IX International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2021
E. Oñate, M. Papadrakakis and B. Schrefler (Eds)

A DEEP LEARNING APPROACH FOR CURVATURE PREDICTION IN ALGEBRAIC VOLUME OF FLUID METHODS

ZACHARIAS KRAUS*, JONAS FRIEDRICH*, FELIX KÖHLER*, MICHAEL SCHÄFER*

*Institute of Numerical Methods in Mechanical Engineering
Technical University of Darmstadt
Dolivostraße 15, 64293 Darmstadt, Germany
e-mail: **kraus@dyn.tu-darmstadt.de**, **koehler@fnb.tu-darmstadt.de**,
web page: `www.fnb.tu-darmstadt.de`

Key words: Deep Learning, Machine Learning, Volume of Fluid Method

Abstract. The accurate prediction of the curvature of fluid-fluid interfaces is crucial for appropriately modeling the surface forces when computing two-phase flows with immiscible fluids. The volume of fluid (VOF) method is often used for these computations to specify the different fluids and the interface by the so-called volume fraction field. In this study, a deep artificial neural network is trained to predict the interface curvature from the volume fraction values. This approach is investigated within an algebraic VOF framework. A rudimentary interface resharpening algorithm is introduced for the input stencils to enhance the accuracy and robustness when the interface can not be captured entirely sharp. The performance of different neural network architectures is evaluated by generic test data and the computation of two oscillating droplet flow configurations.

1 Introduction

The volume of fluid (VOF) method is often used when calculating two-phase flows with immiscible fluids. The so-called volume fraction field determines the different fluids and their interface. Based on the original VOF method to track free boundaries in 1981 by Hirt et al. [7] two types of VOF methods for the prediction of incompressible, immiscible multiphase flows emerged: Geometric and algebraic methods.

Geometric VOF methods reconstruct the interface explicitly from the volume fraction field. The accuracy of geometric VOF methods and the sharpness of the interface are their strengths, but the computational cost, on the other hand, is their weakness [3]. In contrast algebraic VOF methods solve an additional transport equation for the color function; hence no explicit interface reconstruction is applied. In order to keep the interface as sharp

as possible, special discretization techniques in space and time are necessary [4]. Algebraic VOF methods have been modified and improved to compensate for the drawbacks of the original algorithm. For example, the numerical diffusion is reduced by applying a high-resolution differencing scheme, like the M-CICSAM [16], to discretize the transport equation for the color function. Although sharp in theory, the VOF approach produces a non-sharp interface that stretches over a few computational cells in many challenging flow computations [15].

For flow configurations where the surface tension, acting as a surface force, can not be neglected, Brackbill et al. [2] introduced the continuum surface force (CSF) model. The surface tension is modeled as a body force and is imposed as a continuous, three dimensional effect across an interface. Liovic et al. [8] concluded that the accurate modeling of the surface force depends primarily on the accurate prediction of the interface curvature. Different appropriate methods have been developed over the last decades, which rely on either the direct derivation from the volume fraction field or on discrete differential operators applied to an explicit description of the location of the interface [11].

This study focuses on curvature estimation methods which are based on the implicit representation of the interface and investigates a new machine learning-based approach. Qi et al. [13] proposed a new approach that applies a neural network to find a relation between volume fractions and the corresponding curvature. Stencils of volume fraction values serve as the input of the neural network and the nondimensionalized curvature is obtained as the output. The neural network has to be trained on a data set before being used within a flow computation. Patel et al. [10] further developed and investigated this method and evaluated its performance within a flow solver.

Here, the neural network approach is investigated within an algebraic VOF framework instead of a geometric one in previous works. Different neural network architectures, including deep neural networks, are assessed. An additional pre-processing step is introduced to enhance the accuracy and robustness of this method within flow computations. Section 2 briefly describes the computational methods and the applied VOF framework. Section 3 outlines the introduced deep learning framework and the results are discussed in Section 4.

