Accelerating Simulation of Two-Phase Flows with Neural PDE Surrogates

Yoeri Poels [* 1 2] **Koen Minartz** [* 2] **Harshit Bansal** [* 3] **Vlado Menkovski** [2]

Abstract

Simulation is a powerful tool to better understand physical systems, but generally requires computationally expensive numerical methods. Downstream applications of such simulations can become computationally infeasible if they require many forward solves, for example in the case of inverse design with many degrees of freedom. In this work, we investigate and extend neural PDE solvers as a tool to aid in scaling simulations for two-phase flow problems, and simulations of oil expulsion from a pore specifically. We extend existing numerical methods for this problem to a more complex setting involving varying geometries of the domain to generate a challenging dataset. Further, we investigate three prominent neural PDE solver methods, namely the UNet, DRN, and U-FNO, and extend them for characteristics of the oil-expulsion problem: (1) spatial conditioning on the geometry; (2) periodicity in the boundary; (3) approximate mass conservation. We scale all methods and benchmark their speed-accuracy trade-off, evaluate qualitative properties, and perform an ablation study. We find that the investigated methods can accurately model the droplet dynamics with up to three orders of magnitude speed-up, that our extensions improve performance over the baselines, and that the introduced varying geometries constitute a significantly more challenging setting over the previously considered oil expulsion problem.

---

[*]Equal contribution [1]Swiss Plasma Center, École Polytechnique Fédérale de Lausanne [2]Data and Artificial Intelligence Cluster, Eindhoven University of Technology [3]Centre for Analysis, Scientific Computing, and Applications, Eindhoven University of Technology. Correspondence to: Yoeri Poels <yoeri.poels@epfl.ch>, Koen Minartz <k.minartz@tue.nl>, Harshit Bansal <h.bansal@tue.nl>, Vlado Menkovski <v.menkovski@tue.nl>.
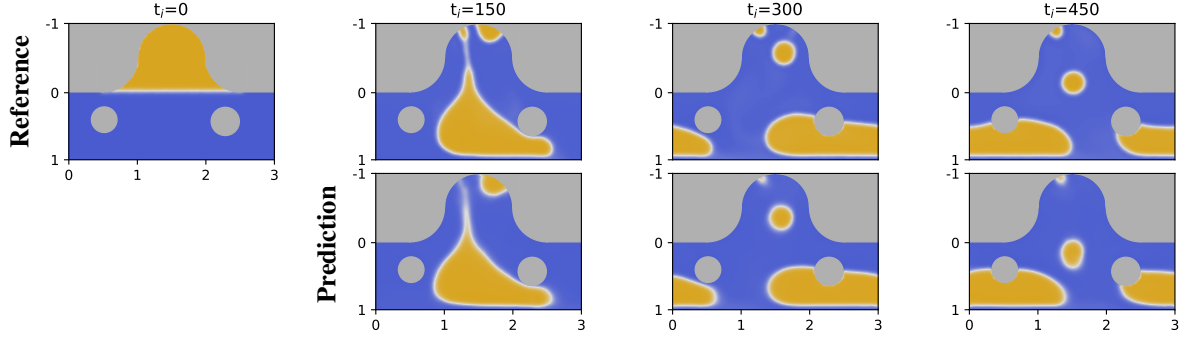
1. Introduction

Multiphysics phenomena emerge from the interplay of different physical aspects in a system. Such phenomena appear naturally in various domains, such as digital microfluidics and reservoir engineering, among many others (Pollack et al., 2002; Linga et al., 2019). In this work, we focus on a particular type of multiphysics process, namely multiphase electrohydrodynamics (Linga et al., 2019). Specifically, we investigate the industrially relevant application of oil expulsion from a pore. A two-phase flow of water and oil is modeled, where electrowetting is applied to an oil droplet to change the droplet's properties using an electrical charge. An illustration of this process can be found in Figure 1.

Some of the key challenges in this application are to control the spatiotemporal evolution of the dynamics, alongside the coalescence and break-down of the oil droplets. Generally we desire a specific dynamic response, which makes configuring operating conditions like electrical charge and rate of flow in the presence of varying geometries a challenging task. It is essentially an ill-posed inverse problem, requiring several forward solves to find a good solution. However, solving a high-fidelity forward model requires substantial computational resources, making it unpractical for multi-query optimization or real-time control.

Moreover, while extensive studies have been done to obtain stable, accurate, robust, and efficient high-fidelity numerical methods for simulating the aforementioned phenomena (Linga et al., 2019; 2020; Almasi et al., 2021), these methods were deployed in a setting without geometrical obstacles. To create a more challenging setting, we adapt these methods for obstacles in the domain. However, this extension leads to a further increase in computational cost, making the need for fast approximate methods significant.

To alleviate the issue of computationally expensive simulations, we propose the use of neural PDE solvers for fast surrogate modeling. We employ these models in an autoregressive manner by approximating the time-stepping operator of the system. While constructing approximate solutions of PDEs with neural networks has gained considerable traction (Azizzadenesheli et al., 2024), there is no comprehensive study of neural PDE surrogates applied to two-phase flows modeling oil extraction. In this work, we conduct such an investigation. In summary, our contribu-

Figure 1. The oil expulsion problem illustrated. *Top*: Numerical solver reference. *Bottom*: Neural surrogate model (UNet) prediction.

tions are as follows:

- We extend the numerical method for the oil expulsion problem as introduced in (Linga et al., 2019) to account for geometrical obstacles in the spatial domain;

- We generate a dataset of approximately 1000 simulations, alongside a simplified dataset without geometrical obstacles of approximately 500 simulations as a baseline setting, following Linga et al. (2019);

- We extend three prevalent neural PDE surrogate modeling methods (Stachenfeld et al., 2022; Gupta & Brandstetter, 2023; Wen et al., 2022) to account for the characteristics specific to the oil expulsion problem;

- We provide a benchmark of these models, evaluating their scaling in terms of accuracy and inference speed, and provide a sensitivity analysis comparing the surrogate models with the baseline high-fidelity solver.

2. Background and related work

2.1. Oil expulsion problem

The oil expulsion problem is modeled as a multiphase electrohydrodynamical system. Specifically, it is governed by the following system of coupled PDEs:

$$\partial_t(\rho(\phi)\mathbf{v}) + \nabla \cdot (\rho(\phi)\mathbf{v} \otimes \mathbf{v})$$
$$- \nabla \cdot [2\mu(\phi)\mathcal{D}\mathbf{v} + \mathbf{v} \otimes \rho'(\phi)M(\phi)\nabla g_\phi] + \nabla p \tag{1}$$
$$= -\phi\nabla g_\phi - \sum_i c_i \nabla g_{c_i},$$

$$\nabla \cdot \mathbf{v} = 0, \tag{2}$$
$$\partial_t \phi + \mathbf{v} \cdot \nabla \phi - \nabla \cdot (M(\phi)\nabla g_\phi) = 0, \tag{3}$$
$$\partial_t c_j + \mathbf{v} \cdot \nabla c_j - \nabla \cdot (K_j(\phi)c_j\nabla g_{c_j}) = 0, \tag{4}$$
$$\nabla \cdot (\varepsilon(\phi)\nabla V) = -\rho_e, \tag{5}$$

where t refers to time, $\mathbf{v}$ to the velocity field, p to the pressure field, ϕ to the phase field, ρ to the density (ρ_e the charge density), μ to the viscosity, c_j to the solute concentration of species $j \in \{H^\pm\}$, g_{c_j} to the electrochemical potential of a species, g_ϕ to the electrochemical potential of the phase field, K_j to the species' diffusivities, M to the phase field mobility, V to the electrostatic potential, ε to the permissivity, and $\mathcal{D}\mathbf{v}$ to the symmetric velocity gradient. Equation 1 describes the momentum balance, Equation 2 the continuity equation, Equation 3 the conservative evolution of the phase field, Equation 4 governs the transport of the concentration field of species, and Equation 5 describes Gauss's law.

