LEARNING MESH-FREE DISCRETE DIFFERENTIAL OPERATORS WITH SELF-SUPERVISED GRAPH NEURAL NETWORKS

Lucas Gerken Starepravo, Georgios Fourtakas, Ajay B. Harish, Tianning Tang & Jack R. C. King
School of Engineering
The University of Manchester
Manchester, M13 9PL, United Kingdom
lucas.gerkenstarepravo@postgrad.manchester.ac.uk

Steven Lind
School of Engineering
Cardiff University
Cardiff, CF24 3AA, United Kingdom

ABSTRACT

Mesh-free numerical methods provide flexible discretisations for complex geometries, but classical discrete differential operators typically trade low computational cost for limited accuracy, or high accuracy for substantial per-stencil computation. We introduce a parametrised framework for learning mesh-free discrete differential operators using a graph neural network trained via polynomial moment constraints derived from truncated Taylor expansions. The model maps local geometric stencils directly to discrete operator weights. This demonstrates that neural networks can learn classical polynomial consistency conditions while retaining robustness to irregular neighbourhood geometry. The learned operators depend only on local geometry, are resolution-agnostic, and can be reused across particle configurations and governing equations. We evaluate the framework using standard numerical analysis diagnostics, showing improved accuracy over Smoothed Particle Hydrodynamics, and a favourable accuracy–cost trade-off relative to a representative high-order consistent mesh-free method in the moderate-accuracy regime. Applicability is demonstrated by solving the weakly compressible Navier–Stokes equations using the learned operators. An open-source implementation, including datasets and evaluation tools, is available at https://github.com/uom-complexfluids/nemdo.

1 INTRODUCTION

Partial differential equations (PDEs) are foundational to the modelling of physical systems across science and engineering. Yet, analytical solutions are rarely available for realistic problems of interest. Thus, numerical methods are often used to approximate the governing equations. Mesh-free numerical methods have significant potential, as discretisation relies solely on local connectivity information between collocation points without requiring topological connectivity. Nodes can be positioned flexibly to satisfy resolution requirements and conform to complex geometries (Liu & Gu, 2005; Vacondio et al., 2021).

Developed in the 1970s, Smoothed Particle Hydrodynamics (SPH) (Lucy, 1977) is today perhaps the most widely used mesh-free numerical method. In SPH, commonly employed in a Lagrangian formulation, particles are advected with the underlying velocity field, and differential operators are approximated via radial kernel interpolation over neighbouring particles. While SPH's original formulation had practical applications in solving astrophysical systems and computational fluid dynamics (Monaghan, 1994), it is inherently inconsistent, significantly limiting the accuracy and stability

that it can achieve (Liu & Gu, 2005; Quinlan et al., 2006). This motivated the development of more accurate numerical methods that enforce polynomial consistency through the solution of a local linear system (Schrader et al., 2010; Gross et al., 2020; King et al., 2020). However, consistent mesh-free methods are typically implemented in an Eulerian framework (Flyer et al., 2016; King, 2024), as they require solving dense local linear systems for each particle. In a Lagrangian setting, where the particle configuration evolves in time, this requires solving a local linear system for every particle at every time step, leading to substantial computational overhead. As a result, mesh-free simulations commonly face a trade-off between classical SPH kernels, which are computationally efficient but inconsistent and exhibit poor convergence, and consistency-corrected methods, which offer improved accuracy at the expense of significantly higher computational complexity.

In the current work, we introduce a self-supervised graph neural network framework that learns discrete *mesh-free* differential operators directly from irregular particle configurations, named Neural Mesh-Free Differential Operator (NeMDO). Rather than learning PDE solutions or problem-specific closures, NeMDO constructs local operator weights by learning polynomial consistency constraints derived from Taylor expansions, yielding operators with a clear mathematical foundation. Our results show that graph neural networks can learn these consistency constraints and predict operator weights with structural properties analogous to those of classical mesh-free discretisations.

The primary objective of NeMDO is to demonstrate that neural networks can approximate polynomial consistency constraints for mesh-free discrete differential operators, and provide a flexible framework for constructing mesh-free operators whose accuracy and computational cost can be systematically controlled. The resulting operators are strictly local, generalise across heterogeneous node configurations and resolutions, and are independent of any particular governing equation. We validate the learned operators using established numerical analysis tools, including convergence studies, modal response analyses, with additional stability and ablation studies reported in the appendix. We further demonstrate applicability by solving the weakly compressible Navier–Stokes equations.

2 Background & Related Work

2.1 Classical Numerical Approximation of Differential Operators

Given the discretisation of a spatial domain Ω by a set of collocation points $\mathcal{P} := \{\mathbf{x}_i\}_{i=1}^N \subset \Omega \subset \mathbb{R}^d$, a general local discrete approximation of a differential operator acting on ϕ can be written as

$$L^D(\phi(\mathbf{x}_i)) = \sum_{j \in \mathcal{N}_i} (\phi(\mathbf{x}_j) - \phi(\mathbf{x}_i)) w_{ji}^D, \tag{1}$$

where L^D denotes the discrete approximation of the differential operator D, $\mathcal{N}_i$ is the local support (or computational stencil) associated with node i, and w_{ji}^D are the corresponding stencil weights. This formulation encompasses a broad class of numerical methods given appropriate stencils and indexing—including finite difference method (Lele, 1992), finite element method (Clough, 1990), SPH (Lucy, 1977), and mesh-free high-order methods such as the Local Anisotropic Basis Function Method (LABFM) (King et al., 2020)—with the specific discretisation determined by how the weights w_{ji}^D and neighbourhoods $\mathcal{N}_i$ are constructed.

The accuracy and utility of the discretised solution depends on the *convergence, consistency*, and *stability* of the numerical method. For a discretisation to be consistent, the discrete operator must reproduce the Taylor moments up to a prescribed order. In mesh-free methods, these consistency conditions are typically enforced by solving local reconstruction problems in each particle neighbourhood. Representative approaches include the reproducing kernel particle method (Liu et al., 1995), generalised moving least squares (Trask et al., 2017; 2018), the radial basis function-finite difference methods (Bayona & Kindelan, 2013; Bayona et al., 2015), and LABFM (King et al., 2020). While solving these local linear systems enforces polynomial consistency and can yield high-order convergence, these methods introduce high computational cost when computing stencil weights.

In SPH, the stencil weights are obtained with a compact isotropic kernel (Koschier et al., 2020). The most used SPH kernel functions are the B-spline functions (Schoenberg, 1946) and the Wendland functions (Wendland, 1995), where the error of the integral approximation is $\mathcal{O}(h^2)$ in a con-

tinuum (Monaghan, 2005)—where h is the smoothing length of the kernel—with convergence of $\mathcal{O}(h)$ or lower in practice (Vacondio et al., 2013; Ferrand et al., 2013; Fourtakas et al., 2019) due to discretisation error and particle disorder. A consistency analysis of discrete SPH indicates that gradient operators are zeroth-order accurate in h on disordered particles, and certain Laplacian formulations may diverge as resolution increases. Nonetheless, convergence is often observed within a finite resolution window and for sufficiently well-behaved particle configurations and low wavenumbers (Quinlan et al., 2006).

2.2 Machine Learning for Partial Differential Equations

Learning-based PDE solvers encompass a broad class of approaches that learn solution mappings directly from data, either at the level of global operators or local particle interactions. Neural operator methods aim to learn mappings between function spaces, approximating the solution operator of a PDE from paired input–output functions, with representative examples including DeepONets (Lu et al., 2021), Fourier Neural Operators (FNOs) (Li et al., 2021), and graph-based neural operators (Anandkumar et al., 2019). These models enable inference across varying discretisations and resolutions and, in some cases, transfer across related equations or parameter regimes (Subramanian et al., 2023), but their focus remains on learning global solution mappings rather than local numerical operators. Learning-based particle methods leverage the locality of particle discretisations, such as SPH, to learn particle-wise updates or local solution operators directly from solution trajectories (Li & Farimani, 2022; Toshev et al., 2023; 2024; 2025). While effective for specific flow classes, these approaches are typically trained end-to-end on time-dependent solution data, tying the learned representations closely to the governing equations and regimes observed during training.

Learning Low-Level Numerical Operators: A complementary line of work focuses on learning numerical discretisations or low-level operator components. Rather than approximating solutions directly, these approaches learn local numerical building blocks—such as derivative stencils or basis functions—while preserving the structure of established discretisation schemes. Examples include learning data-driven finite-difference stencils to enable resolution coarsening (Bar-Sinai et al., 2019; Zhuang et al., 2021; Kochkov et al., 2021), extending these to finite-volume methods (de Romémont et al., 2024), and incorporating localised kernels into neural operators to improve accuracy (Liu-Schiaffini et al., 2024). Related architectures, such as PDE-Net (Long et al., 2018) and GMLS-Nets (Trask et al., 2019), can also learn local differential operator actions from solution data for system identification or forecasting, but do not explicitly construct reusable discrete operators as functions solely from stencil geometry. Despite these advancements, the ability of such models to approximate differential fields is often restricted to the spectral characteristics and functional forms present in the training distribution.

More recently, Choi et al. (2025) proposed data-driven finite element methods (DD-FEM), which replaces classical polynomial bases with locally learned basis functions. This formulation enables reuse across geometries, mesh types, and boundary conditions, highlighting the potential of learning reusable numerical building blocks that remain compatible with classical discretisation principles. The framework presented in the present manuscript is closely aligned with this operator-level perspective, but targets the learning of mesh-free discrete differential operators rather than local basis functions in mesh-based discretisations.

3 Method

In this section, we introduce NeMDO (Neural Mesh-Free Differential Operator), a new approach for computing weights used in discrete differential operators in disordered mesh-free simulations. Rather than computing these weights with a smoothing kernel or by solving per-particle linear systems, we formulate operator construction as a learning problem. We employ our framework to predict discrete operator weights based solely on the relative positions of neighbouring particles, enabling local mesh-free differential approximations.