2 COMPUTATIONAL TWO-PHASE FLOW AND VOF METHOD

The two-phase flow configurations considered in this work are described with the one-fluid formulation of the incompressible Navier-Stokes equations (NSE) [12] as

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_i)}{\partial x_i} = 0, \tag{1}$$

$$\frac{\partial (\rho u_i)}{\partial t} + \frac{\partial (\rho u_i u_j)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] - \frac{\partial p}{\partial x_i} + \rho g_i + f_{\sigma,i}, \tag{2}$$

with the density ρ, the dynamic viscosity μ, the velocity vector u_i and the pressure p. The additional source term $f_{\sigma,i}$ is introduced into the momentum equation (2) to account for

the surface tension force on the fluid-fluid interface. The material properties density and viscosity change locally according to the prevailing fluid and are blended between both fluids over the interface. The VOF method is used to distinguish the different phases and identify the corresponding fluid in each computational point. The different fluids are represented by the volume fraction α which takes values from zero to one, where zero specifies the first and one the second fluid. The volume fraction is between zero and one for the fluid-fluid interface. To capture the interface and advect the volume fraction a transport equation for the volume fraction field is introduced:

$$\frac{\partial \alpha}{\partial t} + u_i \frac{\partial \alpha}{\partial x_i} = 0. \tag{3}$$

To guarantee a bounded solution while maintaining the sharpness of the interface, the high-resolution schemes M-CICSAM [16] is used to calculate the face values of the volume fractions. Although the algebraic VOF approach, in theory, produces a sharp interface, a non-sharp interface can occur in the application [15], especially for challenging surface tension dominated flows.

The surface tension term is realized with the continuum surface force (CSF) model by Brackbill et al. [2]. The surface tension is represented by $f_{\sigma,i} = \sigma\kappa\hat{n}_i$ with the constant surface tension coefficient σ, the curvature of the interface κ and the unit normal vector to the interface $\hat{n}_i$. By defining the normal vector in the form of the volume fraction field and appropriately discretizing the term, the surface tension term can be described as

$$f_{\sigma,i} = \sigma\kappa\frac{\partial \alpha}{\partial x_i}. \tag{4}$$

Since the volume fraction gradient vanishes outside the vicinity of the interface, the surface tension can be implemented as a volume force. Apparently, the accurate prediction of the interface curvature is crucial for the appropriate modeling of the surface tension force. Many different methods for calculating the curvature have been proposed over the years.

One of the most commonly used approaches for curvature prediction is the height function (HF) technique. In contrast to other curvature methods, the HF method is understood as an integral method that relies on geometric means [1]. A local height function is constructed around an interface cell from volume fractions and differentiated to obtain the curvature. In 2D, a 7×3 volume fraction stencil in the direction of the largest component of the normal in the interface cell is used. The volume fraction values are summed up over the three columns. The curvature is then computed with the corresponding derivatives [1].

The second curvature prediction algorithm that serves for comparison in the following investigation is a coupled VOF level-set (CVOFLS) method based on the work of Park et al. [9]. The CVOFLS method combines the advantages of the VOF and the level-set (LS) method. A strength of level-set approaches compared to the VOF algorithms is the smoothly varying color function across the interface; hence the interface normal

vector and the curvature are easily obtained [5]. In each curvature computation, the level-set function gets temporarily constructed from the volume fractions, but without an additional advection of the level-set function. Therefore, the information of the level-set function has no explicit impact on the VOF advection algorithm. While this approach provides very accurate results, it is computationally very expensive.

The two-phase flow computation framework and the curvature prediction methods described above are implemented in an in-house finite volume based flow solver. The equations are solved numerically on block-structured grids.

3 DEEP LEARNING FRAMEWORK FOR CURVATURE PREDICTION

A recent curvature prediction approach for VOF methods proposed by Qi et al. [13] uses an artificial feed-forward neural network, or just neural network (NN). The neural network is trained on generically generated data and predicts the interface curvature based on stencils of volume fraction values.

Figure 1 outlines this procedure performed for each computational point of the grid along the interface individually, as in a conventional approach. A stencil of volume fraction values around each computational point is used as input features for the neural network. The individual volume fractions are always arranged in a vector of the same order before being passed to the neural network. The corresponding curvature value is obtained by the neural network's output, which is nondimensionalized by the cell height Δ of the computational grid. The neural network has to be trained and tested on a broad, well-distributed data set before being used in a flow simulation. During training, the weights and biases of the neural network are determined and optimized to predict the curvature accurately. The neural network is called deep when it consists of at least two hidden layers between the input and output layer. The trained neural network with its corresponding weights and biases is implemented into the flow solver afterwards. A comprehensive introduction on neural networks and deep learning is given by Goodfellow et al. [6].