To obtain high-fidelity solutions, we employ a linear operator splitting scheme (Linga et al., 2019) that results in three different subproblems - *(i)* hydrodynamic flow, *(ii)* phase field transport, and *(iii)* chemical transport under an electric field. We build on the implementation in BERNAISE (Linga et al., 2019), a FEniCS-based (Alnæs et al., 2015) high-fidelity solver, and adapt it to account for multiple geometrical obstacles and different practically relevant boundary conditions. For more details on the initial conditions, boundary conditions and the employed numerical method, we refer to Linga et al. (2019), to which we added boundaries around the obstacles. In particular, we have a non-zero Neumann condition for the electric potential, and a no-slip condition for the velocity field.

2.2. Neural PDE surrogate modeling

Constructing approximate solutions of PDEs with neural networks has seen a surge in interest over the past few years, with example applications in fluid dynamics (Li et al., 2021; Brandstetter et al., 2022; Gupta & Brandstetter, 2023), climate and weather modeling (Nguyen et al., 2023; Bonev et al., 2023; Gao et al., 2023), and thermonuclear fusion (Poels et al., 2023; Gopakumar et al., 2024), among others. However, compared to these settings, the simulation of two-phase flow problems with neural networks that we consider is relatively underexplored.

One relatively popular used approach for solving two-phase flow problems with neural networks is Physics-Informed

Neural Networks (PINNs) (Raissi et al., 2019). In this context, example applications of PINNs include the simulation of two-phase flows in porous media (Zhang et al., 2023; Hanna et al., 2022), simulation of the interface of two fluids (Zhu et al., 2023), simulating two-phase material microstructures (Ren & Lyu, 2024), and the simulation of the dynamics of bubbles (Lin et al., 2021b; Zhai et al., 2022; Buhendwa et al., 2021). Although PINNs inherently exploit domain knowledge by minimizing the residual of the PDE directly, they generally need to be optimized from scratch for every PDE instance and parameterization. As one of the main goals of our work is accelerating two-phase flow simulations, this makes them impractical for our setting.

Consequently, we focus on a different approach, namely autoregressive models and neural operators. We will refer to these methods as neural surrogate models. Compared to PINNs, neural surrogate models typically cannot exploit domain knowledge as easily, and rely more on the availability of data. However, once trained, they can be used to get approximate solutions to multiple parameterizations of the system of PDEs in question. There have been various applications of popular neural surrogate architectures to two-phase flow and multiphase flow problems, for example Deep-O-Nets (Lu et al., 2024; Lin et al., 2021a), Fourier Neural Operators (FNOs) (Hassan et al., 2023; Jiang et al., 2023), and convolutional architectures (Hassan et al., 2023). However, none of these works provide a comprehensive, independent and consistent comparison of neural PDE architectures in this setting.

3. Method

3.1. Model formulation

We first introduce some notation and describe the general formulation along which we design the surrogate models. We denote the state of the system at time t by x^t. Further, a sequence of k states, spanning from sequence element t up to but not including $t + k$ is denoted as $x^{t:t+k}$. Regarding the autoregressive approach, rather than using a model to predict a single timestep at a time, we formulate the model f_θ to operate on a *bundle* of k timesteps for both its input and output:

$$\hat{x}^{t:t+k} = f_\theta(x^{t-k:t}). \qquad (6)$$

Notably, operating on the bundle level as opposed to single timesteps has been shown to improve performance, both in terms of computational efficiency as well as accuracy (Brandstetter et al., 2022).

3.2. Model designs

We structure the model design space along the Encode-Process-Decode framework (Sanchez-Gonzalez et al., 2020). Here, the task of the encoder is to map the input bundle $x^{t-k:t}$ to an abstract embedding h^t, representing the entire bundle $x^{t-k:t}$. The processor then operates on h^t to produce an output embedding o^t. Finally, the decoder maps o^t to a prediction for the next bundle $\hat{x}^{t:t+k}$. The architectural choices for the encoder, processor, and decoder components that we consider in this work are shortly described below.

Encoder. As encoder we consider a simple convolutional neural network with a 1-by-1 kernel. Specifically, the different timepoints in the bundle are first flattened and taken as channels for the convolutional layer. Then, two convolutional layers are applied that operate on single grid points.

Processor. For the processor module, we consider three popular neural PDE surrogate modeling architectures; schematic overviews can be found in Figure 2. The first architecture is the Dilated ResNet (DRN) (Stachenfeld et al., 2022), illustrated in Figure 2a. The DRN is a convolutional model which effectively integrates information at different spatial scales by varying the dilation of subsequent convolutional layers: with larger dilations, long-range interactions can be modeled in a parameter-efficient manner.

The second architecture is the U-FNO (Wen et al., 2022), which is based on the Fourier Neural Operator (FNO) (Li et al., 2021); see Figure 2b. The FNO primarily relies on transforming the spatial signal to a frequency representation using the Fast Fourier Transform (FFT), and subsequently multiplying the spectral coefficients with a learned weight matrix R. Notably, this operation is equivalent to a global convolution, enabling the FNO to model long-range interactions. The U-FNO adds another parallel UNet branch, enabling it to better model high-frequency information, which is prominent in our setting near droplet boundaries. We either use only U-FNO layers or alternate them with regular FNO layers, as proposed in Wen et al. (2022).

The third architecture we consider is the UNet (Ronneberger et al., 2015), see Figure 2c. The UNet processes the input in two phases. In the downsampling phase, convolutional layers and strided convolutions are applied to downsample the grid to coarser representations. In the upsampling phase, transposed convolutions upsample the coarse grids to finer resolutions. These are concatenated with the outputs from the downsampling phase and processed by convolutional layers. We consider a UNet variant including residual blocks and normalization layers, which has achieved state-of-the-art performance in neural PDE surrogate modeling (Gupta & Brandstetter, 2023; Lippe et al., 2023).

