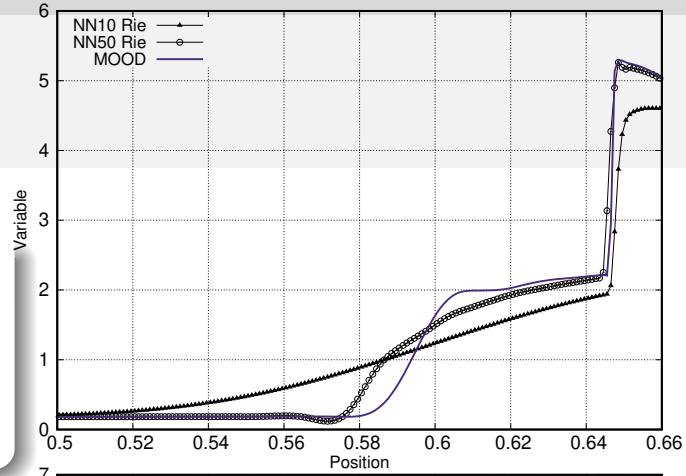



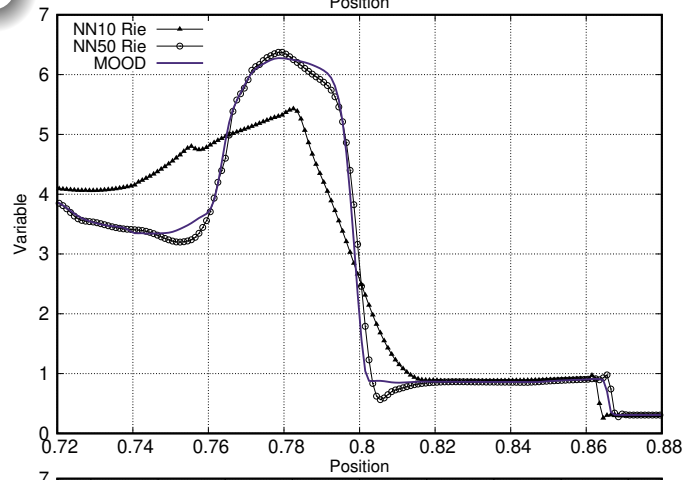
# Numerical experiences

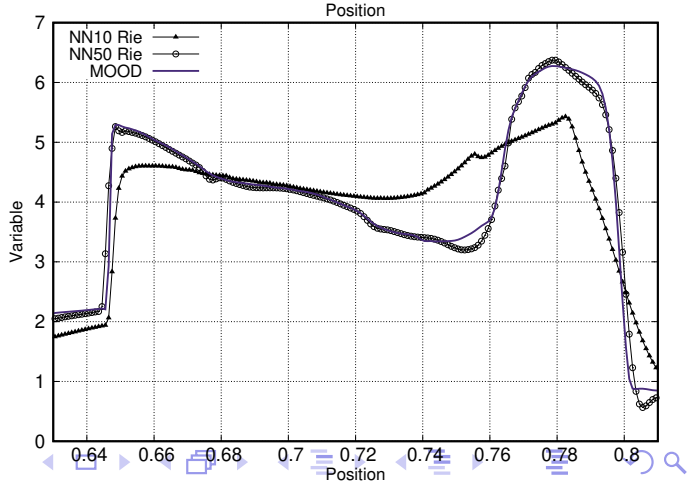
## Blastwave problem: interaction of waves