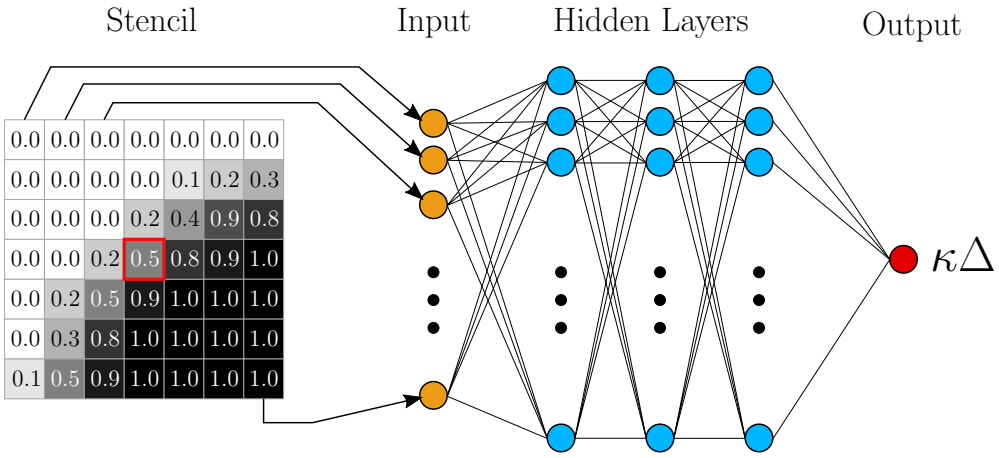


Figure 1: Neural network-based curvature prediction from volume fraction stencil

The following study investigates this neural network-based curvature prediction approach for an algebraic VOF method. Following the work of Qi et al. [13] and Patel et al. [10], who used geometric VOF methods, the approach is adapted and extended for the underlying computational fluid dynamics (CFD) framework to increase its accuracy and robustness. Stencils of 7×7 volume fraction values are used here. The overall procedure is built on 1) a data generation process, 2) a newly introduced resharpening algorithm for the input stencils and 3) the training and testing of different neural network architectures and the evaluation within flow computations.

The data generation process is similar to the one proposed by Patel et al. [10]. However, it is extended to include a broader range of possible volume fraction stencil configurations and to account for not perfectly sharp interfaces, as described in detail in Section 3.1.

Additionally, a rudimentary interface resharpening algorithm for the input stencils is introduced since the fluid interface can not always be conserved sharp in computations with the algebraic VOF method. This should increase the consistency of the neural network-based curvature prediction between the training on generic data and the usage within an actual flow simulation. Therefore, as outlined in Section 3.2, the algorithm should enhance the accuracy and robustness of this approach. Furthermore, the volume fraction stencils are rotated such that the normal vector of the interface always points to the same quadrant of the stencil before used as inputs.

The central part of this study in Section 4 investigates the influence of different neural network architectures with varying numbers of layers and nodes as well as the influence of the proposed resharpening algorithm. In contrast to previous work, the trained neural networks are evaluated not only by their performance on the generic data but also by their performance in flow computations. Two oscillating droplet configurations are considered and are described in Section 3.3. Since the overall results from flow computations are included in determining the final neural network layout, a more consistent solution for actual flow computations can be found.

The approach is investigated within the algebraic VOF framework with a one-fluid formulation as described in Section 2. The trained neural network is implemented directly in the in-house finite volume flow solver for the fluid simulations. The data generation is conducted in a Python framework and Google's Tensorflow is used for training and testing the neural networks. The study is limited to the two-dimensional space to focus on particular aspects of the approach.

3.1 Data Generation

A comprehensive data set of pairs of volume fraction stencils and corresponding nondimensionalized curvatures of arbitrary fluid-fluid interface configurations is required to train and test the neural networks. The data set should cover a wide range of different interface configurations to represent as many situations of a 'real' CFD simulation as possible. The algorithm used in this study to generate such a data set is loosely based on the work of Patel et al. [10] and is outlined in the following.