Decoder. As decoder we consider a temporal convolutional model, which first reshapes o^t appropriately and then applies a learned convolution along the temporal axis. This structure has shown to improve predictions by smoothing the output signal across the bundle (Brandstetter et al., 2022).
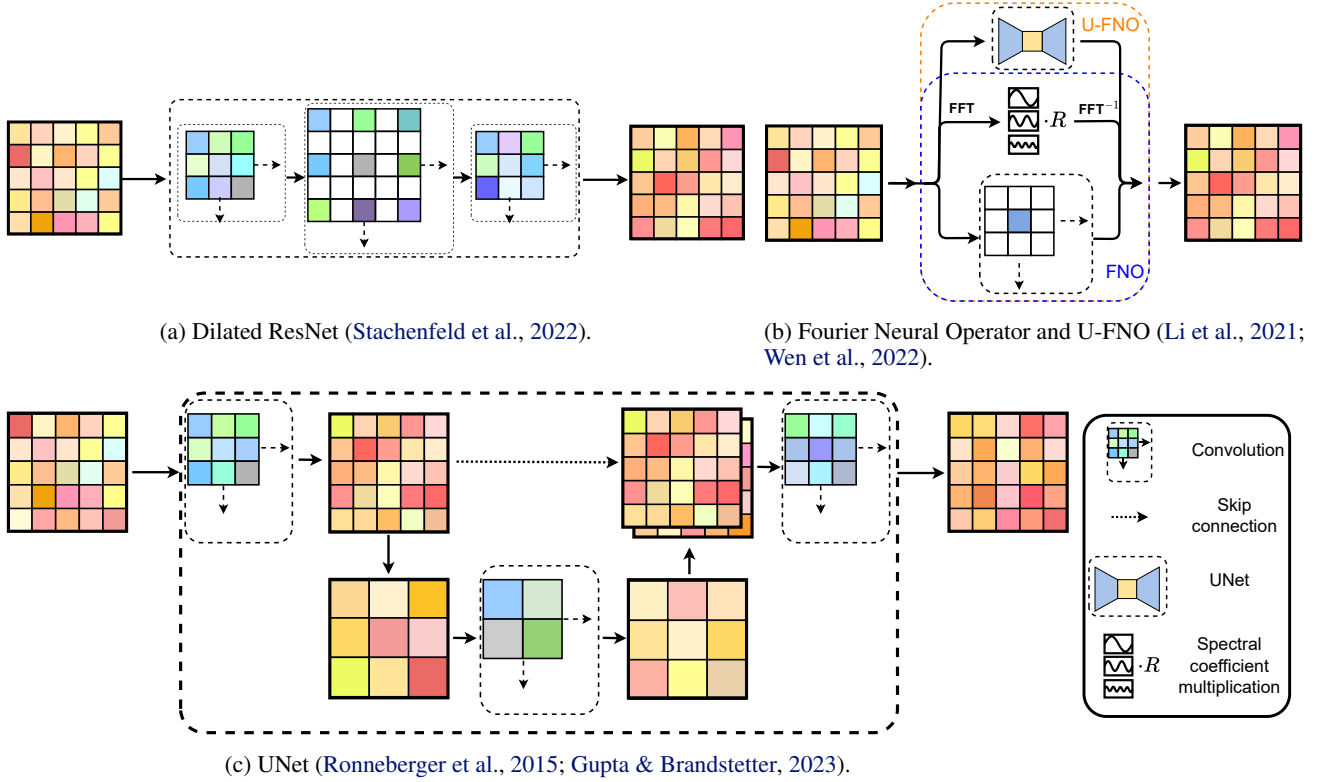
(a) Dilated ResNet (Stachenfeld et al., 2022).

(b) Fourier Neural Operator and U-FNO (Li et al., 2021; Wen et al., 2022).

(c) UNet (Ronneberger et al., 2015; Gupta & Brandstetter, 2023).

Figure 2. Schematic overview of model architectures.

3.3. Adaptations for two-phase flow problems.

While the aforementioned architectures have shown great results on various datasets, typical problems like turbulence and climate modeling do not necessarily share the same characteristics as two-phase flow problems. To tailor the methods to this specific task, we introduce several modifications that are compatible with all model architectures.

First, we condition the models on the geometry of the system. At each autoregressive step the field describing the boundaries – the obstacles and the pore – is concatenated along the channel axis, both for the encoder input and for the hidden layers in the processor. For the latter it is resampled to match the hidden representations' spatial grid if necessary, which only concerns UNet among the considered architectures. Additionally, after each step, the output is post-processed to zero out the non-fluid areas. This follows a similar approach to channel-based conditioning techniques for scalar PDE parameters, for example as in Takamoto et al. (2023), but additionally preserves the spatial structure of the conditioning signal internally throughout all processor layers.

Second, all of the architectures are adapted to respect the horizontal periodic boundary conditions of the problem setting. For the convolutional models, adding appropriate circular padding takes care of this property. For the spectral com-

ponent of the U-FNO, the FFT already respects periodicity, requiring no further adaptations.

Third, we know that the mass of both phases should be preserved over time. However, in practice, the reference solver accumulates small numerical errors that slightly change the balance. To account for this we apply a correction after each prediction step, similar to McGreivy & Hakim (2023), but rather than enforcing exact mass conservation we only do so approximately. We apply a smooth clipping using a hyperbolic tangent, scaled according to the mass deviation and a maximum allowed deviation. For the total mass m_{t+i} at future step $t + i$, using as reference the mass m_t at last input timestep t, we renormalize as follows:

$$\hat{m}_{t+i} = m_t \cdot \left(1 + i \cdot \epsilon \tanh\left(\frac{m_{t+i}/m_t - 1}{i \cdot \epsilon}\right)\right), \quad (7)$$

where ϵ denotes the fraction of maximum mass deviation per timestep. Near perfect conservation results in almost no rescaling due do the linear behavior around 0, with a smooth clipping in the limits.

4. Experiments

4.1. Data

We construct a dataset consisting of simulations of water flowing over a dead-end pore filled with oil, similar to Linga

et al. (2019). We simulate up to physical time $T = 50$. The spatial domain consists of a tube with a width of 3 and a height of 1, with a pore of a radius uniformly sampled from $[0.05, 0.25]$ at the top. Horizontal boundaries are periodic, effectively making it an infinite length tube. A surface charge is applied to induce electrowetting; charge values are sampled from the range $[-10, -1]$, biased towards more negative charges. The reasoning is that greater amplitudes result in more interesting behavior (Linga et al., 2019); at lower charges the droplet sometimes does not leave the pore.

To further increase the problem complexity we extend the numerical solver such that each simulation has one or two circular obstacles added to the internal domain. These circles have radii uniformly sampled from [0.05, 0.25] and have uniformly sampled center coordinates, while ensuring that they do not overlap, and taking into account a small minimum distance from the domain boundaries. We generate 920 simulations for this dataset, where 70% is used for training, 10% for validation, and 20% for testing. In order to illustrate the comparative complexity, we also generate a baseline dataset with no obstacles, for which we generate 552 simulations with the same data split. For both datasets we model the phase field ϕ, which describes the composition of the liquid as a value in $[-1, 1]$: -1 denotes only water, 1 only oil, and other values indicate a mix of both.

The solver's temporal discretization $dt = 0.02$ is sampled every 5 timesteps resulting in 500 timesteps per simulation. The finite element mesh with a resolution of 60 cells across the diameter is interpolated to an equidistant spatial grid of 96-by-64. The horizontal dimension is fixed, the vertical dimension is padded depending on the size of the pore, to unify all samples in the dataset to the same spatial dimension.