Learning Mesh-Free Discrete Differential Operators: Given a discretised domain with collocation points $\mathcal{P}$, we associate to each point $\mathbf{x}_i$ a local neighbourhood $\mathcal{N}_i := \{\mathbf{x}_j \in \mathcal{P} : ||\mathbf{x}_{ji}||_2 \leq h\}$. Our objective is to construct a local discrete approximation L^D of a differential operator D acting on ϕ, such that the operator evaluation at $\mathbf{x}_i$ is approximated by a weighted sum over neighbouring

samples

$$L^D(\phi(\mathbf{x}_i)) = \sum_{j \in \mathcal{N}_i} (\phi(\mathbf{x}_j) - \phi(\mathbf{x}_i)) w_{ji}^D(\theta), \tag{2}$$

where $w_{ji}^D(\theta)$ are the trainable weights parametrised by θ, conditioned on the stencil geometry. We model the mapping from neighbourhood geometry to operator weights using a learnt function shared across all stencils. The learnt mapping does not depend on the field samples $\phi(\mathbf{x}_i)$, the governing equation, or the global simulation domain. Instead, it defines a reusable local operator constructor that can be applied consistently across particle distributions, resolutions, and governing equations.

Local Operator Parametrisation: For each local neighbourhood $\mathcal{N}_i$ a graph is defined $\mathcal{G}_i := (\mathcal{V}_i, \mathcal{E}_i)$, where the node set $\mathcal{V}_i \equiv \mathcal{N}_i$ consists of the particle i and its neighbours. The edge set $\mathcal{E}_i := \{(i,j), (j,i) : j \in \mathcal{N}_i \setminus \{i\}\}$ defines star-shaped, bidirectional edges between the central particle and its neighbours. In contrast to dense radius graph seen in particle-based learning approaches (Li & Farimani, 2022), this star-shaped construction yields linear complexity in the neighbourhood size, and the graph connectivity is directly constructed from the neighbourhood lists already computed in standard mesh-free methods, requiring no additional spatial searches or geometric preprocessing. Node attributes encode the normalised relative position $\hat{\mathbf{x}}_{ji} := \mathbf{x}_{ji}/z_i,\ z_i := \max_{j \in \mathcal{N}_i} \|\mathbf{x}_{ji}\|_2$, between neighbour particle j and the central particle i, where $\|\cdot\|_2$ denotes the Euclidean norm.

The graphs $\mathcal{G}_i$ are processed by a shared parametric function f_θ, implemented by a graph neural network (GNN). The operator parametrisation is defined as

$$\{\hat{w}_{ji}^D(\theta)\}_{j \in \mathcal{N}_i} \leftarrow f_\theta\left(\mathcal{G}_i, \{\hat{\mathbf{x}}_{ji}\}_{j \in \mathcal{N}_i}\right), \tag{3}$$

The relative positions are first embedded into a latent representation using a neural network. Message passing is then performed over the stencil graph, and a final network maps the latent node representations to normalised discrete operator weights, $\{\hat{w}_{ji}(\theta)\}_{j \in \mathcal{N}_i}$, associated with the target differential operator. A schematic of the framework can be seen in Figure 1.

Translation invariance is enforced by encoding geometry through relative position vectors, while permutation invariance is achieved through symmetric aggregation operations over neighbouring nodes. Scale robustness across particle resolutions is introduced by normalizing relative distances with respect to the maximum distance between neighbouring particles and the central particle. The proposed architecture is not explicitly rotation-equivariant; instead, rotational robustness is obtained implicitly through training on neighbourhoods with diverse node configurations. More details of the architecture can be found in Appendix C.

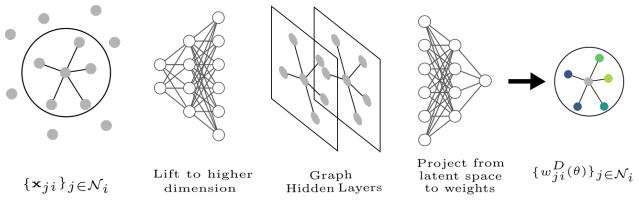


Figure 1: We learn the mapping from relative position of particles within a local neighbourhood, to a local set of weights that approximate differential operators. The learned operator is physics-agnostic and translation-invariant. The architecture consists of a lifting the relative positions to a higher dimension with a neural network, followed by a stack of message-passing graph layers, and a final output head that maps latent representations to operator weights.

Self-Supervised Learning with Polynomial Consistency Constraints: To learn the parameters θ, we employ a training objective derived from polynomial consistency conditions imposed by a truncated Taylor expansion. Consequently, the method does not require labelled target weights for individual stencils. Instead, the training objective is constructed by forming a normalised Taylor monomial basis directly from the normalised relative positions $\hat{\mathbf{X}}_{ji} := \left[\hat{x}_{ji}, \hat{y}_{ji}, \frac{\hat{x}_{ji}^2}{2}, \hat{x}_{ji}\hat{y}_{ji}, \frac{\hat{y}_{ji}^2}{2}, \frac{\hat{x}_{ji}^3}{6}, \dots\right]^T$. The normalised predicted operator weights $\{\hat{w}_{ji}^D(\theta)\}_{j \in \mathcal{N}_i}$ are combined with this basis to form discrete moment estimates, and deviations from the target operator moments—constant across stencils for a given differential operator—are penalised during training.

The learned discrete operator L^D approximates the action of the continuous differential operator D and is polynomially consistent up to order p, with the leading order error of $\mathcal{O}(s_i^{p+1-m})$, where m is the order of the target differential operator, and s is the average particle spacing.

For a given differential operator, the target moment vector $\mathbf{M}^D \in \mathbb{R}^a$ encodes the exact action of D on the Taylor monomial basis. As an illustrative example, consider the second-order approximation of the Laplacian operator in two dimensions. Using the monomial basis $\hat{\mathbf{X}}_{ji}$, the corresponding target moment vector is given by $\mathbf{M}^{\Delta} = [0,\ 0,\ 1,\ 0,\ 1]^T$. The polynomial consistency constraints are learned by minimising the squared residual of the predicted operator action on the Taylor monomials and the target moments, over all particles in the stencil,

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left\|\sum_{j\in\mathcal{N}_i}\hat{\mathbf{X}}_{ji}\,\hat{w}_{ji}^{D}(\theta) - \mathbf{M}^{D}\right\|_2^2, \tag{4}$$

where θ is optimised using stochastic gradient descent. A detailed derivation of the loss function is provided in Appendix A.

Data Generation: Training is performed on local particle neighbourhoods $\mathcal{N}_i$ sampled from synthetic disordered point cloud distributions obtained by applying stochastic perturbations to regular Cartesian stencils. Specifically, each grid node is independently displaced by a noise term sampled from a uniform distribution over $[-0.5\epsilon/s_u, 0.5\epsilon/s_u]$ per coordinate, where s_u denotes the average particle spacing of the Cartesian grid. This controlled perturbation strategy enables systematic analysis of operator sensitivity with respect to geometric disorder.

Unless stated otherwise, models are trained at a disturbance level of $\epsilon/s_u = 1.0$, which is significantly larger than the particle disorder typically encountered in practical mesh-free simulations. This choice ensures that the learned operators remain robust under severe geometric irregularity. In the Appendix D.3, we evaluate the effect of different levels of noise.

Operator Rescaling and Reuse: For a target differential operator of order m, the predicted weights are rescaled by z_i^{-m} to recover the correct physical dimensions, reflecting the m-th order spatial scaling of the operator and ensuring consistency across resolutions. The resulting weights are then used within standard mesh-free discretisations to approximate the action of the operator D on arbitrary fields.

Since operator construction depends only on local geometry, trained models can be reused across different resolutions, domains, and governing equations without retraining, provided the neighbourhood size $|\mathcal{N}_i|$ remains fixed. This enables learned operators to be deployed as drop-in numerical components within existing mesh-free solvers.

4 Experiments

The presented framework is assessed using a combination of polynomial reproduction residuals, derivative error analysis on a test function, modal response, and behaviour in fluid simulations. To assess the computational efficiency of the proposed framework, we report a wall-clock time analysis as a function of the L_2 error on a test function. Unless otherwise stated, all results are reported for a canonical configuration consisting of second-order gradient or Laplacian operators ($p = 2$). Details on hyper-parameters and the training dataset for each specific model can be found in Appendix C.

4.1 Polynomial Consistency and Convergence

We now verify that the learned operators satisfy the polynomial reproduction conditions imposed in the training loss and that this translates into accurate differential approximations. For each operator type, we evaluate the learned stencil weights on monomials up to degree p. For brevity, we only show the first derivative results for the x direction, noting that results in the y direction are analogous.

Table 1 reports moment residuals for learned NeMDO operators, averaged over independently perturbed neighbourhood realisations at fixed stencil size. For the gradient operator, residuals are consistently on the order of 10^{-5} across all first- and second-order monomials. The Laplacian operator exhibits larger residuals, on the order of 10^{-4}, reflecting the increased sensitivity of second-order

Table 1: Moment residuals for learned NeMDO operators. Mean absolute error (MAE) and standard deviation are reported for each monomial moment, averaged over independently perturbed neighbourhood realisations at fixed neighbourhood size. Residuals quantify the extent to which the learned discrete operators satisfy the imposed Taylor consistency constraints (smaller values are better). The superscripts in the learned operator identify the target operator.

Operator	**Metric**	x	y	$x^2/2$	xy	$y^2/2$
$\text{NeMDO}^{x}_{p=2}$	MAE	$8.84e-5$	$7.93e-5$	$5.77e-5$	$5.59e-5$	$4.74e-5$
	St.d.	$1.14e-4$	$1.04e-4$	$7.38e-5$	$7.35e-5$	$6.25e-5$
$\text{NeMDO}^{\Delta}_{p=2}$	MAE	$5.22e-4$	$5.09e-4$	$3.69e-4$	$4.09e-4$	$3.62e-4$
	St.d.	$6.91e-4$	$6.69e-4$	$4.94e-4$	$5.49e-4$	$4.77e-4$

derivatives to geometric perturbations, a behaviour also observed in classical high-order mesh-free and spectral discretisations (Lin et al., 2025). Overall, these results indicate that the polynomial consistency constraints can be learnt with a self-supervised framework and generalise for unseen stencil arrangements.