The algorithm is based on ellipse geometries and generates one data pair for each execution. The geometrical ellipse parameters and the curvature are always randomly chosen at the beginning of the algorithm. Therefore, the obtained data is uniformly distributed over the curvature when a large data set is created and the neural network is trained equally well on different curvature values. Additionally, the volume fraction stencils are slightly 'smeared out' by applying a kernel since the interface is often not captured completely sharp over the entire computation in the used algebraic VOF framework.

The data is generated on a quadratic domain with the length $L = 1$ with the origin in the bottom left corner. An equidistant square grid discretizes the domain with cell height $\Delta = 10^{-3}$. The created ellipse is described by the standard equation $x^2/a^2 + y^2/b^2 = 1$ with the width $2a$ and the height $2b$ and an aspect ration e which is here defined as $e = b/a$. The center of the ellipse (x_0, y_0) is placed in the middle of the domain initially. The algorithm is carried out in a Python environment and can be described as:

1. Randomly choose a curvature in the range of $\kappa_{\min}\Delta = 10^{-5}$ and $\kappa_{\max}\Delta = 0.4$.
2. Randomly choose an aspect ratio e and half the height b of an ellipse within
$$-\frac{L\Delta}{\kappa e} \leq b \leq \frac{-e^2 L\Delta}{\kappa} \tag{5}$$
to guarantee the existence of the chosen curvature on the ellipse surface.
3. Determine the specific location (x, y) on the ellipse surface with the chosen curvature $\kappa\Delta$ according to:
$$x = \pm\sqrt{\frac{\left(L\Delta\frac{-e^2b^2}{\kappa}\right)^{\frac{2}{3}} - b^2}{e^4 - e^2}} + x_0, \qquad y = \pm\sqrt{b^2 - e^2\left(x - x_0\right)^2} + y_0. \tag{6}$$
4. Rotate the ellipse by a random angle θ and translate it randomly within up to one cell width in x- and y-direction.
5. Find the cell in which the specific point (x, y) (translated and rotated with the ellipse) is located.
6. Shift the ellipse randomly between $-2L\Delta$ and $2L\Delta$ along the interface normal.
7. Calculate the volume fraction values of a 7×7 stencil by numerical integration.
8. Create two 'smeared' interface volume fraction stencils by folding with the following coefficients
$$Q = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{7}$$
once and twice. Blend the original stencil with the 'smeared' ones by interpolating between all three by a random number between 0.5 and 1.5.
9. Keep created stencil and curvature or inverted both to represent a swapped fluid-fluid configuration by a 50% chance.

3.2 Interface Resharpening Algorithm for Input Stencils

To ensure a low implementation and computation effort, the resharpening is performed for the x- and y-coordinates individually. Consequently, the original stencil is transformed into two stencils with adjusted volume fraction values. Since the adjusted volume fraction can not be computed for edge cells, these cells are eliminated and the stencil size is reduced by two in each direction during this procedure. The 7×7 stencil is transformed into two 5×5 stencils. However, the total number of input elements to describe the interface changes only slightly from 49 to 50. The algorithm is implemented for the training and also in the flow solver as follows:

1. Initialize two new output stencils with reduced size by two, one stencil for the x- and y-coordinates, respectively.
2. Find the cells in the original stencil where $\alpha > 0.5$ and their left or right (x) and upper or lower (y) neighbour cells containing a volume fraction $\alpha < 0.5$, respectively.
3. Determine the location of the 'sharp' interface ($\alpha = 0.5$) in the cell pairs by linear interpolation. Proportionally calculate an adjusted volume fraction for the cell containing the interface. Assign the volume fraction according to the present fluid to the other cell.
4. If one cell is to be assigned two values, take the mean of both new values.
5. Set the volume fraction of the remaining cells to $\alpha = 1$ or $\alpha = 0$ according to the present fluid.

3.3 Test Cases: Oscillating Droplets

An ethanol droplet is placed in a zero-gravity air environment with two different initial distortions: an elliptical and a square droplet. When starting the computation from a zero velocity field, both droplets contract and begin to oscillate due to the surface tension, as shown in Fig. 2. Viscous forces damp the oscillation until a circular droplet in steady state is reached. These challenging test cases are selected due to their surface tension dominated nature. They are described in detail by Strubelj et al. [14].