4.2. Experimental setup

The neural surrogate models f_θ learn to approximate a time-stepping operator acting on time bundles of the solution's phase field. Each bundle consists of 25 timesteps. Models are optimized using the MSE. To avoid overfitting on single-step predictions, we use pushforward training (Brandstetter et al., 2022). Rather than unrolling once and computing the loss, we do multiple forward predictions of the model and compute the loss using the final prediction, where the loss is propagated only through the last prediction step. Starting at a single unrolling, the number of forward passes is increased by 1 each 25 epochs, up to a maximum of 8 (making the total unrolling window span 200 timesteps). We train for 500 epochs using the Adam optimizer (Kingma & Ba, 2015) with an initial learning rate of 10^{-4}, which is decayed by 0.4 at epochs 25, 125, 250 and 375. All models end with a hyperbolic tangent activation to map to the [-1, 1] phase field range. After each
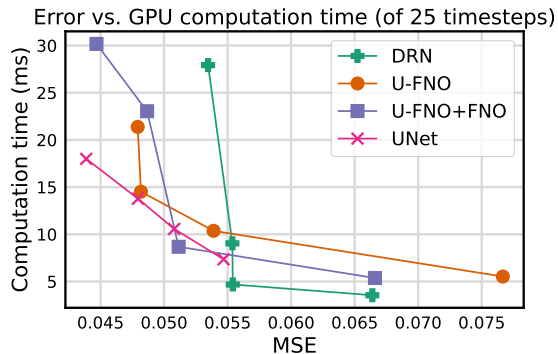

(a) Results on the GPU (NVIDIA RTX 3080).
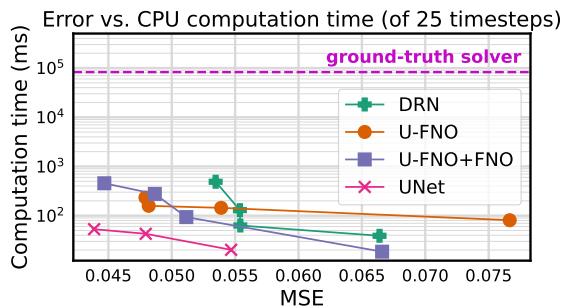
(b) Results on the CPU (AMD Ryzen 9 5900X), with the compute cost of the ground-truth solver as reference. Note that hyperparameter ranges were picked for GPU inference, and there may be settings yielding similar test errors better suited for CPU inference.

Figure 3. Work-precision diagrams for full rollout MSE on the test set versus the inference time of 25 timesteps (1 block), for the GPU and CPU. All models share the same general architecture with the processor parameters scaled. Parameter ranges were scanned to find ranges that scaled well w.r.t. both MSE and inference speed on the GPU; Pareto-optimal settings are plotted.

prediction we enforce boundary conditions within the domain and apply approximate mass conservation with a maximum deviation fraction of $\epsilon = 4 \cdot 10^{-4}$ per timestep. The data and code can be found on https://github.com/yoeripoels/neural-pde-surrogates.

4.3. Results

Model comparison. As main performance metrics we consider inference speed and the test-set MSE of the phase field, calculated over full simulation unrollings. Specifically, the initial block of $t = 0$ up to $t = 25$ is provided as input, after which the model autoregressively predicts $t = 25$ to $t = 500$. Computation time is measured as the inference time of 25 timesteps (1 block) using an NVIDIA RTX 3080 GPU and AMD Ryzen 9 5900X CPU. Models are implemented in PyTorch 2 and make use of `torch.compile` to speed up inference (Ansel et al., 2024). At the time of writing, compilation for complex operators is not supported, making the computation time comparison somewhat unfa-
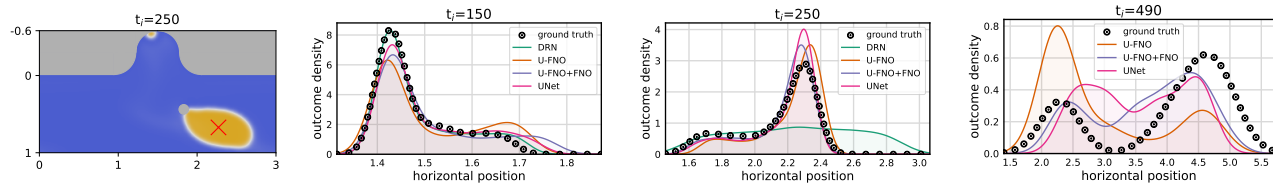
(a) Visualization of a solution at $t = 250$. The red 'X' indicates the tracked center point.

(b) Distributions of the droplet's horizontal position at various timesteps. A position exceeding the domain length (i.e. 3) indicates looping around due to the periodic BCs. The DRN is missing from the last plot due to the droplet having disintegrated at this timestep in multiple cases.

Figure 4. Comparison of distributions with small variations of the charge parameter, for a charge of -8.65 ± 20% (± 15 simulations). At earlier times the distributions mostly overlap, however in the final case there is a significant deviation. The multi-modal behavior – where in some cases the droplet gets stuck on the obstacle, and in others it moves past – is recovered in most cases, but not very precisely.

vorable for U-FNO models.

We compare the different models and their respective speed-accuracy trade-offs in Figure 3. All reported results are on the test set, and hyperparameters were selected such that they scale well with performance on the validation set; see Appendix A for details. In general the UNet seems to provide the most favorable scaling. To place the computation time in context, we also plot the reference solvers cost for an equal timeframe, on equal hardware, in Figure 3b. We see that the neural surrogates can perform inference orders of magnitude faster than the numerical solver even on the same hardware. Futhermore, it is highly non-trivial to adapt existing numerical codes to exploit the parallel computation of GPUs, whereas neural surrogates can exploit this out-of-the-box. We note that this comparison needs to be viewed in the context of neural surrogates introducing some simulation error and requiring the generation of high-quality training data, as opposed to the numerical solver. Nonetheless, neural surrogates could enable many use cases, such as inverse design, where approximate solutions for a large search space of parameters that lie within the training distribution suffice.

In addition to low MSE, in almost all cases the surrogates also produce simulations that qualitatively match well with the numerical solver. See for example Figure 1, depicting a test set simulation alongside the best-performing UNet's predictions. For more qualitative results we refer to Appendix B.

Parameter sensitivity. As mentioned before, small changes in the input parameters can lead to significant changes in the spatiotemporal profile of the droplet behavior. An example of this bifurcating behavior occurs when, given certain charge parameters, the droplet gets stuck on an obstacle, but detaches after some time. This detachment time depends on the exact values of the electrical charge. In these cases, the pixel-wise MSE is a poor metric to evaluate whether the neural surrogate qualitatively matches the behavior of the numerical solver, as a small error in the predicted time of detachment would already lead to a large MSE.

In Figure 4 we evaluate such a case from the test set. We investigate the droplet movement for various perturbations of the surface charge parameter, with otherwise identical conditions. Each neural surrogate model architecture is evaluated using the configuration with the lowest MSE of Figure 3. To compare whether they qualitatively match the reference simulations, we track the droplet's center and plot the distribution of the horizontal coordinate at several points in time. Over smaller time windows the distribution clearly matches the one generated by the numerical solver for most models. However, near the end of the simulation, the bimodal distribution is only approximately recovered, suggesting that neural PDE surrogates struggle to model such bifurcating dynamics over long time horizons.