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution



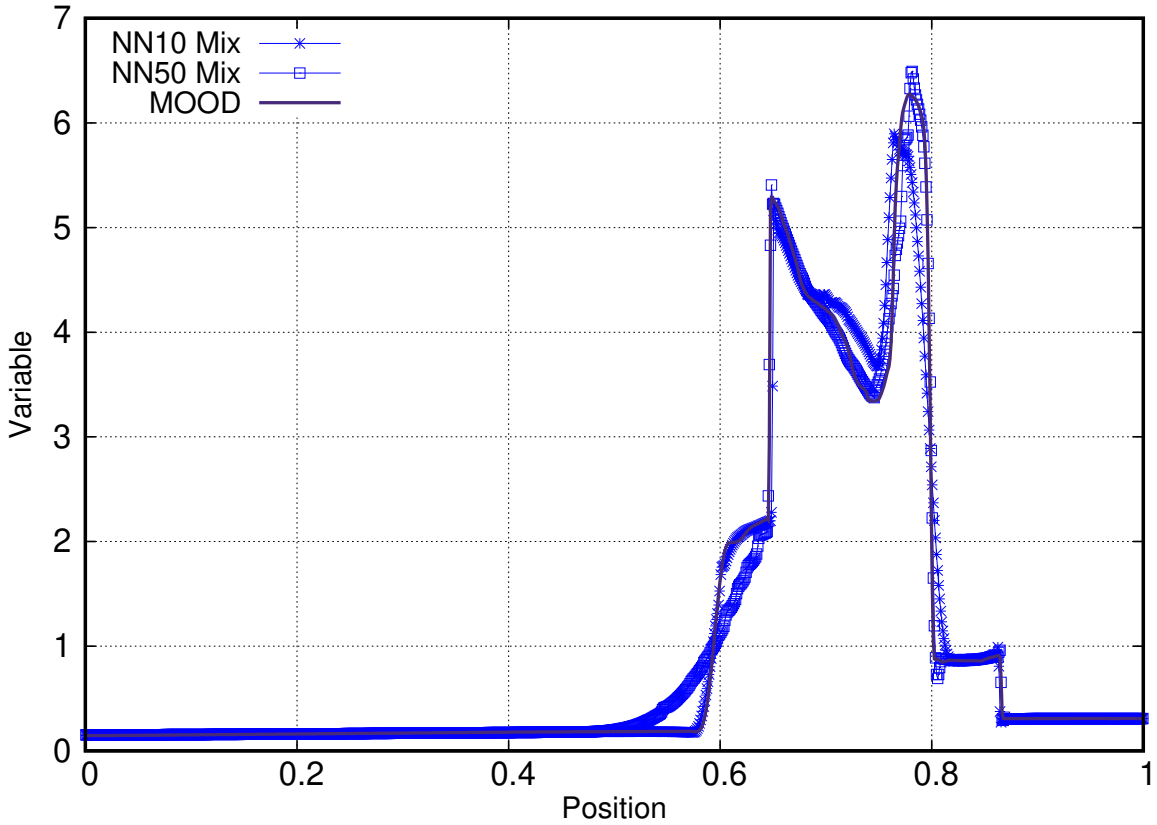



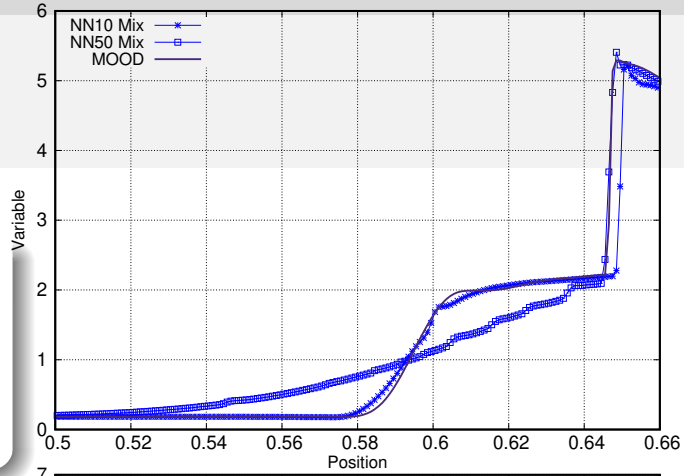



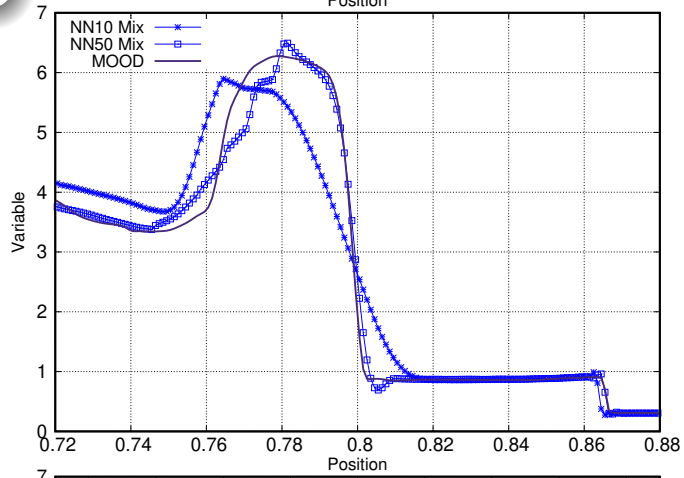
# Numerical experiences

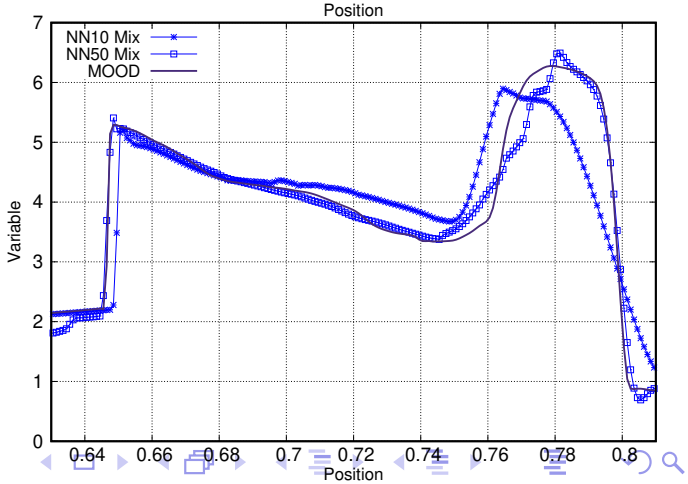
## Blastwave problem: interaction of waves

