Conceptually closest to our work is the recent PDE-Refiner model class (Lippe et al., 2024). PDE-Refiner draws inspiration from diffusion models to apply a small number of refinement steps on learned Eulerian solvers. The refinement steps substantially improve the modeling of high frequency components, which yields more stable long-term predictions and better physics modeling, at the cost of increased inference time and a dedicated training routine. We point out that because PDE-Refiner is designed for Eulerian systems, it does not have the notion of dynamic particle coordinates underlying Lagrangian methods. Thus, extending PDE-Refiner to the Lagrangian description is not trivial, as one could choose to refine the accelerations or velocities or directly the particle coordinates, and such investigations are beyond the scope of this work. Furthermore, for particle systems, we do not have efficient ways to accurately evaluate high spatial frequencies over point clouds akin to the FFT on grids, and additionally, the physical setup of our problems does not involve high spatial frequencies.

4. Experiments

Our analyses are based on the datasets of Toshev & Adams (2024), accompanying the LagrangeBench paper (Toshev et al., 2023a). These datasets represent challenging coarse-grained temporal dynamics and contain long trajectories, i.e., up to thousands of steps. We test the difference in performance of two popular GNN-based simulators: (i) when the external forces are removed from the model outputs (indicated by subscript g), (ii) when an SPH relaxation is performed that is implied by a pressure term (indicated by subscript p), and (iii) when an SPH relaxation is performed implied by a viscosity term (indicated by subscript ν).

GNN-based simulators. The Graph Network-based Simulator (GNS) model (Sanchez-Gonzalez et al., 2020) is a popular learned surrogate for physical particle-based simulations and our main model. The architecture is kept simple, based on the encoder-processor-decoder principle, where the processor consists of multiple graph network blocks (Battaglia et al., 2018). Our second model, the Steerable E(3)-equivariant Graph Neural Network (SEGNN) (Brandstetter et al., 2022a) is a general implementation of an E(3) equivariant GNN, where layers are directly conditioned on steerable attributes for both nodes and edges. The main building block is the steerable MLP, i.e., a stack of learnable linear Clebsch-Gordan tensor products interleaved with gated non-linearities (Weiler et al., 2018). SEGNN layers are message-passing layers (Gilmer et al., 2017) where steerable MLPs replace the traditional non-equivariant MLPs for both message and node update functions. These two models were chosen as they present the current state-of-the-art surrogates for Lagrangian fluid dynamics (Toshev et al., 2023a), and also because they are representative of two fundamentally different classes of GNNs: non-equivariant (GNS) and equivariant (SEGNN).

Implementation of SPH relaxation. In our experience, it suffices to perform the relaxation operation for 1-5 steps, depending on the problem. We summarize the used hyperparameters in Table 3 and Appendix B. Given that the learned surrogate is trained on every 100th SPH step, these additional SPH relaxation steps only marginally increase the rollout time – by a factor of 1.05-1.15 per relaxation step for a 10-layer 128-dimensional GNS model simulating the 2D RPF case, see Table 4 and Appendix E. In the same table, we observe an increase in runtime for 3D RPF and GNS-10-128 of roughly 1.4x per relaxation step, but we believe that this comes from the much more compute-intense neighbor search, which is reevaluated at every relaxation step. However, as the relaxation does not need to be implemented in a differentiable framework (what we currently do), much more efficient implementations, e.g. in C++, can significantly reduce these runtimes. For more compute-intense models like SEGNN the slowdown factor reduces, as the relaxation has a fixed computational cost independent of the particular GNN model.

Most computational overhead of the relaxation is due to its neighbor list, which has significantly more edges than the default neighbor list of the GNN-based simulators. The GNN graph generation uses the default radial cutoff distance from LagrangeBench, which corresponds to roughly 1.5 average particle distances. In contrast, the SPH relaxation uses the Quintic spline kernel with a cutoff of 3 average particle distances, i.e., the SPH relaxation operates on 2^d more edges, with dimension $d \in \{2, 3\}$. Therefore, our approach can be regarded as a multiscale approach, similar to the learned multi-scale interatomic potential presented by (Fu et al., 2023a). The difference is that in our approach, only the part using the smaller cutoff is a neural network, and the longer-range interactions simply stabilize the system in terms of better density distributions.

Training with SPH relaxation. An appealing idea is to use the SPH relaxation as a regularization during training, in the hope that we can omit running relaxations at inference time. We tried various ways of implementing this idea, but none of them improved rollout performance, see Appendix H.