Ablations. Finally, we confirm the benefit of the adaptations made to each of the model architectures. Table 1 denotes these results: for each model, we take the best architectures (column 'All') and disable the made adaptations. Column '\Inv' denotes models with the invariances one can reasonably learn from data disabled. That is, these models do not have periodic boundary conditions enforced through the architecture, and do not have any mass conservation built in. For all models adding periodicity and mass conservation improved model performance. Column '\BC' denotes models with no enforced spatial boundary conditions within the domain, that is, obstacles within the domain are not explicitly passed as conditioning in the internal model layers, and are not corrected after each autoregressive step. In most cases disabling this adaptation led to significant performance degradation. Finally, to confirm the relative complexity of the adaptations that were made to the original oil expulsion setting, we benchmark all models on a dataset with no obstacles, as in Linga et al. (2019). The results are shown in column '\Obs'. Besides the absence of obstacles, the domains and simulation conditions are identical, making the error metrics lie on the same scale. The no-obstacle dataset contains approximately half of the number of training set simulations, and model hyperparameters were not optimized for this setting. Nevertheless, the test errors are

Table 1. Evaluations using the settings with the lowest errors. *Middle:* Ablations of the introduced adaptations. \Inv denotes the enforced (approximate) invariances disabled, that is, no enforced periodicity and approximate mass conservation. \BC denotes no enforcement of the spatial boundary conditions of the geometry. *Right:* Results when applying the model to the dataset with no obstacles, denoted as \Obs. Even with no hyperparameter optimization, these models have considerably lower errors, indicating the challenging nature of the more complex setting we consider.

Model	All	\Inv	\BC	\Obs
DRN	**0.0535**	0.0647	0.362	0.0168
U-FNO	**0.0479**	0.0513	0.395	0.0186
U-FNO+FNO	**0.0447**	0.0573	0.059	0.0167
UNet	**0.0439**	0.0465	0.112	0.0199

two to three times lower, indicating that the dynamics get significantly more challenging by varying the geometry.

5. Conclusion

We investigated neural PDE solvers for two-phase flows, more precisely for the expulsion of oil droplets using electrowetting. We extended the existing simulation setting of Linga et al. (2019) to more complex scenarios by introducing varying complex geometries in the domain. Here, the computational cost of for example inverse design tasks becomes prohibitively expensive using high-fidelity forward solves, making fast surrogate modeling approaches crucial. Further, the setting forms a harder and consequently more interesting benchmark for neural surrogate approaches. We investigated and extended three prevalent neural PDE solver methods in terms of their speed-accuracy trade-off, where we found the UNet to scale most favorably. Further, the UNet and U-FNO models both showed comparatively good qualitative performance when comparing aggregate properties of the droplet dynamics. Still, all models struggled to preserve these properties over long simulation horizons. Lastly, we showed the benefits of the introduced model extensions for the oil expulsion setting.

A future research direction would be to investigate more two-phase flow datasets (Linga et al., 2019; 2020; Almasi et al., 2021) and to evaluate more neural PDE solver architectures, for example a selection of Lu et al. (2021); Cao (2021); Brandstetter et al. (2022); Tran et al. (2023); Li et al. (2023); Hao et al. (2023); Hagnberger et al. (2024); Serrano et al. (2023). Additionally, a probabilistic approach could help in taking the uncertainties in bifurcating dynamics into account (Cachay et al., 2023; Yang & Sommer, 2023; Minartz et al., 2023; Bergamin et al., 2024) or to mitigate autoregressive error accumulation (Lippe et al., 2023). Finally, one could investigate the speed-up compared to reference numerical solvers and their scalability in more de-

tail. In particular, while inference of the resulting networks is significantly faster, a dataset of expensive simulations must be generated up front. An interesting question is to consider the amortized cost of using neural surrogates in a downstream application. For example, one could do a large number of inverse problem optimizations using either only the high-fidelity numerical method, or by first generating a dataset, training a neural surrogate, and then consequently using a combination of both for the optimization. Such an evaluation could better evaluate the real-world applicability of the investigated methods.

6. Acknowledgements

References

Almasi, F., Shadloo, M., Hadjadj, A., Ozbulut, M., Tofighi, N., and Yildiz, M. Numerical simulations of multi-phase electro-hydrodynamics flows using a simple incompressible smoothed particle hydrodynamics method. *Computers & Mathematics with Applications*, 81:772–785, 2021. ISSN 0898-1221. doi: 10.1016/j.camwa.2019.10.029. URL https://www.sciencedirect.com/science/article/pii/S0898122119305243.

Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., and Wells, G. N. The fenics project version 1.5. *Archive of Numerical Software*, 3(100):9–23, 2015. doi: 10.11588/ANS.2015.100.20553. URL http://journals.ub.uni-heidelberg.de/index.php/ans/article/view/20553.

Ansel, J., Yang, E., He, H., Gimelshein, N., Jain, A., Voznesensky, M., Bao, B., Bell, P., Berard, D., Burovski, E., Chauhan, G., Chourdia, A., Constable, W., Desmaison, A., DeVito, Z., Ellison, E., Feng, W., Gong, J., Gschwind, M., Hirsh, B., Huang, S., Kalambarkar, K., Kirsch, L., Lazos, M., Lezcano, M., Liang, Y., Liang, J., Lu, Y., Luk, C. K., Maher, B., Pan, Y., Puhrsch, C., Reso, M., Saroufim, M., Siraichi, M. Y., Suk, H., Zhang, S., Suo, M., Tillet, P., Zhao, X., Wang, E., Zhou, K., Zou, R., Wang, X., Mathews, A., Wen, W., Chanan, G., Wu, P., and Chintala, S. PyTorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming*

Languages and Operating Systems, Volume 2, ASPLOS '24, pp. 929–947, 2024. ISBN 9798400703850. doi: 10.1145/3620665.3640366. URL https://doi.org/10.1145/3620665.3640366.

Azizzadenesheli, K., Kovachki, N., Li, Z., Liu-Schiaffini, M., Kossaifi, J., and Anandkumar, A. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 6(5):320–328, May 2024. ISSN 2522-5820. doi: 10.1038/s42254-024-00712-5. URL https://doi.org/10.1038/s42254-024-00712-5.

Bergamin, F., Diaconu, C., Shysheya, A., Perdikaris, P., Hernández-Lobato, J. M., Turner, R. E., and Mathieu, E. Guided autoregressive diffusion models with applications to PDE simulation. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024. URL https://openreview.net/forum?id=1avNKFEIOL.

Bonev, B., Kurth, T., Hundt, C., Pathak, J., Baust, M., Kashinath, K., and Anandkumar, A. Spherical Fourier neural operators: Learning stable dynamics on the sphere. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2806–2823. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/bonev23a.html.

Brandstetter, J., Worrall, D. E., and Welling, M. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vSix3HPYKSU.