Next, we evaluate the learned operators on a smooth test function with known analytical derivatives. Relative L_2 errors between the predicted and exact derivatives are reported at varying average particle spacing s. Details of the convergence test setup are provided in Appendix E.3. The learned operators are compared against classical discretisations, including SPH kernels based on the quintic spline and Wendland C2, as well as a representative formally consistent method, LABFM.

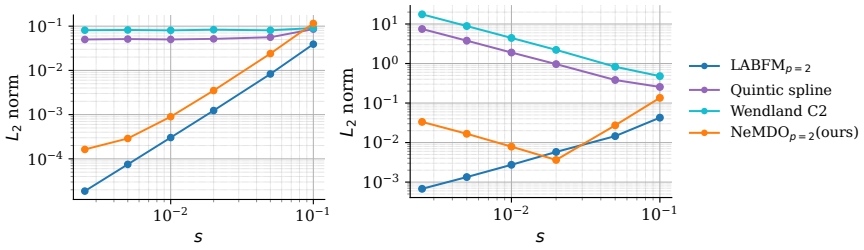


Figure 2: Convergence of the discrete x-derivative operator (*left*) and Laplacian (*right*) on a smooth test function with $\epsilon/s_u = 0.5$, noting that NeMDO was trained with $\epsilon/s_u = 1.0$. Relative L_2 error versus particle spacing s for the learned NeMDO operator, a second-order LABFM operator, and two uncorrected SPH kernels (quintic spline and Wendland C2).

Figure 2 reports the convergence behaviour of the derivative and Laplacian operators. All convergence results are computed at a particle disorder level of $\epsilon/s_u = 0.5$, since SPH convergence is known to degrade significantly at higher disorder levels (Quinlan et al., 2006). For the gradient, $\text{NeMDO}_{p=2}$ closely follows the second-order $\text{LABFM}_{p=2}$ scheme, exhibiting a decay rate consistent with second-order convergence as $s \to 0$. Across all resolutions, both the learned and LABFM operators substantially outperform the SPH kernel. For most mesh-free operators (including SPH and LABFM), achieving the expected convergence behaviour requires a sufficiently smooth underlying function, moderate particle disorder, and an adequate support size. Under these conditions, the limiting accuracy in consistent methods is typically bounded by the accuracy of the solution of the linear system (Shankar et al., 2014). NeMDO is not a formally consistent method; thus, the observed error floor is instead governed by the residual moment errors and standard deviation reported in Table 1.

For the Laplacian, LABFM exhibits the expected first-order convergence, achieving the lowest errors over the range of resolutions considered. The learned Laplacian displays an apparent second-order decay at coarse resolutions; however, this behaviour is only observed over a small range of resolutions, and while first-order convergence is expected, we observe second-order convergence because higher-order truncation error terms dominate the error when s is large (this effect is strongly dependent on the choice of ϕ). As resolution increases, the error saturates at a limiting value of $\mathcal{O}(10^{-3})$,

after which the error grows as $\mathcal{O}(s^{-1})$, similarly to traditional SPH formulations (Meng et al., 2025). This behaviour is consistent with the presence of a residual consistency error in the Laplacian operator, with similar divergence observed in formally consistent mesh-free methods and spectral SPH formulations, albeit at higher resolutions (King & Lind, 2022; Lin et al., 2025). Despite this limitation, the learned Laplacian remains significantly more accurate than uncorrected SPH kernels across the entire range of s.

Overall, these results suggest that learning operator weights to approximate polynomial consistency provides meaningful improvements over traditional SPH and exhibits similar behaviour to the formally consistent method within a finite resolution range.

4.2 Modal Response

Insight into the behaviour of the derivative operators can be gained from an analysis of the modal response.

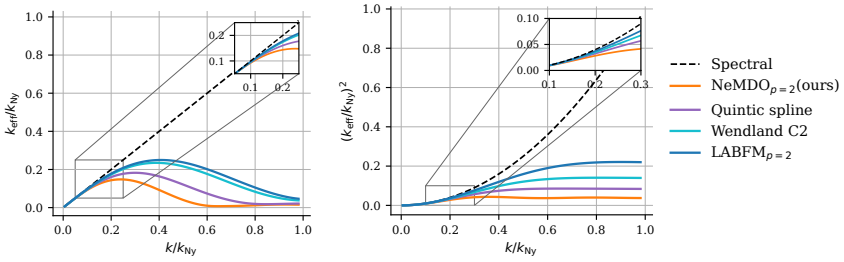


Figure 3: Resolving power of the gradient operator (*left*) and Laplacian (*right*) on disordered particle distributions with noise level $\epsilon/s_u = 1.0$. The horizontal axis shows the true wavenumber k normalised by the Nyquist wavenumber k_{Ny}, while the vertical axis shows the corresponding effective wavenumber $k_{\text{eff}}/k_{\text{Ny}}$. The dashed black line indicates the ideal response, corresponding to a spectral method.

Figure 3 shows the modal response of the different discretisations. For both the gradient and Laplacian operators, all methods recover the true modal response at low wavenumbers and progressively deviate as k/k_{Ny} increases, indicating larger errors for high wavenumber modes. For both operators, LABFM$_{p=2}$ remains closest to the spectral reference, followed by the Wendland C2 and quintic spline kernels, while the learned operator exhibits the lowest resolving power. Although the SPH kernels exhibit a stronger modal response than NeMDO, this spectral advantage is confined to sufficiently low resolutions and did not translate into smaller error during the convergence study in practice due to their pronounced sensitivity to node disorder and the resolution range investigated in Figure 2. The dependence of the stability properties of the operators on disordered nodes is explored in Appendix E.2, where we find that NeMDO yields the most stable and uniform spectral response compared to both SPH and LABFM. Combined, these results suggest a trade-off between resolving power and stability. The dependence of the modal response on stencil size is further examined in Appendix D.

4.3 Computational Cost and Accuracy

We now quantify the cost–accuracy trade-off of NeMDO relative to LABFM and SPH. For each configuration, we measure the wall-clock time required to compute stencil weights for the x-derivative on a fixed particle cloud and evaluate the resulting L_2 error of the test function in equation 26, detailed explanation about the setup is found in Appendix E.5. We report results for NeMDO$^1_{p=2}$, NeMDO$^2_{p=2}$, NeMDO$^3_{p=2}$, corresponding to increasing numbers of trainable parameters, to examine how computational cost and limiting accuracy scale with model capacity.

Figure 4 shows that NeMDO$^1_{p=2}$ achieves substantially lower error than the SPH kernels at comparable or lower cost, and yields a speedup of roughly $10\times$ over LABFM$_{p=2}$ up to its limiting error of $4e-3$. NeMDO$^2_{p=2}$ offers a more balanced trade-off, with a 1.5–$2\times$ speedup relative to LABFM$_{p=2}$ and up to two orders of magnitude lower error than the SPH baselines. Increasing the number of

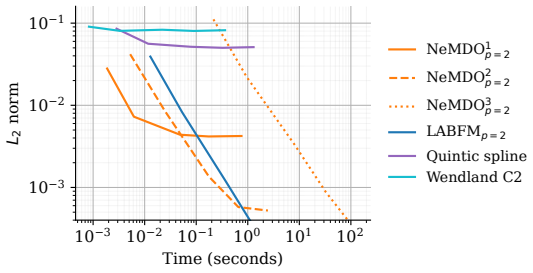


Figure 4: Evaluation of the trade-off between parameter count, wall-clock forward time, and accuracy for NeMDO and baseline kernel methods when computing stencil weights. The vertical axis shows the L_2 error of the x-derivative of the test function in equation 26 at a different resolutions; horizontal lines mark the resolution at which each operator converges (i.e., further refinement does not reduce this error). All NeMDO models are trained with geometric noise $\epsilon/s_u = 1.0$, while the convergence tests are performed at $\epsilon/s_u = 0.5$.

network parameters further in NeMDO$^3_{p=2}$ reduces the error floor again, at the expense of increased forward time.

It is important to note that all NeMDO models in Figure 4 were trained with geometric noise of magnitude $\epsilon/s_u = 1.0$, so they retain their accuracy for particle disorder up to this level. The figure reports results for $\epsilon/s_u = 0.5$, which is approximately the highest noise level at which the SPH kernels still exhibit convergence, while LABFM$_{p=2}$ (and other consistent methods) maintain the same convergence for particle disorder from $\epsilon/s_u = 0 - 1.0$ for second-order approximations (King et al., 2020). Additional experiments (Appendix D.3) indicate that training with milder noise makes the learning problem simpler, yielding lower errors for the same architecture. In this sense, the cost–accuracy curves in Figure 4 provide a conservative, worst-case characterisation of NeMDO's performance. Taken together, these results demonstrate a clear accuracy–computational cost trade-off enabled by NeMDO, which is not readily accessible to traditional mesh-free discretisation approaches.

4.4 Application to Partial Differential Equations

To assess the effectiveness of the learned operators in practical settings, we use them in mesh-free simulations of the compressible Navier–Stokes equations in an Eulerian unstructured node distribution. While SPH is generally implemented in Lagrangian schemes, particle clustering and instabilities can significantly affect the quality of the simulation (Price, 2012; Sun et al., 2018; Lyu et al., 2022). Our objective in this work is to assess the quality of the differential operator; thus, we constrain our investigation to Eulerian simulations. We consider the two-dimensional Taylor–Green vortex.

Weakly compressible SPH is known to generate pressure oscillations; thus, to improve numerical stability we dealiase the solution with a high-order filter at each time-step (Jameson et al., 1981). Furthermore, high-order collocated methods are known to frequently lead to spurious oscillations at small scales (Lamballais et al., 2011); thus, we also use a high-order filter in LABFM, and for our framework we train a hyperviscous operator GNN$^{\text{hyp}}_{p=4}$.