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution



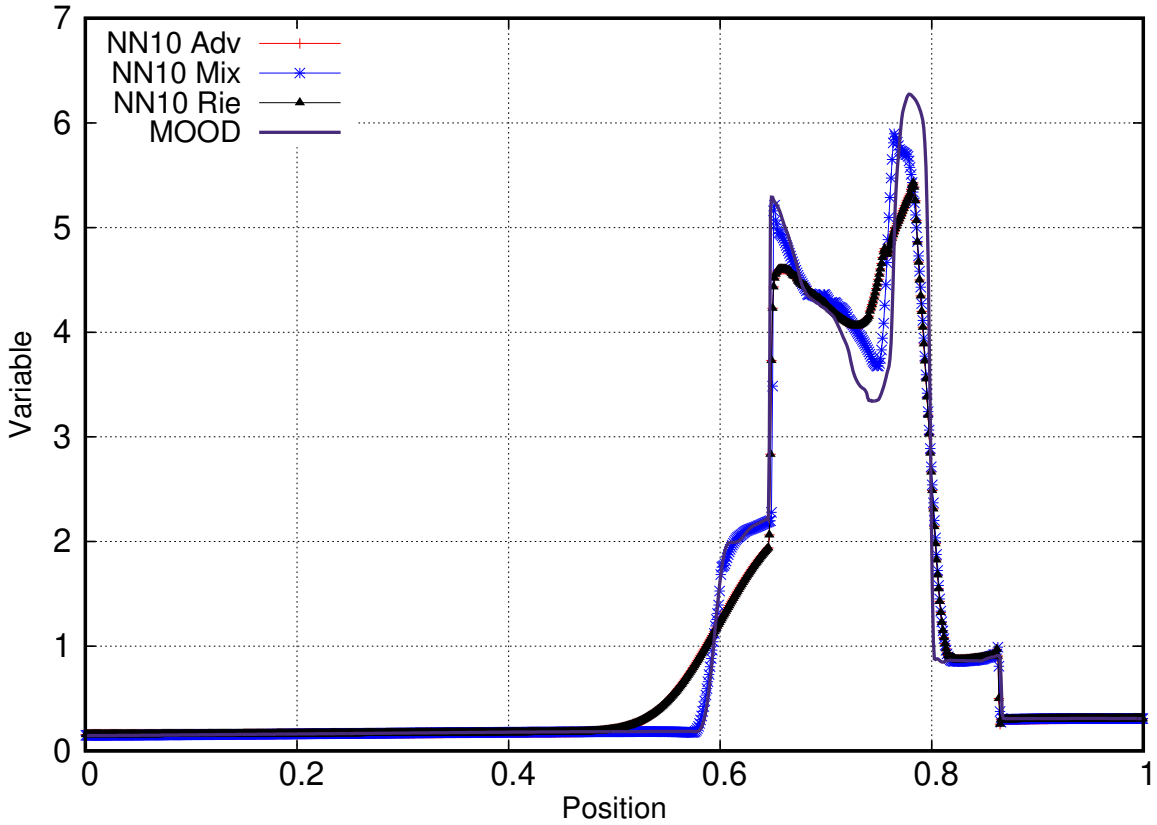



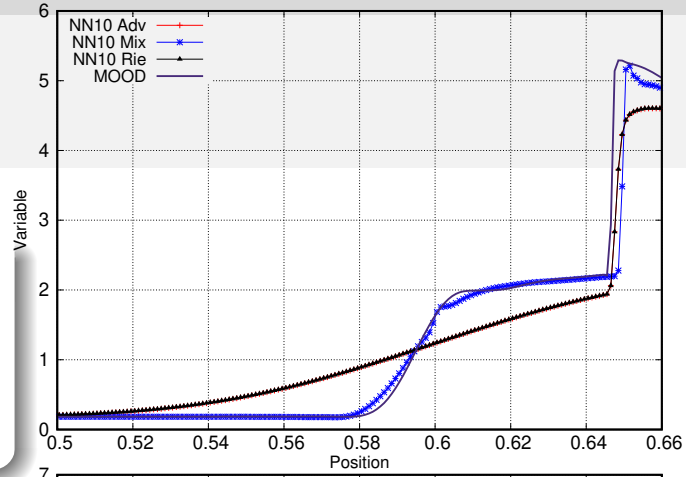