Both droplets are placed in the center of a square domain of $L = 75\,\mathrm{mm}$ length. The average radius of the elliptic drop is $r = 21.84\,\mathrm{mm}$ and of the square droplet $r = 22.56\,\mathrm{mm}$. The elliptic droplet is initially vertically distorted by 25 % and the second initial condition is set to a square of the length $l = 40\,\mathrm{mm}$. The material properties are for both cases the same: $\rho_1 = 787.88\,\mathrm{kg\,m^{-3}}$, $\mu_1 = 2.4 \cdot 10^{-2}\,\mathrm{Pa\,s}$, $\rho_2 = 1.1768\,\mathrm{kg\,m^{-3}}$ and $\mu_2 = 2 \cdot 10^{-3}\,\mathrm{Pa\,s}$, where the index 1 indicates the droplets and 2 the surrounding air. The surface tension between the fluids is $\sigma = 0.023\,61\,\mathrm{N\,m^{-1}}$.

The performance of a curvature prediction method is evaluated by the comparison to the analytic oscillation period, which is $\tau_0 = 1.513\,\mathrm{s}$ for the ellipse and $\tau_0 = 0.502\,\mathrm{s}$ for the square, following the analytical derivation of Strubelj et al. [14]. The computations are performed on equidistant square grids and the north point movement is compared for different curvature prediction methods.

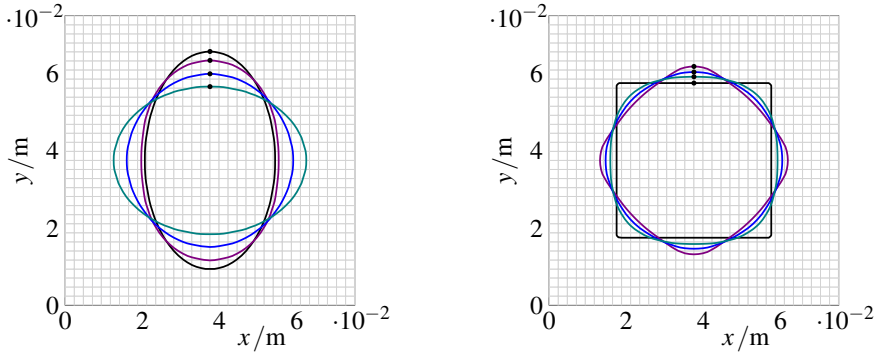


Figure 2: Oscillating droplet with ellipse (left) and square (right) initial distortion.

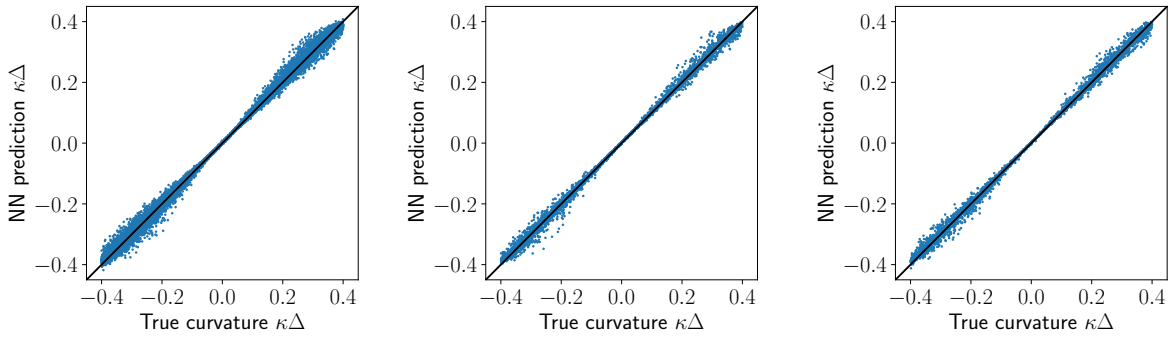
4 RESULTS AND DISCUSSION

The framework presented in Section 3 is investigated concerning different neural network layouts and applying the proposed resharpening algorithm. The best setup for neural network-based curvature prediction in the algebraic VOF method is found in terms of accuracy, general applicability and robustness directly within flow computations. Fifteen different neural network architectures with up to five hidden layers and 200 nodes each layer maximum are evaluated with and without resharpening algorithm for the input stencils. The rectified linear unit (ReLU) is used as the activation function in the hidden layers. The mean square error (MSE) of the nondimensionalized curvature is used as the loss function while the curvature is scaled to $[-1, 1]$ in the output layer.