Overview of results. Our results on 400-step rollouts using the GNS model are summarized in Table 1 and are averaged over all test trajectories and over the trajectory length. See Table 2 for the SEGNN results. As error measures, we use (a) the mean-squared error of positions (MSE_{400}), (b) the Sinkhorn divergence, which quantifies the conservation of the particle distribution, and (c) the kinetic energy error (MSE_{Ekin}) as a global measure of the physical behavior. The viscous term is shown only for reverse Poiseuille flow because it did not improve the performance on the

other datasets. We note that by splitting the test sets into sequences of length 400, we obtain only 12-25 test trajectories, leading to noisy performance estimates. We discuss the necessity for larger datasets later in this section. For various parameter ablations, the evolution of error metrics with error bounds, and three more error metrics (density MAE, Dirichlet energy, and Chamfer distance), see Appendix G. Overall, all neural SPH-enhanced simulators achieve better performance than the baseline GNNs, often by orders of magnitude, allowing for significantly longer rollouts and significantly better physics modeling.

Note on error thresholds. We note that upon tuning the parameters of our method, it either improves performance or converges to the baseline, with the latter being what mainly happens to RPF 3D according to Appendix B. We hypothesize that the baseline already produces very good particle distributions, and there is little potential for improvement. It thus seems necessary to define a threshold of when a learned simulator performs *well enough* in the sense of the requirements of the downstream task of interest. We refer to physical thresholds like the *chemical accuracy* in computational chemistry or the *energy and forces within threshold* measure used in the Open Catalyst project (Chanussot et al., 2021), both of which are designed to quantify whether a computational model is useful for practical applications. We stress the importance and leave the derivations of such thresholds for Lagrangian fluid simulations to future work.

	Model	MSE_{400}	Sinkhorn	MSE_{Ekin}
2D TGV	GNS	$5.3e-4$	$5.4e-7$	$5.6e-7$
	GNS_p	$4.8e-4$	$1.7e-8$	$4.8e-7$
2D RPF	GNS	$2.7e-2$	$3.6e-7$	$4.3e-3$
	GNS_g	$2.7e-2$	$2.7e-7$	$3.7e-4$
	$\text{GNS}_{g,p}$	$2.7e-2$	$2.9e-8$	$4.1e-4$
	$\text{GNS}_{g,p,\nu}$	$2.7e-2$	$3.0e-8$	$1.4e-4$
2D LDC	GNS	$3.3e-2$	$3.1e-4$	$1.1e-4$
	GNS_p	$1.6e-2$	$2.8e-7$	$1.2e-6$
2D DAM	GNS	$1.9e-1$	$3.8e-2$	$4.6e-2$
	GNS_g	$8.0e-2$	$1.3e-2$	$9.4e-3$
	GNS_p	$9.7e-2$	$7.1e-3$	$5.8e-3$
	$\text{GNS}_{g,p}$	$8.4e-2$	$7.5e-3$	$2.1e-3$
3D TGV	GNS	$4.8e-2$	$4.1e-6$	$3.6e-2$
	GNS_p	$4.6e-2$	$9.0e-7$	$4.2e-2$
3D RPF	GNS	$2.3e-2$	$4.4e-7$	$1.7e-5$
	GNS_g	$2.3e-2$	$4.4e-7$	$4.1e-5$
	GNS_p	$2.3e-2$	$1.0e-7$	$1.5e-5$
	$\text{GNS}_{g,p}$	$2.3e-2$	$1.3e-7$	$4.1e-5$
3D LDC	GNS	$3.2e-2$	$2.0e-5$	$1.3e-7$
	GNS_p	$3.2e-2$	$1.1e-6$	$2.9e-8$

Table 1. Performance measures averaged over a rollout of 400-steps. An additional subscript g indicates that external forces are removed from the model outputs, subscript p indicates that the SPH relaxation has a pressure term, and subscript ν that the viscosity term is added to the SPH relaxation. The numbers in the table are averaged over all test trajectories. MSE_{400} corresponds to: MSE_{120} for 2D TGV, MSE_{55} for 3D TGV, and MSE_{395} for 2D DAM, as these are the full trajectory lengths excluding initial history size $H = 5$.

4.1. External Force Treatment

In this section, we study the influence of the proposed external force treatment without combining it with the SPH relaxation. As only the dam break and reverse Poiseuille flow datasets have external force features, we focus on them.