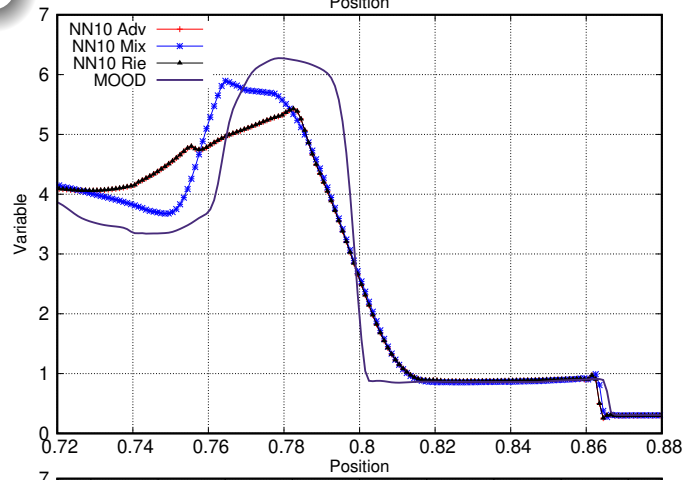
# Numerical experiences

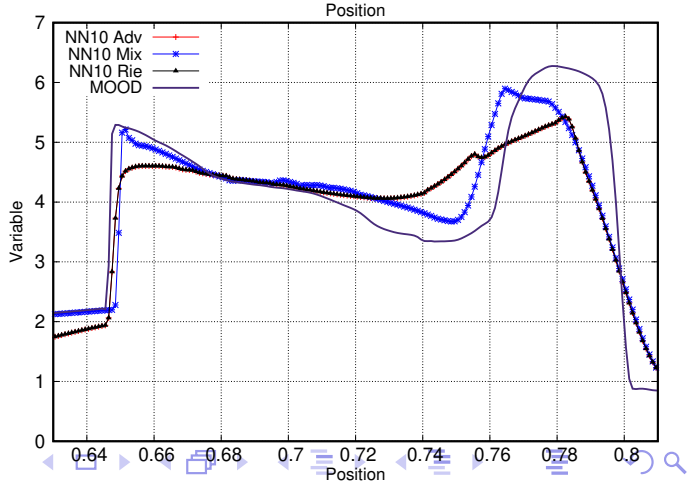
## Blastwave problem: interaction of waves