Governing Equations: We solve the two-dimensional compressible Navier–Stokes equations in conservative form

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial \rho u_k}{\partial x_k} &= 0 \\ \frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_k}{\partial x_k} &= -\frac{1}{\text{Ma}^2}\frac{\partial \rho}{\partial x_i} + \frac{1}{\text{Re}}\frac{\partial^2 u_i}{\partial x_k \partial x_k} \end{aligned} \tag{5}$$

where ρ is the density, u_i is the i-th component velocity, and Re is the Reynolds number, and Ma is the Mach number. The system is closed with a barotropic equation of state absorbed into the momentum equation.

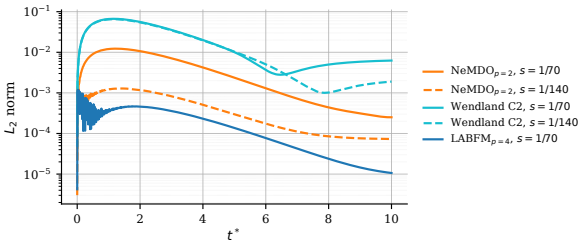


Figure 5: Relative root mean squared error of different operators with respect to the analytical solution (equation 27) for velocity. s indicates the average particle spacing, and t^* indicates characteristic time units of the flow.

Figure 5 reports the relative root-mean-square error of the Taylor–Green vortex with Reynolds number of 100 and Mach number of 0.1, computed using NeMDO$_{p=2}$, SPH with the Wendland C2 kernel, and LABFM$_{p=4}$ as a representative high-order mesh-free discretisation. The relative error is computed with respect to the analytical solution (Appendix E.4.1). Both SPH and NeMDO exhibit convergent behaviour with increasing resolution; however, the SPH error saturates at $\mathcal{O}(10^{-3})$, whereas NeMDO achieves errors approximately one order of magnitude smaller across the tested resolutions. The observed performance of NeMDO is consistent with the operator-level accuracy and stability diagnostics reported earlier, indicating that the learned operators can be directly embedded within a compressible flow solver and used as a standalone spatial discretisation without problem-specific tuning. In contrast, SPH required either high-order filtering or kernel correction to obtain stable solutions. Because NeMDO is trained without access to PDE solution data, the learned operators can be deployed in a zero-shot manner at the differential operator level, enabling direct reuse within PDE solvers without retraining.

Limitations: In its present form, each model is trained for a fixed stencil size $|\mathcal{N}_i|$ and does not natively support variable-size neighbourhoods at inference time. In practice, this constraint introduces limited additional bookkeeping. Allowing variable-size neighbourhoods would provide greater flexibility and simplify integration with adaptive particle distributions. Additionally, the achievable accuracy degrades as particle disorder increases unless the model capacity is correspondingly increased. This behaviour reflects the growing difficulty of satisfying polynomial moment constraints under highly irregular stencils and is not unique to NeMDO: classical SPH discretisations and consistency-corrected mesh-free methods similarly suffer from reduced accuracy and conditioning issues under strong node disorder at higher reproduction orders. These effects become especially relevant in fully Lagrangian settings, where particle densities and neighbourhood sizes can vary significantly over time. In such cases, the framework would likely need to be coupled with particle-shifting or regularisation strategies (Xu et al., 2009; Lind et al., 2012; Oger et al., 2016), as is standard practice in SPH-based methods, to maintain suitable stencil quality.

5 Conclusion

We presented a self-supervised learning framework for constructing mesh-free discrete differential operators on unstructured particle sets. Graph neural networks are trained to predict local stencil weights that approximate polynomial moment constraints derived from truncated Taylor expansions, yielding operators that are physics-agnostic, local, and robust to moderate particle disorder. Through moment analyses, wavenumber response, and Taylor–Green vortex simulations, we demonstrated that the learned operators can reproduce differential operators more accurately than standard SPH discretisations, while remaining competitive with traditional consistent methods for a range of resolutions. Furthermore, our method demonstrate a clear accuracy-computational cost trade-off, which is not readily accessible to traditional mesh-free differential operators.

Future work will focus on training objectives based on spectral and modal targets, and on different architectures with better robustness under highly disordered particle configurations. Another promising avenue is the integration of symbolic regression techniques with machine learning to discover compact kernels. More broadly, this work suggests that learning low-level numerical building blocks provides a principled pathway for integrating machine learning into mesh-free PDE solvers.

ACKNOWLEDGMENTS

Jack R. C. King is funded by the Royal Society via a University Research Fellowship (URF\R1\221290). We would like to acknowledge the assistance given by Research IT and the use of the Computational Shared Facility at The University of Manchester.

REFERENCES

S. Adami, X. Y. Hu, and N. A. Adams. A transport-velocity formulation for smoothed particle hydrodynamics. *Journal of Computational Physics*, 241:292–307, 2013. doi: 10.1016/j.jcp.2013.01.043.

A. Anandkumar, K. Azizzadenesheli, K. Bhattacharya, N. Kovachki, Z. Li, B. Liu, and A. Stuart. Neural operator: Graph kernel network for partial differential equations. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2019. URL https://openreview.net/forum?id=fg2ZFmXFO3.

Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019. doi: 10.1073/pnas.1814058116.

V. Bayona and M. Kindelan. Propagation of premixed laminar flames in 3D narrow open ducts using RBF-generated finite differences. *Combustion Theory and Modelling*, 17(5):789–803, 2013. doi: 10.1080/13647830.2013.801519.

V. Bayona, N. Flyer, G. M. Lucas, and A. J. G. Baumgaertner. A 3-d RBF-FD solver for modeling the atmospheric global electric circuit with topography (GEC-RBFFD v1.0). *Geoscientific Model Development*, 8(10):3007–3020, 2015. doi: 10.5194/gmd-8-3007-2015.

Y. Choi, S. W. Cheung, Y. Kim, P.-H. Tsai, A. N. Diaz, I. Zanardi, S. W. Chung, D. M. Copeland, C. Kendrick, W. Anderson, T. Iliescu, and M. Heinkenschloss. Defining foundation models for computational science: A call for clarity and rigor, 2025. URL https://arxiv.org/abs/2505.22904.

R. W. Clough. Original formulation of the finite element method. *Finite Elements in Analysis and Design*, 7(2):89–101, 1990. doi: 10.1016/0168-874X(90)90001-U.

G. de Romémont, F. Renac, J. Nunez, and F. Chinesta. A data-driven learned discretization approach in finite volume schemes for hyperbolic conservation laws and varying boundary conditions, 2024. URL https://arxiv.org/abs/2412.07541.

W. Dehnen and H. Aly. Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. *Monthly Notices of the Royal Astronomical Society*, 425(2):1068–1082, 2012. doi: 10.1111/j.1365-2966.2012.21439.x.

M. Ferrand, D. R. Laurence, B. D. Rogers, D. Violeau, and C. Kassiotis. Unified semi-analytical wall boundary conditions for inviscid, laminar or turbulent flows in the meshless SPH method. *International Journal for Numerical Methods in Fluids*, 71(4):446–472, 2013. doi: 10.1002/fld.3666.

N. Flyer, B. Fornberg, V. Bayona, and G. A. Barnett. On the role of polynomials in RBF-FD approximations: I. interpolation and accuracy. *Journal of Computational Physics*, 321:21–38, 2016. doi: 10.1016/j.jcp.2016.05.026.

B. Fornberg and E. Lehto. Stabilization of RBF-generated finite difference methods for convective PDEs. *Journal of Computational Physics*, 230(6):2270–2285, 2011. doi: 10.1016/j.jcp.2010.12.014.

G. Fourtakas, J. M. Dominguez, R. Vacondio, and B. D. Rogers. Local uniform stencil (LUST) boundary condition for arbitrary 3-d boundaries in parallel smoothed particle hydrodynamics (SPH) models. *Computers & Fluids*, 190:346–361, 2019. doi: 10.1016/j.compfluid.2019.06.009.

B. J. Gross, N. Trask, P. Kuberry, and P. J. Atzberger. Meshfree methods on manifolds for hydrodynamic flows on curved surfaces: A generalized moving least-squares (GMLS) approach. *Journal of Computational Physics*, 409:109340, 2020. doi: 10.1016/j.jcp.2020.109340.

A. Jameson, W. Schmidt, and E. Turkel. Numerical solution of the Euler equations by finite volume methods using Runge–Kutta time stepping schemes. In *Proceedings of the 14th Fluid and Plasma Dynamics Conference*. AIAA, 1981. doi: 10.2514/6.1981-1259.

J. R. C. King. A mesh-free framework for high-order direct numerical simulations of combustion in complex geometries. *Computer Methods in Applied Mechanics and Engineering*, 421:116762, 2024. doi: 10.1016/j.cma.2024.116762.

J. R. C. King and S. J. Lind. High-order simulations of isothermal flows using the local anisotropic basis function method (LABFM). *Journal of Computational Physics*, 449:110760, 2022. doi: 10.1016/j.jcp.2021.110760.

J. R. C. King, S. J. Lind, and A. M. A. Nasar. High order difference schemes using the local anisotropic basis function method. *Journal of Computational Physics*, 415:109549, 2020. doi: 10.1016/j.jcp.2020.109549.

D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118 (21):e2101784118, 2021. doi: 10.1073/pnas.2101784118.

D. Koschier, J. Bender, B. Solenthaler, and M. Teschner. Smoothed particle hydrodynamics techniques for the physics-based simulation of fluids and solids. In *Proceedings of the Eurographics Annual Conference*, 2020. doi: 10.2312/EGT.20191035.

E. Lamballais, V. Fortuné, and S. Laizet. Straightforward high-order numerical dissipation via the viscous term for direct and large eddy simulation. *Journal of Computational Physics*, 230(9): 3270–3275, 2011. doi: 10.1016/j.jcp.2011.01.040.