Dam break (DAM). We saw a major performance boost on dam break when removing external forces from the target (GNS_g), see Table 1 and Appendix G.1. This simple modification of the training objective improves all considered measures by at least a factor of 2 and by as much as a factor of 5 on a rollout of the full dam break trajectory, i.e., 400 steps. For up to 20-step rollouts, GNS_g training does not improve the position error, which is in accordance with Sanchez-Gonzalez et al. (2020) and their Fig. C1. However, as the simulation length goes beyond 50 steps, numerical errors quickly accumulate and lead to artifacts like the one visible in the top part of Fig. 1. This particular failure mode in the front part of the dam break wave develops by first compressing the fluid to as much as $1.5\rho_{ref}$, and then the smallest instability in the tip causes particles to fly up. From there on, GNS starts acting as if the right wall has already been reached and fails to model the double wave structure from the reference solution, see Appendix A.

Force smoothing in reverse Poiseuille Flow. The external force of the reverse Poiseuille flow 2D dataset is provided as a function corresponding to the instantaneous force, but when we train towards the effective dynamics over multiple original solver steps, we need to adjust this force. In particular, when predicting the effective dynamics over $M = 100$ temporal coarse-graining steps provided by LagrangeBench, a reverse Poiseuille flow particle might jump back and forth across the boundary that separates the left- and right-ward forcing. Thus, it is not possible to infer the aggregated external force directly from only knowing the particle coordinates at step M. We, therefore, apply a convolution of a Gaussian function with the forcing function. Since the forcing in RPF is a step function, this specific convolution possesses an analytical solution, i.e., the error function $\text{erf}(\cdot)$. We use $\text{erf}(\cdot)$ as a drop-in replacement to the original force function. See Appendix D for more details and visualization of the force before and after the convolution.

Reverse Poiseuille flow (RPF). See Fig. 3 for a subset of our ablation results on RPF 2D with GNS-10-128, or the full results on RPF 2D/3D and GNS/SEGNN in Appendix G.3. When removing external forces from the target of the GNS model (GNS_g), we observed that using the original, i.e., not smoothed, force leads to highly unstable dynamics in the shearing region, which causes the failure of the dynamics after less than 50 steps, see $\text{GNS}_{g_{raw}}$ in Figs. 27 and 28.

When switching to the smoothed force function, the system becomes much more stable to perturbations and significantly improves the kinetic energy error. It is important to note that the kinetic energy is paramount to RPF, as this physical system is characterized by constant kinetic energy up to small fluctuations.

Looking at the 20-step position MSE reported in LagrangeBench, the GNS$_g$ training leads to worse performance, roughly by a factor of 1.5 (see the beginning of the evolution in Fig. 3). This is important to note because we trade off worse short-term behavior in favor of better long-rollout performance, with the latter being the practical use-case we target. In this context, the LagrangeBench datasets pre-define a split of 50/25/25, which is far from enough if we want stable error estimates on rollouts of 400-step length, as also discussed, e.g., in Fu et al. (2023b).

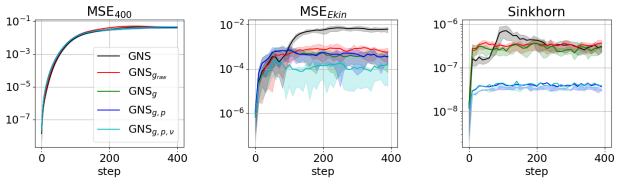

Figure 3. Ablations on RPF 2D with GNS-10-128 over the simulation length. Adapted from Fig. 26 in Appendix G.3.

4.2. SPH Relaxation

This section presents the results of our SPH relaxation on its own, and also in combination with the proposed external force treatment. We divide the discussion based on common characteristics of the datasets into periodic boundary cases, cases with wall boundaries, and free surface problems.

4.2.1. Periodic boundaries

Taylor-Green vortex (TGV). We did not expect the SPH relaxation to be very beneficial to the Taylor-Green vortex cases because (a) the trajectories are rather short with 125 and 60 steps in the 2D and 3D cases, respectively, and also (b) TGV represents a decaying problem, making it less prone to clustering in later stages of the trajectory. But according to Table 1, we get a consistent improvement of the position error MSE$_{400}$ of $\sim 5\%$ and significant Sinkhorn improvements on the 2D and 3D datasets.