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution



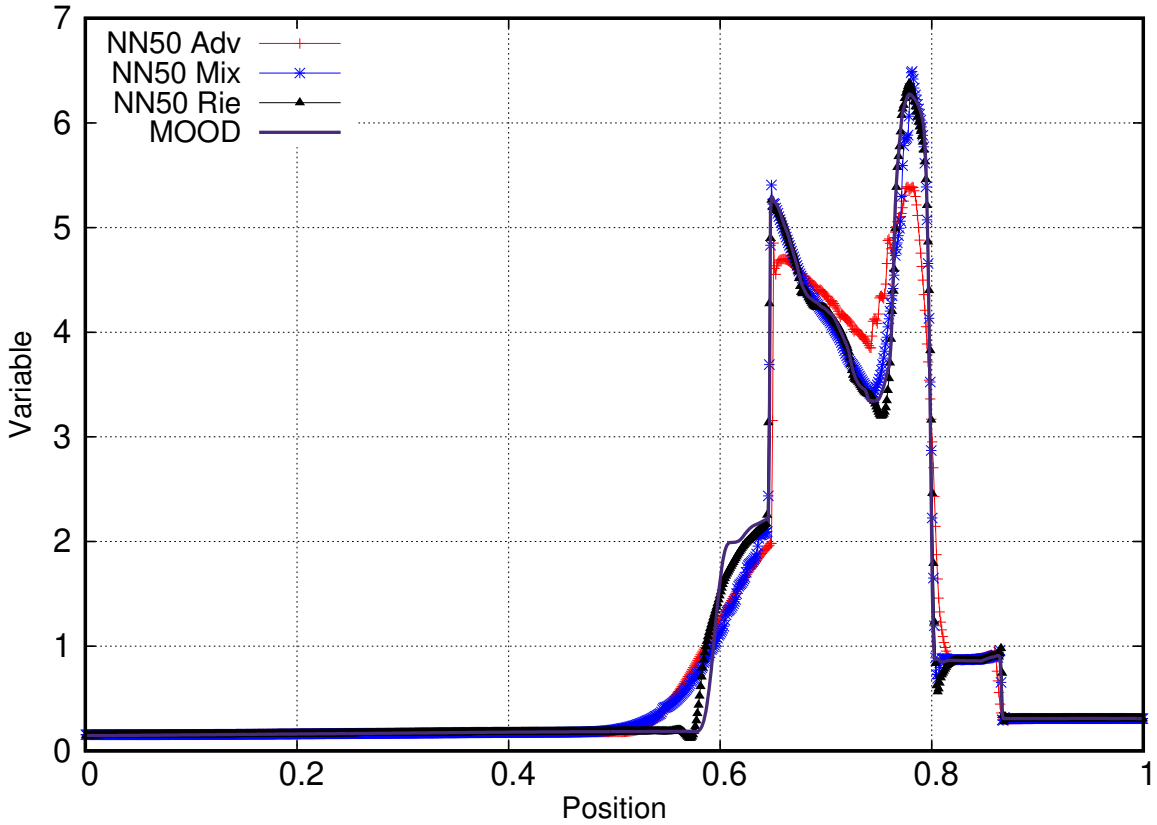



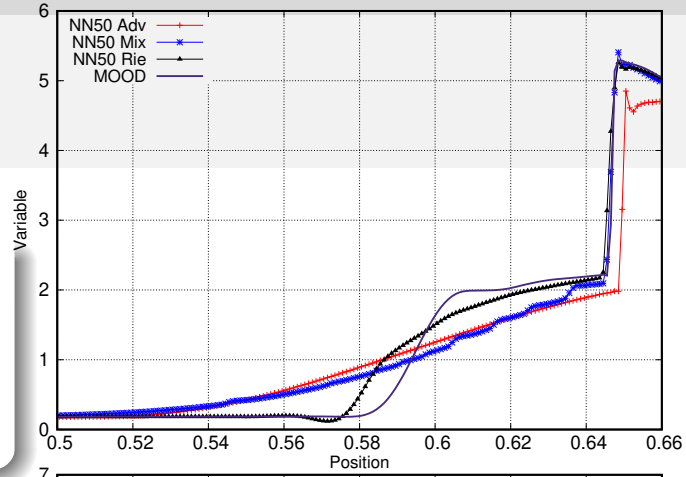



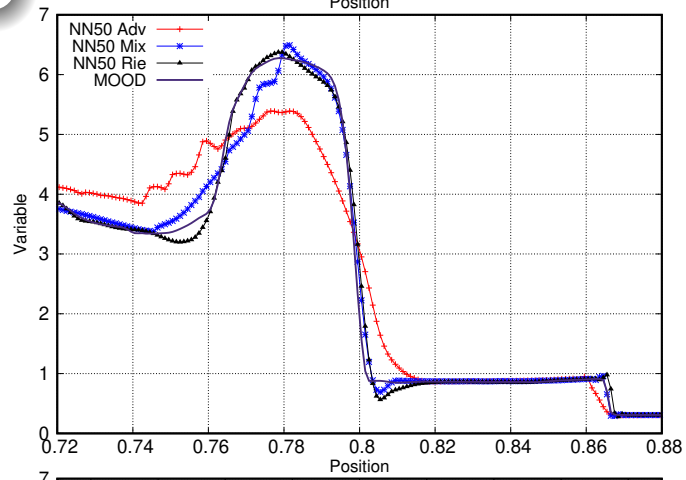
# Numerical experiences

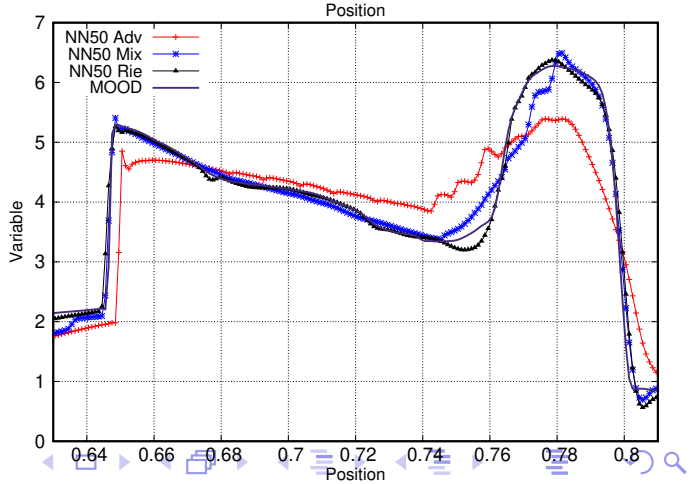
## Blastwave problem: interaction of waves

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution



# Numerical experiences

**Blastwave problem: interaction of waves**

$(\rho, u, p)_L = (1, 0, 1000)$, $(\rho, u, p)_M = (1, 0, 1000)$ and $(\rho, u, p)_R = (1, 0, 100)$, $T = 0.038$, 1000 uniform cells, reference solution