S. K. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103(1):16–42, 1992. doi: 10.1016/0021-9991(92)90324-R.

Y. Li, C. Gu, T. Dullien, O. Vinyals, and P. Kohli. Graph matching networks for learning the similarity of graph-structured objects, 2019. URL https://arxiv.org/abs/1904.12787.

Z. Li and A. B. Farimani. Graph neural network-accelerated Lagrangian fluid simulation. *Computers & Graphics*, 103:201–211, 2022. doi: 10.1016/j.cag.2022.02.004.

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

M. Lin, G. Fourtakas, and B. D. Rogers. A novel high-order spectral incompressible smoothed particle hydrodynamics (ISPH) scheme with an immersed boundary method (IBM). *Journal of Computational Physics*, 540:114264, 2025. doi: 10.1016/j.jcp.2025.114264.

S. J. Lind, R. Xu, P. K. Stansby, and B. D. Rogers. Incompressible smoothed particle hydrodynamics for free-surface flows: A generalised diffusion-based algorithm for stability and validations for impulsive flows and propagating waves. *Journal of Computational Physics*, 231(4):1499–1523, 2012. doi: 10.1016/j.jcp.2011.10.027.

G. R. Liu and Y. T. Gu. *An Introduction to Meshfree Methods and Their Programming*. Springer, Dordrecht, 1 edition, 2005. ISBN 978-1-4020-3228-8. doi: 10.1007/1-4020-3468-7.

M. B. Liu and G. R. Liu. Smoothed particle hydrodynamics (SPH): An overview and recent developments. *Archives of Computational Methods in Engineering*, 17(1):25–76, 2010. doi: 10.1007/s11831-010-9040-7.

W. K. Liu, S. Jun, and Y. F. Zhang. Reproducing kernel particle methods. *International Journal for Numerical Methods in Fluids*, 20(8–9):1081–1106, 1995. doi: 10.1002/fld.1650200824.

M. Liu-Schiaffini, J. Berner, B. Bonev, T. Kurth, K. Azizzadenesheli, and A. Anandkumar. Neural operators with localized integral and differential kernels, 2024. URL https://arxiv.org/abs/2402.16845.

Z. Long, Y. Lu, X. Ma, and B. Dong. PDE-net: Learning PDEs from data, 2018. URL https://arxiv.org/abs/1710.09668.

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

L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *The Astronomical Journal*, 82:1013–1024, 1977. doi: 10.1086/112164.

H.-G. Lyu, P. Sun, A. Colagrossi, and A.-M. Zhang. Towards SPH simulations of cavitating flows with an EoSB cavitation model. *Acta Mechanica Sinica*, 39, 2022. doi: 10.1007/s10409-022-22158-x.

Z.-F. Meng, P.-N. Sun, P.-P. Wang, B. C. Khoo, and A.-M. Zhang. High-order SPH: A review of the method and applications. *Archives of Computational Methods in Engineering*, 2025. doi: 10.1007/s11831-025-10346-0.

J. J. Monaghan. Simulating free surface flows with SPH. *Journal of Computational Physics*, 110 (2):399–406, 1994. doi: 10.1006/jcph.1994.1034.

J. J. Monaghan. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68(8):1703, 2005. doi: 10.1088/0034-4885/68/8/R01.

J. P. Morris, P. J. Fox, and Y. Zhu. Modeling low reynolds number incompressible flows using SPH. *Journal of Computational Physics*, 136(1):214–226, 1997. doi: 10.1006/jcph.1997.5776.

G. Oger, S. Marrone, D. Le Touzé, and M. de Leffe. Sph accuracy improvement through the combination of a quasi-Lagrangian shifting transport velocity and consistent ALE formalisms. *Journal of Computational Physics*, 313:76–98, 2016. doi: 10.1016/j.jcp.2016.02.039.

D. J. Price. Smoothed particle hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*, 231(3):759–794, 2012. doi: 10.1016/j.jcp.2010.12.011. Special Issue: Computational Plasma Physics.

N. J. Quinlan, M. Basa, and M. Lastiwka. Truncation error in mesh-free particle methods. *International Journal for Numerical Methods in Engineering*, 66(13):2064–2085, 2006. doi: 10.1002/nme.1617.

I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. part b. on the problem of osculatory interpolation. a second class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4:112–141, 1946. URL https://api.semanticscholar.org/CorpusID:125667988.

B. Schrader, S. Reboux, and I. F. Sbalzarini. Discretization correction of general integral PSE operators for particle methods. *Journal of Computational Physics*, 229(11):4159–4182, 2010. doi: 10.1016/j.jcp.2010.02.004.

V. Shankar, G. B. Wright, R. M. Kirby, and A. L. Fogelson. A radial basis function (rbf)-finite difference (fd) method for diffusion and reaction-diffusion equations on surfaces, 2014. URL https://arxiv.org/abs/1404.0812.

S. Subramanian, P. Harrington, K. Keutzer, W. Bhimji, D. Morozov, M. Mahoney, and A. Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior, 2023. URL https://arxiv.org/abs/2306.00258.

P. N. Sun, A. Colagrossi, S. Marrone, M. Antuono, and A. M. Zhang. Multi-resolution delta-plus-SPH with tensile instability control: Towards high reynolds number flows. *Computer Physics Communications*, 224:63–80, 2018. doi: 10.1016/j.cpc.2017.11.016.

A. P. Toshev, G. Galletti, J. Brandstetter, S. Adami, and N. A. Adams. Learning lagrangian fluid mechanics with e(3)-equivariant graph neural networks, 2023. URL https://arxiv.org/abs/2305.15603.

A. P. Toshev, J. A. Erbesdobler, N. A. Adams, and J. Brandstetter. Neural sph: Improved neural modeling of lagrangian fluid dynamics, 2024. URL https://arxiv.org/abs/2402.06275.

A. P. Toshev, T. Kalinov, N. Gao, S. Günnemann, and N. A. Adams. On learning quasi-Lagrangian turbulence. In *ICLR 2025 Workshop on Machine Learning Multiscale Processes*, 2025. URL https://openreview.net/forum?id=R1Lrk1EffC.

N. Trask, M. Perego, and P. Bochev. A high-order staggered meshless method for elliptic problems. *SIAM Journal on Scientific Computing*, 39(2):A479–A502, 2017. doi: 10.1137/16M1055992.

N. Trask, M. Maxey, and X. Hu. A compatible high-order meshless method for the Stokes equations with applications to suspension flows. *Journal of Computational Physics*, 355:310–326, 2018. doi: 10.1016/j.jcp.2017.10.039.

N. Trask, R. G. Patel, B. J. Gross, and P. J. Atzberger. GMLS-nets: A framework for learning from unstructured data, 2019. URL https://arxiv.org/abs/1909.05371.

R. Vacondio, B. D. Rogers, P. K. Stansby, P. Mignosa, and J. Feldman. Variable resolution for SPH: A dynamic particle coalescing and splitting scheme. *Computer Methods in Applied Mechanics and Engineering*, 256:132–148, 2013. doi: 10.1016/j.cma.2012.12.014.

R. Vacondio, C. Altomare, M. De Leffe, X. Hu, D. Le Touzé, Steven Lind, J.-C. Marongiu, S. Marrone, B. D. Rogers, and A. Souto-Iglesias. Grand challenges for Smoothed Particle Hydrodynamics numerical schemes. *Computational Particle Mechanics*, 8(3):575–588, 2021. doi: 10.1007/s40571-020-00354-1.

H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4:389–396, 1995. doi: 10.1007/BF02123482.

R. Xu, P. Stansby, and D. Laurence. Accuracy and stability in incompressible SPH (ISPH) based on the projection method and a new approach. *Journal of Computational Physics*, 228(18):6703–6725, 2009. doi: 10.1016/j.jcp.2009.05.032.

Z. Zheng and X. Li. Theoretical analysis of the generalized finite difference method. *Computers & Mathematics with Applications*, 120:1–14, 2022. ISSN 0898-1221. doi: https://doi.org/10.1016/j.camwa.2022.06.017. URL https://www.sciencedirect.com/science/article/pii/S0898122122002620.

J. Zhuang, D. Kochkov, Y. Bar-Sinai, M. P. Brenner, and S. Hoyer. Learned discretizations for passive scalar advection in a two-dimensional turbulent flow. *Physical Review Fluids*, 6:064605, 2021. doi: 10.1103/PhysRevFluids.6.064605.

APPENDIX STRUCTURE

The appendix is organised as follows: Appendix A explains in detail the construction of the self-supervised loss function, Appendix B contains detailed description of the standard numerical methods used in the current investigation; Appendix C describes in detail NeMDO architecture and hyper-parameters used in the manuscript; Appendix D contains model ablation, Appendix E provides additional tests not included in the main body of the manuscript and supplementary details regarding experiments conducted in the main body of the manuscript.

REPRODUCIBILITY STATEMENT

The complete implementation for solver development and neural network training, along with the datasets used for training and the scripts for convergence, stability, and resolving power analyses, is available as open-source at `https://github.com/uom-complexfluids/nemdo`. The repository is organized into modular directories for training and testing, including all benchmarked methods and hyperparameter configurations used in this study.

A THEORETICAL CONSTRUCTION OF LOSS FUNCTION

In this section, we explain in more detail the polynomial consistency constraints implemented for model training.