The neural networks are trained and tested on a data set with 10^6 data points generated by the algorithm outlined in Section 3.1. The data set is divided into training, validation and testing data in proportions of 70%, 15% and 15%, respectively. Batch training is performed with a batch size of 128. An early stopping criterion is employed, which stops the training when the validation error does not reduce any more for several epochs of training. For each neural network architecture, 18 individual networks are trained and tested separately to take the random nature of neural networks into account and obtain reproducible results. For all 18 networks, the data set splitting and the initialization of weights and biases is performed randomly.

The training and testing of the different neural network layouts on the data set yield very similar results. Only minor differences in the MSE can be recognized, regardless of whether the resharpening algorithm is applied or not. The different MSEs of the worst and best architecture lie within 8% of the overall average MSE. Furthermore, the training, validation, and test error are very close to each other, indicating a large enough data set. The solution of all 18 individually trained networks differ only slightly.

The results are shown representatively by the neural network with three hidden layers with 200 nodes each and the resharpening algorithm applied, which will be denoted as '200-200-200' according to its architecture. The MSE for the training, validation and testing is $1.0235 \cdot 10^{-3}$, $1.0251 \cdot 10^{-3}$ and $1.0241 \cdot 10^{-3}$, respectively, referring to the



(a) Training (b) Validation (c) Test

Figure 3: Predicted nondimensiolized curvatures over their actual value for one '200-200-200' neural network with resharpening algorithm, every 40th data point.

normalized curvature. Figure 3 shows the curvature predicted by one single '200-200-200' neural network over the actual curvature for the training, validation and test data. Only every 40th data point is shown for clarity. The distribution looks similar for all three parts and the predicted curvature matches the actual value quite well for most data points. The predicted curvature shows a larger deviation for values with a larger magnitude. This effect might be due to the more complex interface geometries on the stencil for such large curvatures.

Although all different neural network architectures yield very similar results on the generic data set regardless of the resharpening algorithm, the performance within the flow computation of both oscillating droplets differ significantly. The oscillating droplet configurations are calculate with all 18 different neural networks for one architecture on a 128×128 grid. The performance is evaluated by different quantities of interest after one period, averaged over all networks.

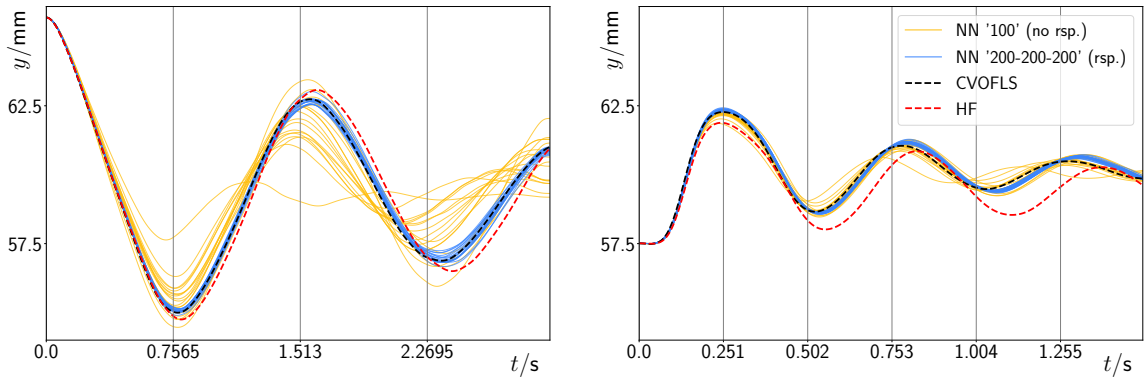
Table 1 provides an overview of the performance for the entire study's most relevant neural network architectures. Here, E_{per} is the mean absolute error between the calculated period and the analytical period and $\hat{\sigma}_{per}$ is the standard deviation of the computed period. Both quantities are normalized by the analytical period. The north point position, representing the amplitude, is compared to the computational solution with the CVOFLS model. Although this reference is not an entirely correct or an analytical solution, the results with the CVOFLS model seem to be quite accurate, as shown later. Moreover, the amplitude evaluation is essential in this study since significant differences occur for different configurations. Accordingly, E_{amp} is the mean absolute error of the north point position after one period compared to the CVOFLS solution normalized by the same value. $\hat{\sigma}_{amp}$ is the corresponding normalized standard deviation. A performance measure P is defined when applying the resharpening algorithm as the mean of all evaluated quantities, each normalized by the maximum value of all shown neural network architectures. Finally, all quantities are multiplied by 100.