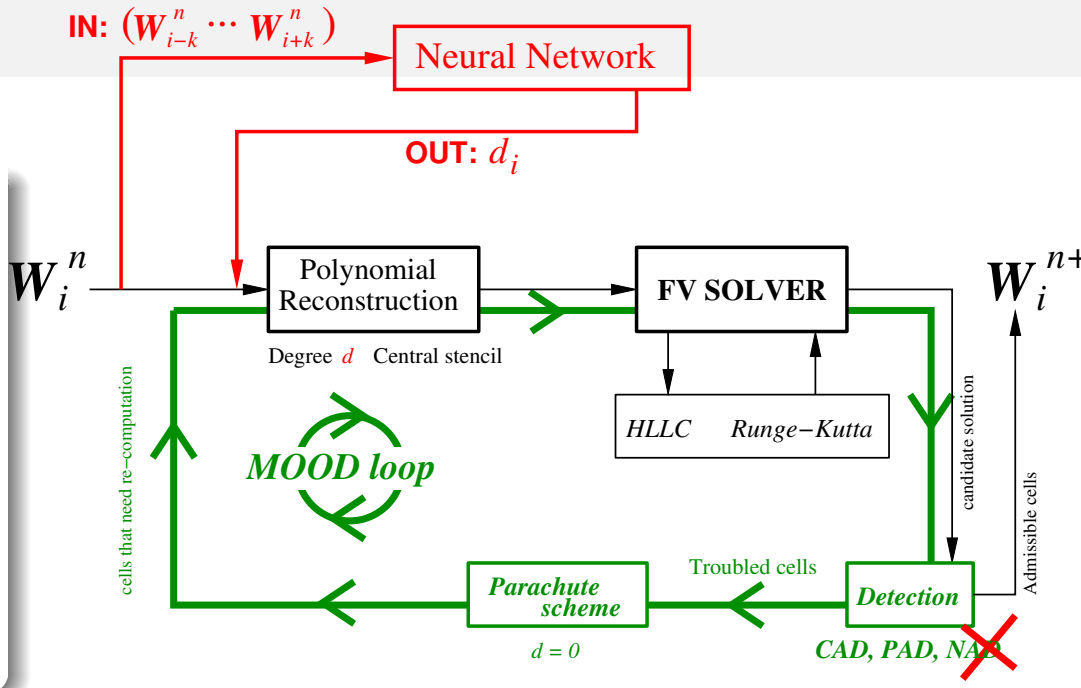
**Observations**

1- Difficulties to perform like MOOD
2- Architecture: large network (50) usually captures the interacting waves better
3- Training: advection profile does not perform well, Rie training with large NN behaves reasonnably well, strangely Mix training with small NN behaves better than large NN.

# Conclusions

## Neural Network needs

- to be trained to recognize "patterns" and classify situations
- architecture ← not clear?
- training dataset ← conpount wave?
- validation dataset ← unknown fcts
- verification dataset ← classical tests
- build once and *a priori* evaluated




## Test the use of a NN in conjonction with a FV High accurate MOOD FV scheme

- Trut: *a posteriori* MOOD loop used to determine $d_i = 0 \rightarrow d_{\max}$
- Given FV data $\boldsymbol{W}_{i\pm k}^n$ in stencil, NN *a priori* predicts which $d_i$ to use
- Tested on advection and Euler 1D equation up to order 4 on uniform grid to replace the NAD criteria
- *a posteriori* MOOD loop still needed for PAD and CAD
- Results are promissing but highly dependent on the NN (arch., training)

# Perspectives

## Neural Network

- Test different architectures $\mathrm{P}\to\mathrm{P}\to\mathrm{P}$, huge NNs ($P = 256$),
- Rule of a thumb to design an appropriate architecture?
- Enough to learn from profiles? Mandatory to have real problems? Exact solutions (then exact $d$)?

## Range of testing

- Order $4, 5, \ldots$, different cascades (some orders may be useless)
- 2D: train with all Riemann problems? What about physical instabilities?
- 3D, // efficiency, AMR
- Other system of PDEs: MHD, M1 model, Baer-Nunziato, etc.
- Implicit scheme?

# THANK YOU!

Bibliography

[1] A. Bourriaud, R. Loubère, R. Turpault, *a priori* Neural Networks vs *a posteriori* MOOD loop. A high accurate Finite Volume scheme testing bed. To be submitted in 09/2019
[2] D. Ray, J. Hesthaven, An artificial neural network as a troubled-cell indicator, JCP 367, 2018, Pages 166-191
[3] N. Discacciati, J.S. Hesthaven, D. Ray, Controlling oscillations in high-order Discontinuous Galerkin schemes using artificial viscosity tuned by neural networks, preprint 2019