We consider a scalar field $\phi : \mathbb{R}^2 \to \mathbb{R}$ sampled on a discrete point cloud $\mathcal{P}$. For a given neighbourhood $\mathcal{N}_i$, where $\mathbf{D}(\phi)|_i$ denotes the vector of partial derivatives of ϕ at particle i, $\mathbf{D}(\phi)|_i := \left[\frac{\partial(\phi)}{\partial x}\Big|_i, \frac{\partial(\phi)}{\partial y}\Big|_i, \frac{\partial^2(\phi)}{\partial x^2}\Big|_i, \frac{\partial^2(\phi)}{\partial x \partial y}\Big|_i, \frac{\partial^2(\phi)}{\partial y^2}\Big|_i, \frac{\partial^3(\phi)}{\partial x^3}\Big|_i, ... \right]^T$, $\mathbf{X}_{ji}$ is the vector of Taylor monomials of the position of j relative to i, the multivariate Taylor expansion of ϕ about point i may be written compactly as

$$(\phi)_j = (\phi)_i \ + \mathbf{X}_{ji} \ \cdot \mathbf{D}(\phi)|_i \tag{6}$$

One can analyse the error in the discrete differential operator L^D by substituting the Taylor expansion into equation 1, obtaining

$$L_i^D(\phi) = \sum_{j \in \mathcal{N}_i} \mathbf{X}_{ji} \ \cdot \mathbf{D}(\phi)|_i \, w_{ji}^D \tag{7}$$

which, when expanded, is

$$\begin{aligned} L_i^D(\phi) = \ & \frac{\partial(\phi)}{\partial x}\Big|_i \sum_j x_{ji} w_{ji}^D + \frac{\partial(\phi)}{\partial y}\Big|_i \sum_j y_{ji} w_{ji}^D + \frac{\partial^2(\phi)}{\partial x^2}\Big|_i \sum_j \frac{x_{ji}^2}{2} w_{ji}^D + \frac{\partial^2(\phi)}{\partial x \partial y}\Big|_i \sum_j x_{ji} y_{ji} w_{ji}^D \\ & + \frac{\partial^2(\phi)}{\partial y^2}\Big|_i \sum_j \frac{y_{ji}^2}{2} w_{ji}^D + \frac{\partial^3(\phi)}{\partial x^3}\Big|_i \sum_j \frac{x_{ji}^3}{6} w_{ji}^D + \cdots \end{aligned} \tag{8}$$

For a target continuous differential operator D, polynomial consistency of order p requires that the discrete operator L_i^D reproduces the action of D exactly on all monomials of total degree at most p, i.e.

$$L_i^D(\phi) = D(\phi)|_i, \quad \forall \text{ polynomials of degree} \leq p. \tag{9}$$

For instance, to approximate $\partial_x \phi$, second-order polynomial consistency requires that the discrete moments satisfy

$$\sum_{j \in \mathcal{N}_i} x_{ji} w_{ji}^x = 1, \quad \sum_{j \in \mathcal{N}_i} y_{ji} w_{ji}^x = 0 \tag{10}$$

with all remaining moments associated with the second-order approximation also set to 0. In this case, the corresponding target moment vector is $\mathbf{M}^x = [1, \ 0, \ 0, \ 0, \ 0]$. When these conditions are satisfied, the resulting discrete operator approximates $\partial_x \phi$ with second-order accuracy for sufficiently smooth ϕ, yielding a truncation error of order $\mathcal{O}(s^2)$.

In our framework, we approximate polynomial consistency by minimizing the error between predicted and target moments over all particles

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left\| \sum_{j \in \mathcal{N}_i} \hat{\mathbf{X}}_{ji} \, \hat{w}_{ji}^D(\theta) - \mathbf{M}^D \right\|_2^2, \tag{11}$$

Thus, the proposed method does not require individual labels for each computational stencil. One can compute the training operator moments directly from the inputs and the predicted weights, and compute the loss based on the fixed target moments.

B STANDARD NUMERICAL METHODS

In this section, we explain SPH and LABFM in detail.

B.1 SMOOTHED PARTICLE HYDRODYNAMICS OPERATORS

In this work we use two standard SPH kernels: the quintic spline (Schoenberg, 1946) and the Wendland C2 kernel (Wendland, 1995). Both have been widely adopted in CFD-oriented SPH simulations (Adami et al., 2013; Sun et al., 2018). We summarise the kernel definitions, support choices, and discrete operators used in this investigation.

For gradients, the following form of the first-order gradient operator is generally used:

$$w_{ji}^{\nabla} = \nabla_i W(\mathbf{x}_{ji}, h) V_j \tag{12}$$

When computing the Laplacian with SPH operators, we employ the Morris operator (Morris et al., 1997):

$$w_{ji}^{\Delta} = \frac{-2\,\mathbf{x}_{ji}}{||\mathbf{x}_{ji}||} \cdot \nabla_i W_{ji} V_j \tag{13}$$

B.2 SMOOTHING KERNELS

Quintic spline: The quintic spline kernel has support $r \in [0, 3]$ and its given by

$$W^{\text{quintic}}(r) = \sigma \begin{cases} (1-r)_+^5 - 6\left(\frac{2}{3}-r\right)_+^5 + 15\left(\frac{1}{3}-r\right)_+^5, & 0 \le r < 1, \\ (1-r)_+^5 - 6\left(\frac{2}{3}-r\right)_+^5, & 1 \le r < 2, \\ (1-r)_+^5, & 2 \le r < 3, \\ 0, & r \ge 3, \end{cases} \tag{14}$$

where r is given by $||\mathbf{x}_{ji}||_2/h$, σ is a normalisation constant that depends on the system's dimension, in two dimensions $\sigma = \frac{7}{478\pi h^2}$ (Liu & Liu, 2010). We set $h = 1.5s$, which yields approximately 60-65 neighbours for the quintic spline.

Wendland C2: The Wendland C2 kernel has a support of $r \in [0, 2]$ and is defined as

$$W^{\text{WC2}}(r) = \sigma(1-r)^4(1+4r). \tag{15}$$

In two dimensions, $\sigma = \frac{7}{\pi h^2}$. We set $h = 1.5s$, which yields approximately 25-30 neighbours for the Wendland C2.

B.3 THE LOCAL ANISOTROPIC BASIS FUNCTION METHOD (LABFM)

In this work, we use the LABFM as an exemplar high-order mesh-free method, noting that it shares characteristics (structure of computational stencil, form of discrete operator, and local linear system) with other high-order mesh-free methods (e.g. generalised finite difference method (Zheng & Li, 2022), GMLS (Trask et al., 2017), radial basis function finite difference (Shankar et al., 2014)). Indeed, in the specific case where LABFM is constructed with Taylor monomials as a basis, the linear systems are identical to those in GMLS. For a comprehensive description and derivation of LABFM, we refer the reader to (King et al., 2020).

In LABFM, a general discrete operator is defined to approximate differential operators as shown in equation 1. The weights w_{ji}^D are given by a weighted sum of anisotropic basis functions (ABFs),

$$w_{ji}^D = \mathbf{W_{ji}} \cdot \boldsymbol{\Psi}_{\boldsymbol{i}}^{\boldsymbol{D}} = \mathrm{W}_{ji}^1 \Psi_{i,1}^D + \mathrm{W}_{ji}^2 \Psi_{i,2}^D + \mathrm{W}_{ji}^3 \Psi_{i,3}^D + ... \tag{16}$$

in which the vector $\mathbf{W_{ji}}$ are the ABFs, and $\mathbf{\Psi_i^D}$ is a coefficient vector. The coefficient vector is obtained by solving the following linear system

$$\mathbf{A_i} \boldsymbol{\Psi}_{\boldsymbol{i}}^{\boldsymbol{d}} = \mathbf{M^D} \tag{17}$$

For a given support region, $\mathbf{A_i}$ is given by:

$$\mathbf{A_i} = \sum_{j \in \mathcal{N}_i} \mathbf{X_{ji}} \otimes \mathbf{W_{ji}}, \tag{18}$$

where the rank of $\mathbf{A_i}$ is given by a,

$$a = \frac{p^2 + 3p}{2} \quad \text{and} \quad a = \sum_{m=2}^{m=p+1} \frac{m(m+1)}{2} \tag{19}$$

in 2 and 3 dimensions, respectively.

B.3.1 Anisotropic Basis Functions

To construct the ABFs, bivariate Hermite polynomials are combined with a radial basis function (RBF). The q-th element of $\mathbf{X}_{ji}$, which is the vector of Taylor monomials, is proportional to $x_{ji}^a y_{ji}^b$. Thus, the q-th ABF is defined as

$$W_{ji}^q = \frac{\psi\left(||\mathbf{x}_{ji}||_2 / h_i\right)}{\sqrt{2^{a+b}}} H_a\left(\frac{x_{ji}}{h_i\sqrt{2}}\right) H_b\left(\frac{y_{ji}}{h_i\sqrt{2}}\right) \tag{20}$$

where H_a is the a-th order univariate Hermite polynomial (of the physicists kind), and ψ is an RBF. In this work, the Wendland C2 kernel is used, following King & Lind (2022).

B.3.2 Global Operator Construction

To solve PDEs and evaluate the stability of the local operators we construct a linear global discrete operator, $\mathbf{G}^D$ (an $N \times N$ matrix), from the local operators L_i^D. The i-th row of $\mathbf{G}^D$ is the rearrangement of the local operator L_i^D, which is achieved by setting the elements of $\mathbf{G}^D$ as

$$\begin{aligned} G_{i,j}^D &= w_{j,i}^D \quad \forall j \neq i \\ G_{i,i}^D &= -\sum_j w_{ji}^D. \end{aligned} \tag{21}$$

The discretised form of the global system is given by

$$\mathbf{G}^D \mathbf{\Phi} = \mathbf{F}, \tag{22}$$

where $\mathbf{F}$ is the source vector.

C NeMDO Architecture and Hyper-parameters

The architecture developed in this work uses a multi-layer perceptron (MLP) to lift the relative normalised positions of the nodes to a high-dimensional latent space, followed by graph message passing layers, and a final MLP mapping the latent representations to operator weights.

$$\mathbf{v}_j^0 = \mathrm{MLP}_\theta^{\text{Emb}}\left(\hat{\mathbf{x}}_{ji}\right), \quad \mathrm{MLP}_\theta^{\text{Emb}} : \mathbb{R}^d \to \mathbb{R}^{F_h}, \quad \forall j \in \mathcal{N}_i \tag{23}$$

$$\hat{w}_{ji} = \mathrm{MLP}_\theta^{\text{Out}}(\mathbf{v}_j^L), \quad \mathrm{MLP}_\theta^{\text{Out}} : \mathbb{R}^{F_h} \to \mathbb{R}, \quad \forall j \in \mathcal{N}_i \tag{24}$$

where $\mathbf{v}_j^0$ and $\mathbf{v}_j^L$ are the node features at the input and output of the graph layers, respectively. The number of dimensions of the system is given by d and the number of hidden features per node after encoding is denoted as F_h.