Viscous term. In addition to external force subtraction, we found it beneficial to use the pressure (p) and viscous (ν) terms during relaxation, termed GNS$_{p,\nu}$. Viscosity, which manifests itself in shearing forces, in general, refers to the idea that if two fluid elements are close to each other but move in opposite directions, then they should both decelerate. Thus, to apply viscosity, we need to again approximate velocities by finite differences between consecutive positions of particles.

Reverse Poiseuille flow (RPF). In Figs. 4 and 10, we show histograms over velocity magnitudes to quantify how the different RPF correction terms impact the dynamics. Firstly, the original GNS model loses its high-velocity components over time, resembling a diffusion process, which makes it more stable with respect to perturbations, but, at the same time, leads to wrong kinetic energy. Secondly, simply changing the training objective by removing the external force (see GNS$_g$) already mitigates the problem of missing high velocities. And by adding the viscous term, which is especially relevant in the shearing region, to the pressure gradient term, we almost perfectly recover the target velocity distribution. See Fig. 3 and Appendix G.3 for further details.

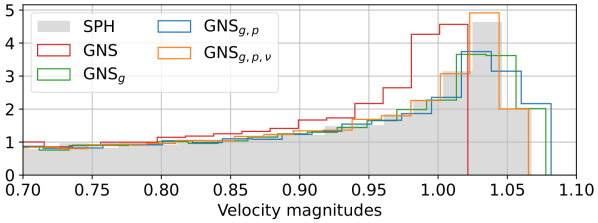

Figure 4. Velocity magnitudes histogram of 2D reverse Poiseuille flow after 400 rollout steps (averaged over all rollouts). Our GNS$_{g,p,\nu}$ matches the ground truth distribution of SPH.

4.2.2. Wall boundaries

A typical failure mode of learned solvers is that one or multiple particles penetrate what should be a solid wall, see top left part of Fig. 5 for LDC 2D and top part of Fig. 8 in Appendix A for DAM 2D. We solve this problem nearly completely with our SPH relaxations.

Relaxation at wall boundaries. The only part we have not discussed yet is how to ensure that particles do not leave the box by passing through the walls. We use the simple and effective approach laid out in the generalized wall boundary condition paper by Adami et al. (2012). The idea of this approach is to enforce the impermeability of the walls by setting the pressure of the dummy wall particles to the average pressure of their adjacent fluid neighbors, see Eq. (27) in Adami et al. (2012), and, thus, constructing a setting of zero pressure gradients normal to the walls. With this boundary condition implementation, we obtain the following one-step relaxation algorithm: 1. density computation for fluid particles, 2. pressure computation for fluid particles through the equation of state, 3. computation of pressure of wall particles via weighted summation over the pressure of adjacent fluid particles, and 4. evaluation of the pressure gradient term, which gives the forces used to integrate the momentum equation Eq. (5) through Eq. (6).

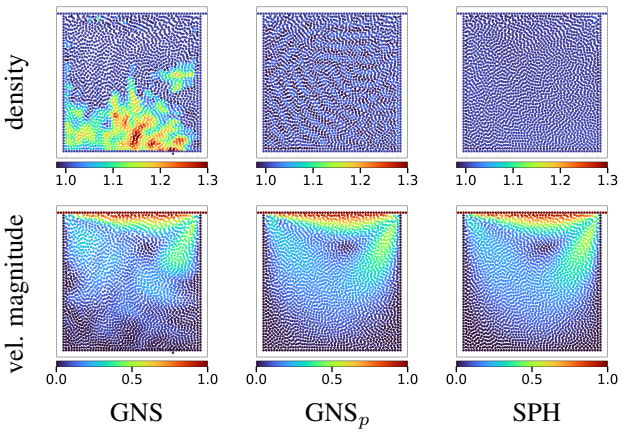

Figure 5. Density and velocity magnitude of 2D lid-driven cavity after 400 rollout steps (left to right): GNS, GNS$_p$, SPH. The colors in the first row correspond to the density deviation from the reference density; the system is considered physical within 0.98-1.02.

Lid-driven cavity (LDC). In the lid-driven cavity example, we observe that the learned model pushes particles away from the fast-moving lid into the lower half of the domain, which has profound consequences. On the one hand, the pressure at the bottom increases to an extent such that one or more particles gradually pass through the bottom wall. On the other hand, since too few particles reside close to the lid, the shearing forces are underrepresented, yielding a loss of kinetic energy, i.e., dynamics are lost. We fix both these issues with an SPH relaxation, forcing particles to be homogeneously distributed within the box, see Figs. 5 and 8. See Appendix G.2 for various hyperparameter sensitivity ablations on LDC 2D/3D and GNS/SEGNN. While tuning the parameters is crucial, once tuned, they seem to work fairly reliably.