Buhendwa, A. B., Adami, S., and Adams, N. A. Inferring incompressible two-phase flow fields from the interface motion using physics-informed neural networks. *Machine Learning with Applications*, 4:100029, 2021. ISSN 2666-8270. doi: 10.1016/j.mlwa.2021.100029. URL https://www.sciencedirect.com/science/article/pii/S2666827021000104.

Cachay, S. R., Zhao, B., Joren, H., and Yu, R. DYffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 45259–45287, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8df90a1440ce782d1f5607b7a38f2531-Paper-Conference.pdf.

Cao, S. Choose a transformer: Fourier or galerkin. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/d0921d442ee91b896ad95059d13df618-Paper.pdf.

Gao, Z., Shi, X., Han, B., Wang, H., Jin, X., Maddix, D., Zhu, Y., Li, M., and Wang, Y. B. PreDiff: Precipitation nowcasting with latent diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 78621–78656, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/f82ba6a6b981fbbecf5f2ee5de7db39c-Paper-Conference.pdf.

Gopakumar, V., Pamela, S., Zanisi, L., Li, Z., Gray, A., Brennand, D., Bhatia, N., Stathopoulos, G., Kusner, M., Deisenroth, M. P., Anandkumar, A., the JOREK Team, and MAST Team. Plasma surrogate modelling using fourier neural operators. *Nuclear Fusion*, 64(5):056025, apr 2024. doi: 10.1088/1741-4326/ad313a. URL https://dx.doi.org/10.1088/1741-4326/ad313a.

Gupta, J. K. and Brandstetter, J. Towards multi-spatiotemporal-scale generalized PDE modeling. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=dPSTDbGtBY.

Hagnberger, J., Kalimuthu, M., Musekamp, D., and Niepert, M. Vectorized conditional neural fields: A framework for solving time-dependent parametric partial differential equations. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=sF9epWkNUG.

Hanna, J. M., Aguado, J. V., Comas-Cardona, S., Askri, R., and Borzacchiello, D. Residual-based adaptivity for two-phase flow simulation in porous media using physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 396:115100, 2022. ISSN 0045-7825. doi: 10.1016/j.cma.2022.115100. URL https://www.sciencedirect.com/science/article/pii/S004578252200295X.

Hao, Z., Wang, Z., Su, H., Ying, C., Dong, Y., Liu, S., Cheng, Z., Song, J., and Zhu, J. GNOT: A general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

Hassan, S. M. S., Feeney, A., Dhruv, A., Kim, J., Suh, Y., Ryu, J., Won, Y., and Chandramowlishwaran, A. Bubbleml: A multiphase multiphysics dataset and benchmarks for machine learning. In *Advances in Neural*

Information Processing Systems, volume 36, pp. 418–449, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/01726ae05d72ddba3ac784a5944fa1ef-Paper-Datasets_and_Benchmarks.pdf.

Jiang, Z., Zhu, M., Li, D., Li, Q., Yuan, Y. O., and Lu, L. Fourier-MIONet: Fourier-enhanced multiple-input neural operators for multiphase modeling of geological carbon sequestration. *arXiv preprint arXiv:2303.04778*, 2023. URL https://doi.org/10.48550/arXiv.2303.04778.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, volume 3, 2015. URL https://doi.org/10.48550/arXiv.1412.6980.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Li, Z., Meidani, K., and Farimani, A. B. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=EPPqt3uERT.

Lin, C., Li, Z., Lu, L., Cai, S., Maxey, M., and Karniadakis, G. E. Operator learning for predicting multiscale bubble growth dynamics. *Journal of Chemical Physics*, 154(10), 3 2021a. ISSN 0021-9606. doi: 10.1063/5.0041203. URL https://www.osti.gov/biblio/1853305.

Lin, C., Maxey, M., Li, Z., and Karniadakis, G. E. A seamless multiscale operator neural network for inferring bubble dynamics. *Journal of Fluid Mechanics*, 929:A18, 2021b. doi: 10.1017/jfm.2021.866. URL https://dx.doi.org/10.1017/jfm.2021.866.

Linga, G., Bolet, A., and Mathiesen, J. Bernaise: A flexible framework for simulating two-phase electrohydrodynamic flows in complex domains. *Frontiers in Physics*, 7, March 2019. ISSN 2296-424X. doi: 10.3389/fphy.2019.00021. URL http://dx.doi.org/10.3389/fphy.2019.00021.

Linga, G., Bolet, A., and Mathiesen, J. Transient electrohydrodynamic flow with concentration-dependent fluid properties: Modelling and energy-stable numerical schemes. *Journal of Computational Physics*, 412:109430, 2020. ISSN 0021-9991. doi: 10.1016/j.jcp.2020.109430. URL https://www.sciencedirect.com/science/article/pii/S0021999120302047.

Lippe, P., Veeling, B., Perdikaris, P., Turner, R., and Brandstetter, J. PDE-Refiner: Achieving accurate long rollouts with neural pde solvers. In *Advances in Neural Information Processing Systems*, volume 36, pp. 67398–67433, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d529b943af3dba734f8a7d49efcb6d09-Paper-Conference.pdf.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, Mar 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00302-5. URL https://doi.org/10.1038/s42256-021-00302-5.

Lu, M., Lin, C., Maxey, M., Karniadakis, G., and Li, Z. Bridging scales in multiscale bubble growth dynamics with correlated fluctuations using neural operator learning. *arXiv preprint arXiv:2403.13299*, 2024. URL https://doi.org/10.48550/arXiv.2403.13299.

McGreivy, N. and Hakim, A. Invariant preservation in machine learned PDE solvers via error correction. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023. URL https://openreview.net/forum?id=RLfs4A0ZJg-.

Minartz, K., Poels, Y., Koop, S., and Menkovski, V. Equivariant neural simulators for stochastic spatiotemporal dynamics. In *Advances in Neural Information Processing Systems*, volume 36, pp. 38930–38957, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7a8d388b7a17df480856dff1cc079b08-Paper-Conference.pdf.

Nguyen, T., Brandstetter, J., Kapoor, A., Gupta, J. K., and Grover, A. ClimaX: A foundation model for weather and climate. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25904–25938. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/nguyen23a.html.

Poels, Y., Derks, G., Westerhof, E., Minartz, K., Wiesen, S., and Menkovski, V. Fast dynamic 1D simulation of divertor plasmas with neural PDE surrogates. *Nuclear Fusion*, 63(12):126012, sep 2023. doi: 10.1088/1741-4326/acf70d. URL https://dx.doi.org/10.1088/1741-4326/acf70d.

Pollack, M. G., Shenderov, A. D., and Fair, R. B. Electrowetting-based actuation of droplets for integrated microfluidics. *Lab Chip*, 2:96–101, 2002. doi: 10.1039/

B110474H. URL http://dx.doi.org/10.1039/B110474H.

Raissi, M., Perdikaris, P., and Karniadakis, G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

Ren, X. and Lyu, X. Mixed form based physics-informed neural networks for performance evaluation of two-phase random materials. *Engineering Applications of Artificial Intelligence*, 127:107250, 2024. ISSN 0952-1976. doi: 10.1016/j.engappai.2023.107250. URL https://www.sciencedirect.com/science/article/pii/S0952197623014343.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241. Springer International Publishing, 2015. ISBN 978-3-319-24574-4. URL https://link.springer.com/chapter/10.1007/978-3-319-24574-4_28.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html.