We define L graph layers as

$$\mathbf{v}_j^l = \mathrm{MLP}_\theta^{l,\text{Upd}}\left(\mathbf{v}_j^{l-1}, \oplus_{k \in \mathcal{Q}_j} \mathrm{MLP}_\theta^{l,\text{Msg}}\left(\mathbf{v}_k^{l-1}\right)\right), \quad l = 1, ..., L, \quad \forall j \in \mathcal{N}_i, \tag{25}$$

where $\mathrm{MLP}_\theta^{l,\text{Upd}}$ and $\mathrm{MLP}_\theta^{l,\text{Msg}}$ are the update and message networks at layer l, $\mathbf{v}_j^l$ denotes the node features of node j, $\mathcal{Q}_j \subset \mathcal{N}_i$ is the set of nodes connected to node j, and $\oplus$ is a differentiable, permutation-invariant aggregation operator. In our implementation, $\oplus$ is realised as an attention-weighted aggregation based on cross-graph matching (Li et al., 2019). All MLPs use hyperbolic tangent activation functions. Models are trained using the adaptive moment estimation (Adam) optimiser.

Below we summarise the NeMDO architectures used in the different experimental sections.

Table 2: NeMDO architectures used in the main experiments.

Operator	$\lvert\mathcal{N}_i\rvert$	# Parameters	F_h	MLP Hidden Layers (Emb/Out/Msg/Upd)	Graph Layers	ϵ/s
NeMDO$^{x}_{p=2}$	35	2.2M	256	3	3	1.0
NeMDO$^{\Delta}_{p=2}$	35	2.2M	256	3	3	1.0

Table 3: NeMDO architectures used in the cost–accuracy experiments.

Operator	$\lvert\mathcal{N}_i\rvert$	# Parameters	F_h	MLP Hidden Layers (Emb/Out/Msg/Upd)	Graph Layers	ϵ/s
NeMDO$^{1}_{p=2}$	10	11.1k	32	1	2	1.0
NeMDO$^{2}_{p=2}$	15	46.3k	64	1	2	1.0

Main experiments. For the polynomial consistency and convergence (Section 4.1), modal response (Section 4.2), Taylor–Green vortex simulations (Section 4.4), and stability (Appendix E.2), we use a single architecture for both the x-derivative and Laplacian operators, denoted NeMDO$^{x}_{p=2}$ and NeMDO$^{\Delta}_{p=2}$. Their hyper-parameters are listed in Table 2. In the computational cost-accuracy plot (Figure 4), this configuration is referred to as NeMDO$^{3}_{p=2}$. The models presented in table 2 were trained with approximately 7 million neighbourhood graphs, with about 2 million samples used for validation and 1 million reserved for testing, yielding a total of roughly 10 million neighbourhoods. The models were trained with a starting learning rate of $1e-5$ and a plateau scheduler for a total of 2,000 epochs.

Ablations: For the ablation studies on Taylor truncation and stencil size (Appendix D), we fix the network architecture and vary only the reproduction order $p \in \{2, 3\}$ and the number of neighbours $|\mathcal{N}_i| \in \{10, 25, 50, 100\}$. All models in this group share the same parameter count and embedding dimension. The models have 183.7k trainable parameters, all MLPs (encoder, decoder, message and update) only have 1 hidden layer, the GNN has 2 graph layers, and all models with an embedding of size 128, with particle disorder of $\epsilon/s = 1.0$, trained for 1,000 epochs, with a starting learning rate of $1e-4$ with a plateau scheduler. The models presented in the ablations were trained with approximately 115k neighbourhood graphs, with about 33k samples used for validation and 16k reserved for testing, yielding a total of roughly 164k neighbourhoods.

Cost–accuracy experiments: For the computational cost and accuracy study in Figure 4, we use smaller-capacity models to probe the trade-off between parameter count, runtime, and limiting error. The architectures are summarised in Table 3, all models were trained for 2,000 epochs. NeMDO$^{3}_{p=2}$ (Figure 4) corresponds to NeMDO$^{x}_{p=2}$ listed in Table 2. The models presented in the computational cost-accuracy experiments (Table 3) were trained with approximately 115k neighbourhood graphs, with about 33k samples used for validation and 16k reserved for testing, yielding a total of roughly 164k neighbourhoods.

D Ablation

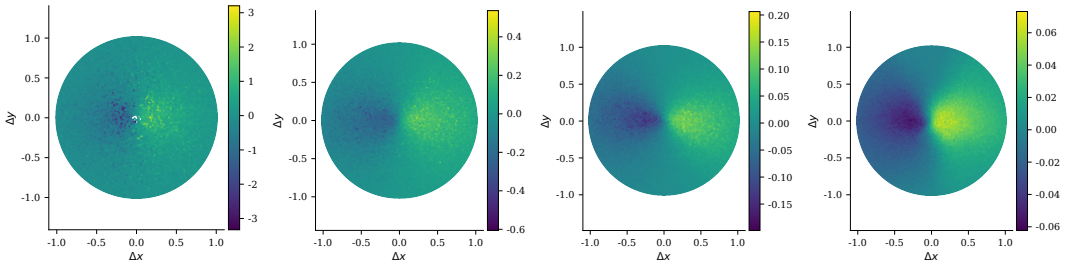
We perform 3 ablations: (i) varying Taylor truncation, (ii) varying stencil size, (iii) varying node disturbance.

D.1 Stencil Size

We first study how the accuracy of the learned operators depends on the stencil size $|\mathcal{N}_i|$. Table 4 reports the average moment error and standard deviation for x-derivative operators as $|\mathcal{N}_i|$ is varied for disordered node distribution, while keeping the network architecture and all other training parameters fixed.

Table 4: Polynomial moment errors for x-derivative operators varying stencil size $|\mathcal{N}_i|$.

$\lvert\mathcal{N}_i\rvert$	**Operator**	**MAE**	**St.d.**
10	NeMDO$_{p=2}$	$4.25e-3$	$8.04e-3$
25	NeMDO$_{p=2}$	$7.77e-4$	$1.04e-4$
50	NeMDO$_{p=2}$	$4.14e-4$	$4.88e-4$
100	NeMDO$_{p=2}$	$1.89e-3$	$1.45e-3$



(a) $n = 10,\ p = 2$ (b) $n = 25,\ p = 2$ (c) $n = 50,\ p = 2$ (d) $n = 100,\ p = 2$

Figure 6: Ablation results for varying stencil size n for node disorder of $\epsilon/s = 1.0$. Colours indicate value of weights. To generate the plots, multiple stencils with normalised relative positions have their weights predicted and plotted on top of each other.

Table 4 shows that increasing $|\mathcal{N}_i|$ from 10 up to 50 neighbours reduces the average moment error, this reflects the improved conditioning of the local Taylor system when more neighbours are included (indicated by the smaller weights in Figure 7). However, further increasing the stencil size to $|\mathcal{N}_i| = 100$ does not lead to additional gains. This suggests the presence of an optimal stencil size, beyond which particle noise, network capacity, and training dataset size, rather than the number of neighbours, become the dominant limiting factors. Consistent with the convergence results in Section 4.1, larger residuals (reported in table 4) translate into lower accuracy. Thus, models with larger moments residual show a larger error floor.

With respect to the modal response (Figure 7 (*left*)), decreasing the stencil size improves the modal response, with smaller stencils staying closer to the spectral line at large k/k_{Ny}. In Figure 7 (*right*), models with smaller stencils lie below the lines with larger stencils at low resolutions, highlighting the improving resolving power when decreasing the number of neighbours.

D.2 Taylor Truncation

We now vary the Taylor truncation by increasing the number of moments included in $\mathbf{M}^D$. In traditional high-order mesh-free numerical methods, this enforces polynomial consistency to a higher degree, which leads to more accurate solutions at equivalent resolution (i.e. improved resolving power). However, the local systems that must be solved become more poorly conditioned for higher orders of approximation; thus, these usually require larger computational stencils, and are more prone to instabilities (Shankar et al., 2014; King et al., 2020).

Figure 8 (*left*) shows the resolving power obtained with increased Taylor truncation order. Consistent with classical high-order numerical methods, including additional terms in the truncated expansion improves the resolving power. This improvement is also reflected in the convergence results shown in Figure 8 (*right*). At coarse resolutions, the model trained with $p = 3$ significantly outperforms the $p = 2$ model; however, it exhibits a larger limiting error. This behaviour reflects the more restrictive nature of higher-order approximations, for which a larger set of polynomial moments must be approximately satisfied. As a result, the learning problem becomes more constrained, leading to reduced limiting accuracy for a fixed model capacity.

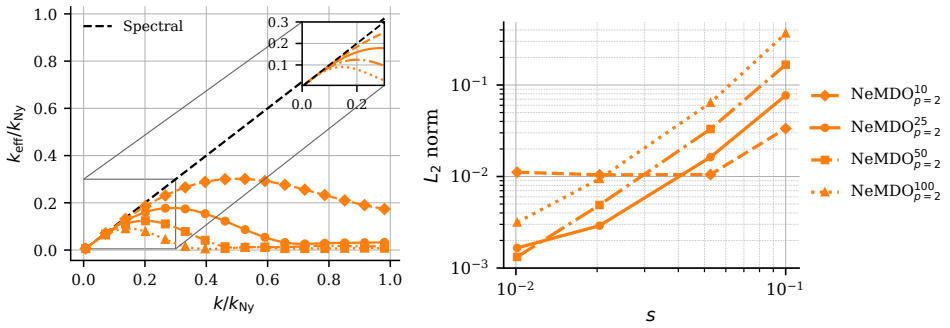


Figure 7: Impact of stencil size on learned x-derivative operators: effective wavenumber (*left*) and convergence (*right*) for a disordered node distribution with noise $\epsilon/s_u = 1.0$. The superscripts in the legend indicate the number of neighbours for each model.