4.2.3. Free surfaces

A major difference between dam break and the other datasets we benchmark is that in dam break we not only care about the particle distribution within the fluid, but also about the volume filled with fluid. The latter is the focus of this section, and it is reflected in the MSE$_{400}$ and Sinkhorn measures, but not in MSE$_{Ekin}$.

Dam break (DAM). Interestingly, by either our external force treatment or the SPH relaxation, we seem to fix the problem of the fan-like spreading of the wavefront. We interpret this as a confirmation that the reason for this failure mode is the high compression at the tip. However, fixing the high compression levels in the bulk fluid requires our SPH relaxation, which we run with as few as three steps. The GNS$_{g,p}$ setup then recovers the correct dynamics with a significantly higher precision as measured by the Sinkhorn divergence, but also the kinetic energy MSE, indicating that the fluid also evolves more physically. Regarding the fluid surface, if we carefully look at the height of the fluid in Figs. 6 to 9, we see that the GNS$_{g,p}$ case very closely resembles the ground truth. See Appendix G.1 for ablations.

4.3. SEGNN Results

We applied the same external force treatment and SPH relaxations to the SEGNN model (Brandstetter et al., 2022a) without further tuning of the neural SPH hyperparameters (see Appendix B) and summarize the results in Table 2. This comparison is useful not only for better comparability but also to show that proper SPH relaxation often depends more on the dataset than on the model – for example, moving the external force out of the 2D RPF case results in a 40 times lower kinetic energy error. However, in some cases, the GNS and SEGNN models behave quite differently. In most cases, SEGNN performs on par with GNS on long trajectories, with the notable SEGNN blowups on LDC 2D, DAM 2D, and RPF 3D. In particular, when we change the treatment of the external force in dam break without applying additional wall boundary conditions, we observe many particles falling through the bottom wall around step 200. Adding the relaxation with wall boundary conditions solves this problem, but investigating the qualitative differences between GNS and SEGNN would be an interesting future work. See Appendix G for our hyperparameter ablation.

	Model	MSE$_{400}$	Sinkhorn	MSE$_{\text{Ekin}}$
2D TGV	SEGNN	$4.0e-4$	$4.4e-7$	$3.9e-7$
	SEGNN$_p$	$3.8e-4$	$1.5e-8$	$2.8e-7$
2D RPF	SEGNN	$2.7e-2$	$3.3e-7$	$4.3e-3$
	SEGNN$_g$	$2.8e-2$	$3.3e-7$	$1.2e-4$
	SEGNN$_{g,p}$	$2.8e-2$	$3.5e-8$	$1.6e-4$
	SEGNN$_{g,p,\nu}$	$2.8e-2$	$3.8e-8$	$7.3e-4$
2D LDC	SEGNN	$7.6e-2$	$2.3e-3$	$9.1e+0$
	SEGNN$_p$	$1.8e-2$	$5.8e-7$	$1.6e-5$
2D DAM	SEGNN	$1.5e-1$	$3.4e-2$	$1.9e-2$
	SEGNN$_g$	$1.6e-1$	$2.1e-2$	$1.9e+1$
	SEGNN$_p$	$1.2e-1$	$9.4e-3$	$1.2e-2$
	SEGNN$_{g,p}$	$8.6e-2$	$4.9e-3$	$2.6e-3$
3D TGV	SEGNN	$4.2e-2$	$6.1e-6$	$2.4e-2$
	SEGNN$_p$	$4.1e-2$	$6.0e-7$	$2.7e-2$
3D RPF	SEGNN	$1.2e-1$	$1.0e-4$	$1.5e+3$
	SEGNN$_p$	$2.6e-2$	$1.3e-5$	$1.8e-2$
	SEGNN$_g$	$2.7e-2$	$2.6e-6$	$9.5e-3$
	SEGNN$_{g,p}$	$2.6e-2$	$7.9e-7$	$5.7e-3$
3D LDC	SEGNN	$3.3e-2$	$2.3e-5$	$1.7e-7$
	SEGNN$_p$	$3.3e-2$	$2.0e-6$	$1.8e-7$

Table 2. SEGNN-10-64 results. Same structure as Table 1.