Overall, the performance of the neural networks while applying the resharpening algo-

rithm is closer to the analytical values and the CVOFLS amplitude than the ones without the algorithm. Although the computed mean period without resharpening algorithm is better for some architectures, the standard deviations are significantly larger, which is due to inconsistent behavior of the individually trained networks. Without the resharpening algorithm, the error seems to increase for most quantities which might be an overfitting related effect. The different neural networks architectures' performance with the resharpening algorithm differs less from another and is relatively close to the references. Good results are already achieved by the neural network with only one hidden layer and 200 nodes. However, the three hidden layer networks with at least 100 nodes each generally perform well and might be favourable also in an even broader study with varying flow configurations. The '200-200-200' network is the best by the defined performance measurement and is investigated further.

The above-concluded results are supported by comparing the north point movement in Fig. 4. The presumably best neural network architecture without resharpening algorithm '100' varies for its different networks significantly. The computation with the '200-200-200' network and resharpening algorithm is very consistent and matches the CVOFLS results well. The analytical period is very well approximated with the '200-200-200' and CVOFLS calculation while the HF computations differs slightly. This is also quantitatively shown by the first period of the different methods in Table 2. The computations with the neural network-based curvature prediction achieve consistent and good results also on different grids, as shown in Table 2 for the oscillating droplets on a 64×64 and a 256×256 grid.

The computational effort for the neural network-based curvature prediction is similar to the one with the HF method but significantly lower than for the CVOFLS. The average time for computing the iteration of one time step with the applied CFD framework is 4.22 s for the CVOFLS, 0.89 s for the HF and 1.08 s for the '200-200-200' NN with resharpening algorithm.



(a) Elliptic droplet (b) Square droplet

Figure 4: North point movement of the oscillating droplets computed on a 128×128 grid including all 18 neural networks for each architecture and resharpening algorithm (rsp.)

Table 1: Performance of different neural network architectures in flow computations of the elliptic and square droplet on a 128×128 grid with and without resharpening algorithm. Coloring by the rank of the value in each evaluated quantity while using the resharpening algorithm. Smaller values are better, likewise for the performance measure P.

	elliptic droplet				square droplet				
Hidden Layers	E_{per}	$\hat{\sigma}_{per}$	E_{amp}	$\hat{\sigma}_{amp}$	E_{per}	$\hat{\sigma}_{per}$	E_{amp}	$\hat{\sigma}_{amp}$	P
with resharpening									
25	4.71	0.82	6.15	5.33	6.97	0.43	10.13	2.80	89.5
100	4.29	0.91	3.69	2.55	6.76	0.38	8.23	2.45	73.2
200	4.30	0.68	2.43	2.60	6.67	0.33	6.92	1.65	62.7
25-25	4.76	1.01	4.48	4.58	7.06	0.31	8.43	4.00	86.3
100-100	4.63	0.91	3.42	3.66	6.88	0.37	7.70	2.25	74.9
200-200	4.42	0.86	1.73	2.38	6.66	0.38	5.88	2.32	65.1
25-25-25	4.68	0.72	4.78	4.61	7.00	0.44	8.91	3.10	83.4
100-100-100	4.39	0.49	2.51	3.06	6.80	0.37	6.22	2.14	63.5
200-200-200	4.48	0.59	1.69	2.13	6.78	0.43	5.41	2.37	61.9
200-100-25	4.25	0.67	1.91	2.89	6.35	0.64	4.60	1.78	65.1
25-100-200	4.51	0.63	3.13	2.32	7.02	0.34	7.67	1.75	65.4
5*200	4.41	0.92	2.16	2.22	7.06	0.50	3.76	3.22	69.5
without resharpening									
100	3.09	4.63	16.57	17.47	5.50	2.17	8.78	13.42	
200	6.76	2.95	23.96	16.25	4.82	1.15	9.00	11.12	
200-200	11.86	7.16	52.22	8.38	1.59	1.28	20.72	8.52	
200-200-200	13.45	8.19	53.33	14.10	1.91	1.86	28.64	10.86	