Serrano, L., Le Boudec, L., Kassaï Koupaï, A., Wang, T. X., Yin, Y., Vittaut, J.-N., and Gallinari, P. Operator learning with neural fields: Tackling pdes on general geometries. In *Advances in Neural Information Processing Systems*, volume 36, pp. 70581–70611, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/df54302388bbc145aacaa1a54a4a5933-Paper-Conference.pdf.

Stachenfeld, K., Fielding, D. B., Kochkov, D., Cranmer, M., Pfaff, T., Godwin, J., Cui, C., Ho, S., Battaglia, P., and Sanchez-Gonzalez, A. Learned coarse models for efficient turbulence simulation. In *International Conference on Learning Representations*, volume 10, 2022. URL https://openreview.net/forum?id=msRBojTz-Nh.

Takamoto, M., Alesiani, F., and Niepert, M. Learning neural PDE solvers with parameter-guided channel attention. In

Proceedings of the 40th International Conference on Machine Learning, volume 202 of *Proceedings of Machine Learning Research*, pp. 33448–33467. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/takamoto23a.html.

Tran, A., Mathews, A., Xie, L., and Ong, C. S. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tmIiMPl4IPa.

Wen, G., Li, Z., Azizzadenesheli, K., Anandkumar, A., and Benson, S. M. U-FNO—an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022. ISSN 0309-1708. doi: 10.1016/j.advwatres.2022.104180. URL https://www.sciencedirect.com/science/article/pii/S0309170822000562.

Yang, G. and Sommer, S. A denoising diffusion model for fluid field prediction. *arXiv preprint arXiv:2301.11661*, 2023. URL https://doi.org/10.48550/arXiv.2301.11661.

Zhai, H., Zhou, Q., and Hu, G. Predicting micro-bubble dynamics with semi-physics-informed deep learning. *AIP Advances*, 12(3):035153, 03 2022. ISSN 2158-3226. doi: 10.1063/5.0079602. URL https://doi.org/10.1063/5.0079602.

Zhang, Z., Yan, X., Liu, P., Zhang, K., Han, R., and Wang, S. A physics-informed convolutional neural network for the simulation and prediction of two-phase darcy flows in heterogeneous porous media. *Journal of Computational Physics*, 477:111919, 2023. ISSN 0021-9991. doi: doi.org/10.1016/j.jcp.2023.111919. URL https://www.sciencedirect.com/science/article/pii/S0021999123000141.

Zhu, X., Hu, X., and Sun, P. Physics-informed neural networks for solving dynamic two-phase interface problems. *SIAM Journal on Scientific Computing*, 45(6): A2912–A2944, 2023. doi: 10.1137/22M1517081. URL https://doi.org/10.1137/22M1517081.

A. Hyperparameters and results

This appendix contains the hyperparameters and exact figures for runtime and accuracy for models that lie on the pareto fronts shown in Figure 3a. For UNets, unless otherwise mentioned, we use a fixed kernel size of 3 and a residual block depth of 1. For the DRN, the per-block dilation rates are fixed at (1, 2, 4, 8, 4, 2, 1). We refer to the respective papers (Stachenfeld et al., 2022; Gupta & Brandstetter, 2023; Wen et al., 2022) and our code for more details.

Table 2. Pareto-optimal hyperparameters and results for DRN.

Layers	Kernel size	Hidden features	MSE	GPU runtime (ms)	CPU runtime (ms)
2	5	384	0.053	27.9	488.8
2	5	192	0.055	9.0	129.4
2	5	128	0.055	4.7	63.1
4	5	64	0.066	3.5	39.0

Table 3. Pareto-optimal hyperparameters and results for U-FNO.

Layers	FNO modes	Hidden features	UNet channel multipliers	MSE	GPU runtime (ms)	CPU runtime (ms)
3	10	192	(1,1)	0.048	21.4	233.7
2	10	192	(1,1)	0.048	14.5	158.3
2	10	192	(1)	0.054	10.4	144.2
1	10	192	(1, 1)	0.077	5.5	80.8

Table 4. Pareto-optimal hyperparameters and results for U-FNO + FNO.

Layers	FNO modes	Hidden features	UNet channel multipliers	MSE	GPU runtime (ms)	CPU runtime (ms)
(FNO, U-FNO) ×2	10	256	(1,1,1,1)	0.045	30.2	455.3
(FNO, U-FNO) ×2	10	256	(1,1)	0.049	23.0	276.3
(FNO, U-FNO)	10	192	(1,1)	0.051	8.7	93.5
(FNO, U-FNO)	6	64	(1)	0.067	5.4	18.5

Table 5. Pareto-optimal hyperparameters and results for UNet.

Channel multipliers	Residual block depth	Hidden features (1st layer)	MSE	GPU runtime (ms)	CPU runtime (ms)
(2,2,1,2)	2	32	0.044	18.0	53.0
(2,2,2)	2	32	0.048	13.8	42.8
(2,2)	2	64	0.051	10.6	66.0
(2,2)	1	32	0.055	7.4	20.2