Table 2: Computed period of elliptic and square droplet for different grid resolutions

	elliptic droplet (in s)			square droplet (in s)		
Method	64×64	128×128	256×256	64×64	128×128	256×256
analytical period	1.513	1.513	1.513	0.502	0.502	0.502
NN '200-200-200'	1.579	1.580	1.572	0.553	0.536	0.534
CVOFLS	1.582	1.572	1.578	0.520	0.523	0.527
HF	1.609	1.613	1.618	0.564	0.554	0.548

5 CONCLUSION

In conclusion, a neural network can be trained to accurately predict the interface curvature within an algebraic VOF framework. Applied in a multiphase flow computation good results are obtained even for challenging flow configurations. Introduction a pre-processing step for the input values like a resharpening algorithm can enhance the accuracy and robustness of this approach.

References

[1] S. Afkhami and M. Bussmann. Height functions for applying contact angles to 2d vof simulations. *International Journal for Numerical Methods in Fluids*, 57(4):453–472, 2008.

[2] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modeling surface tension. *Journal of Computational Physics*, 100:335 – 354, 1992.

[3] F. Denner, D. R. van der Heul, G. T. Oud, M. M. Villar, A. d. S. Neto, and B. G. M. van Wachem. Comparative study of mass-conserving interface capturing frameworks for two-phase flows with surface tension. *International Journal of Multiphase Flow*, 141:37 – 47, 2014.

[4] F. Denner and B. G. M. van Wachem. Compressive vof method with skewness correction to capture sharp interfaces on arbitrary meshes. *Journal of Computational Physics*, 279:127 – 144, 2014.

[5] D. Gerlach, G. Tomar, G. Biswas, and F. Durst. Comparison of volume-of-fluid methods for surface tension-dominant two-phase flows. *International Journal of Heat and Mass Transfer*, 49:740 – 754, 2006.

[6] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[7] C. Hirt and B. Nichols. Volume of fluid (vof) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39:201 – 225, 1981.

[8] P. Liovic, M. M. Francois, M. Rudman, and R. Manasseh. Efficient simulation of surface tension-dominated flows through enhanced interface geometry interrogation. *Journal of Computational Physics*, 229:7520 – 7544, 2010.

[9] I. R. Park, K. S. Kim, J. Kim, and S. H. Van. A volume-of-fluid method for incompressible free surface flows. *International Journal for Numerical Methods in Fluids*, 61:1331 – 1362, 2009.

[10] H. V. Patel, A. Panda, J. A. M. Kuipers, and E. A. J. F. Peters. Computing interface curvature from volume fractions: A machine learning approach. *Computers & Fluids*, 193:104263, 2019.

[11] S. Popinet. Numerical models of surface tension. *Journal of Computational Physics*, 229:7520 – 7544, 2018.

[12] A. Prosperetti and G. Tryggvason. *Computational Methods for Multiphase Flow*. Cambridge University Press, Cambridge, 2009.

[13] Y. Qi, J. Lu, R. Scardovelli, S. Zaleski, and G. Tryggvason. Computing curvature for volume of fluid methods using machine learning. *Journal of Computational Physics*, 377:155 – 161, 2018.

[14] L. Štrubelj, I. Tiselj, and B. Mavko. Simulations of free surface flows with implementation of surface tension and interface sharpening in the two-fluid model. *International Journal of Heat and Fluid Flow*, 30(4):741–750, 2009.

[15] K. J. Vachaparambil and K. Einarsrud. Comparison of surface tension models for the volume of fluid method. *Processes*, 7:542, 08 2019.

[16] T. Waclawczyk and T. Koronowicz. Remarks on prediction of wave drag using vof method with interface capturing approach. *Arch Civ Mech Eng*, 8, 2008.