B. Additional qualitative results

This appendix contains additional qualitative results for all model architectures. For each model architecture we use the hyperparameters that resulted in the lowest MSE. We plot three examples: a case that works well for all in Figure 5a, a mixed case in Figure 5b, and a failure case in Figure 5c. In the last, one can clearly see that the models struggle with the outlier case of an even split of the droplet into two separate droplets.
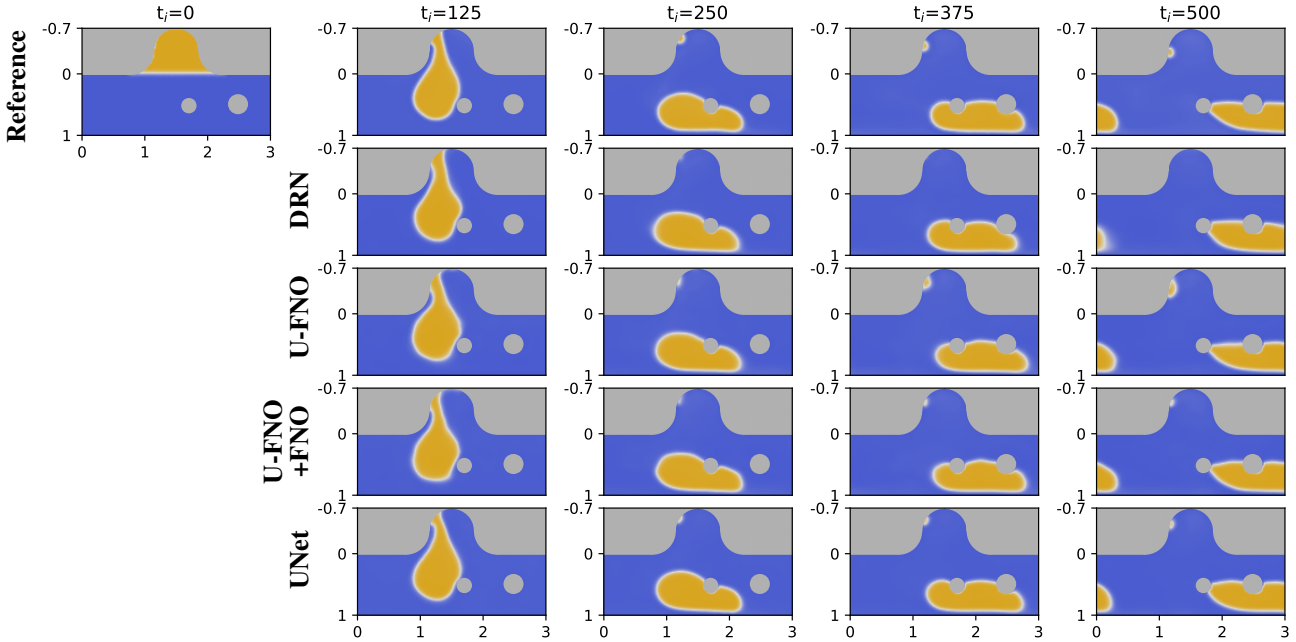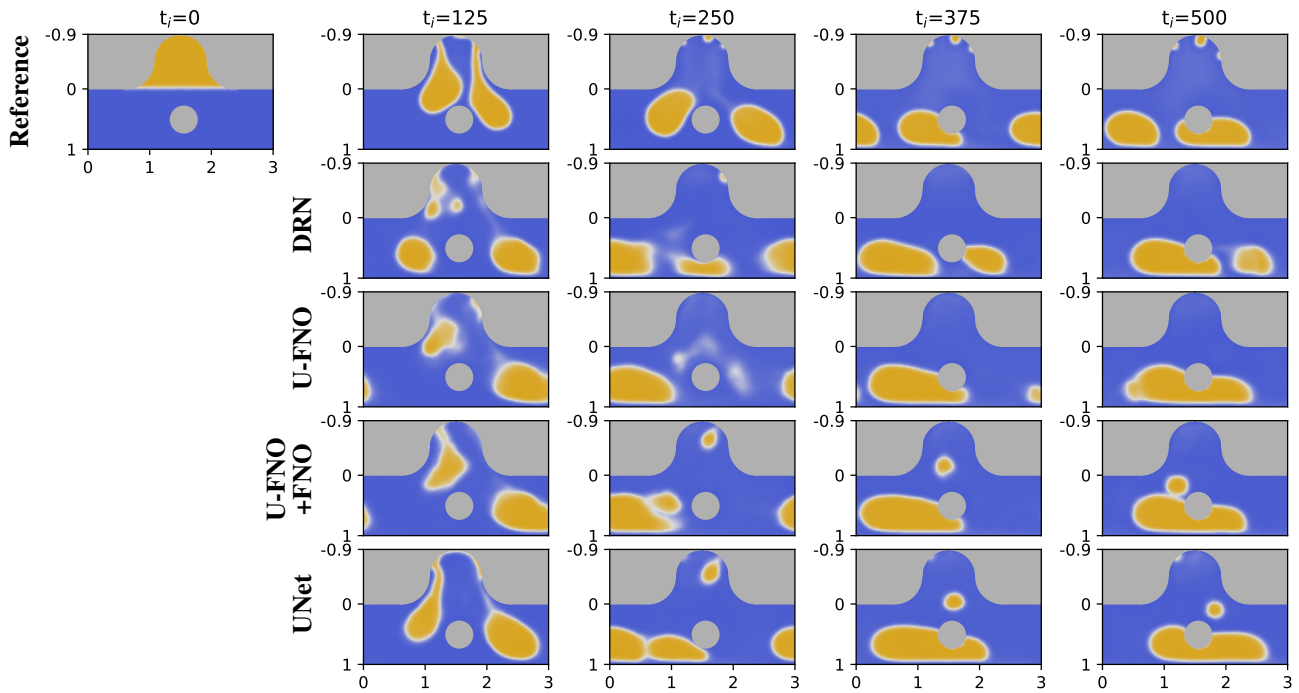

(a) Example with two obstacles; all models qualitatively match the reference solution.
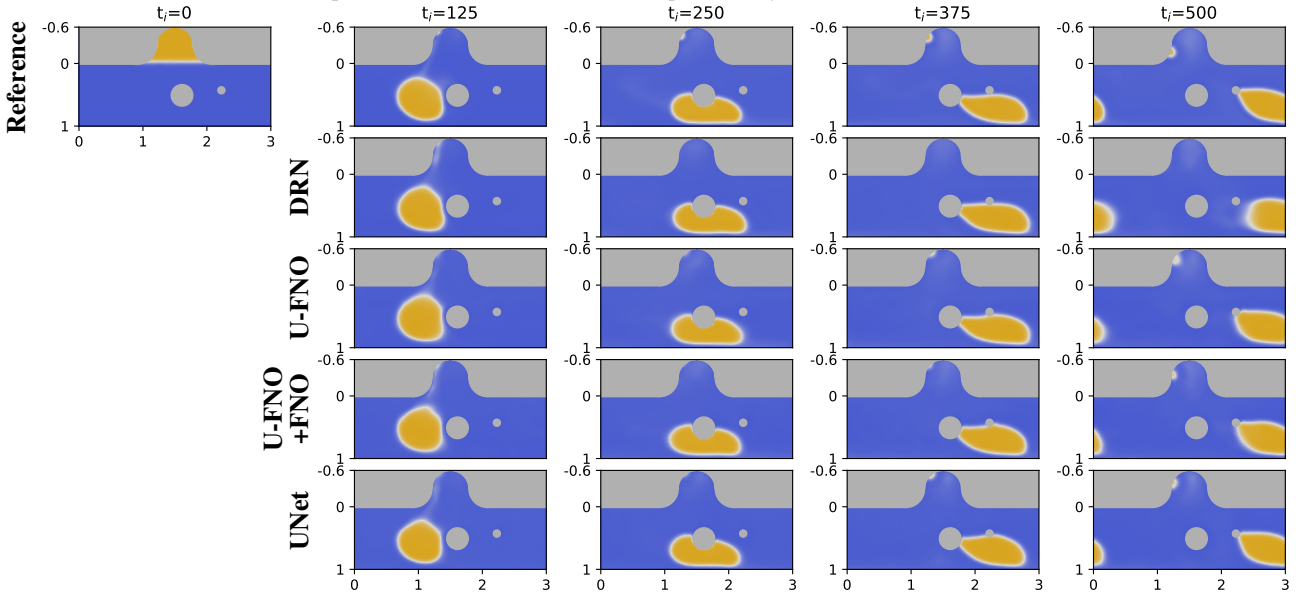
(b) Example with two obstacles; here, the DRN struggles to reproduce the behavior near the final timestep.

(c) Failure mode for the neural PDE surrogates: they cannot accurately reproduce the outlier case of an evenly splitting droplet. At some point during the simulation all models merge the two droplets back into one, which does not happen in the reference solution.

Figure 5. Test set example with predictions of all model classes considered, using the settings within each class that led to the lowest MSE.