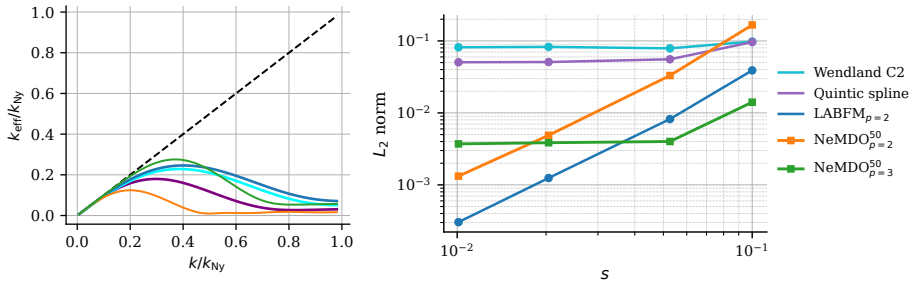


Figure 8: Impact of Taylor truncation on learned x-derivative operators: effective wavenumber (*left*) and convergence (*right*) for a disordered node distribution with noise $\epsilon/s_u = 1.0$. The superscripts in the legend indicate the number of neighbours for each model.

D.3 Kernels Sensitivity to Noise

In this section, we present supplementary ablation results. We keep the same architecture and number of neighbour nodes, and vary the particles disorder used for training.

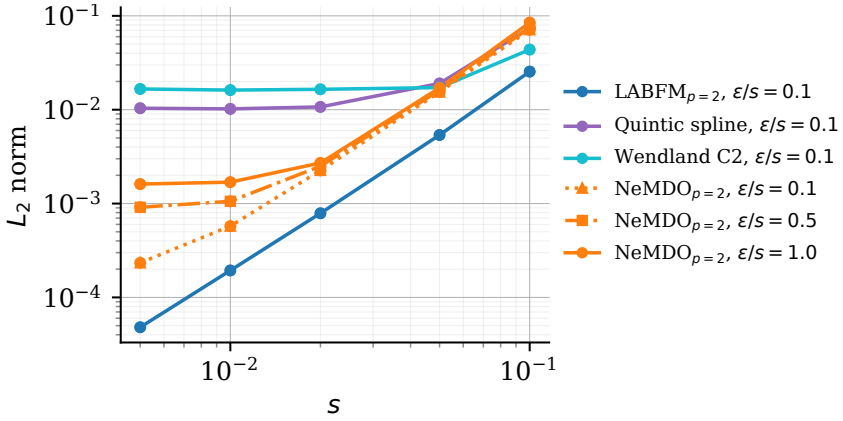


Figure 9: Convergence of the discrete x-derivative operator on a smooth test function (equation 26) with varying particle disturbance ϵ/s training and inference. Relative L_2 error of test function (equation 26) versus normalised spacing s. Learned operators are trained and do inference with the same level of particle disorder.

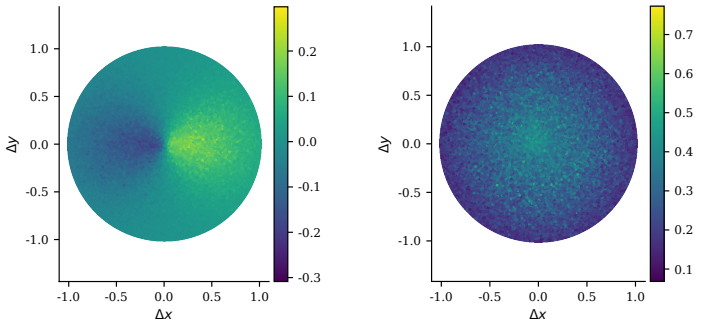
Figure 9 shows that varying the level of particle disturbance during training and inference affects the limiting accuracy of NeMDO. Increased disorder leads to a broader range of stencil geometries that the model must represent, resulting in higher residual errors. These results indicate that, for a fixed model architecture, more homogeneous particle distributions yield improved limiting accuracy. Similar behaviour is observed for SPH kernels and consistency-corrected methods, at higher approximation orders (Quinlan et al., 2006).

E Supplementary Analysis and Details on Experiments

In this section, we present supplementary results and provide additional details on the experiments discussed in the main body of the manuscript. Appendix E.1 presents qualitative results for the learned x-derivative and Laplacian operators. Appendix E.2 reports the stability analysis. Appendix E.3 provides additional details on the convergence study introduced in Section 4.1. Appendix E.4 describes the Taylor–Green vortex (TGV) setup and its analytical solution. Finally, Appendix E.5 details the computational cost evaluation conducted in this study.

E.1 Qualitative Analysis of Learned Operator

To assess the geometric structure of the learned discrete operators, we perform a qualitative analysis of the predicted stencil weights across multiple noisy neighbourhood realisations. Figure 10 visualizes the learned kernel corresponding different differential operators, where weights predicted for many independently perturbed neighbourhoods are overlaid in relative coordinate space.



(a) x-derivative (b) Laplacian

Figure 10: second-order normalised learned operators, colours indicate weight value. Models were trained with particle disturbance of $\epsilon/s = 1.0$ and inferred with the same noise.

Despite significant stochastic variation in local particle configurations, the learned operators exhibit coherent structures. In Figure 10, the x-derivative aggregated weight distribution displays approximate odd symmetry about the origin, $w(-x,-y) \approx -w(x,y)$, consistent with the structure of a first-derivative operator, while decaying in the transverse direction. This symmetry is not enforced explicitly by the architecture or loss, but emerges purely from the polynomial moment constraints used during training. The overlaid kernels further reveal a decay in weight magnitude with increasing distance from the central particle. For Laplacian operator, the learned kernels instead exhibit rotational invariance, with larger weights concentrated near the central particle and decreasing with distance. Together, these symmetry patterns and the consistent decay of weight magnitude away from the central particle mirror structural properties commonly observed in gradient and Laplacian kernels of classical mesh-free methods, including uncorrected SPH and polynomially corrected kernel formulations (Dehnen & Aly, 2012; King et al., 2020).

E.2 Stability

To analyse the stability of the discrete operators, we construct a global discrete derivative matrix $\mathbf{G}^D$ (see Appendix B.3.2). The eigenvalues of $\mathbf{G}^D$ provide insight into the stability properties of

the discretisation. Eigenvalues on the imaginary axis correspond to advective (translational) modes, while eigenvalues with nonzero real parts indicate growth or decay.

For convective derivatives (i.e. first-order spatial derivatives), the continuous operator is purely dispersive, with Fourier modes corresponding to translation and no amplification or attenuation. Accordingly, the eigenvalues μ of a stable discrete approximation should ideally lie on the imaginary axis, i.e. $\Re(\mu) = 0$ for all μ. In contrast, for the Laplacian operator, all eigenvalues should lie on the negative real axis, reflecting the decay of all modes due to diffusion (Fornberg & Lehto, 2011).

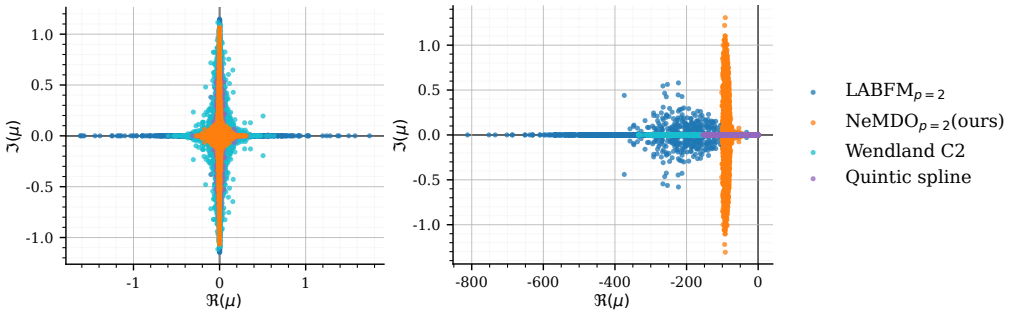


Figure 11: Normalised eigenvalue spectrum of the global x-derivative operator with node disturbance $\epsilon/s_u = 1.0$ and 2500 nodes. *Left*: x-derivative operators; *right*: Laplacian operators.

Figure 11 (*left*) shows the normalised eigenvalues of the discrete x-derivative operator on a noisy particle distribution. The spectrum of the learned operator is tightly clustered near the imaginary axis, with a comparable spread to the quintic spline, and lies closer to the imaginary axis than that of LABFM constructed with the same neighbourhood size $|\mathcal{N}_i|$ and the Wendland C2 kernel. Given that high-order discretisations are known to generate small-scale oscillations (Lamballais et al., 2011), it is notable that the learned operator—trained to approximate second-order polynomial consistency—exhibits stability properties comparable to the quintic spline. This suggests that the learned operator achieves more favourable stability characteristics per neighbour than both corrected and uncorrected kernel-based baselines.

For the Laplacian, no eigenmodes exhibit growth in time ($\Re(\mu) \leq 0 \ \forall \mu$) for any of the operators considered. NeMDO Laplacian displays a larger spread along the imaginary axis than the other methods; however, its eigenvalues are more tightly clustered overall, indicating more uniform damping of modes. In contrast, the Morris Laplacian (Morris et al., 1997) exhibits only real components, corresponding to purely diffusive behaviour, which likely contributes to its widespread use and robustness in SPH simulations. Overall, the learned operators demonstrate a more uniform spectral response across modes in disordered particle configurations for both first- and second-order derivatives, although a large dispersiveness is observed in the Laplacian. It should be noted that the results presented in this section are with particle disorder of $\epsilon/s_u = 1.0$, which represents the worst-case scenario regarding node distribution compared to standard simulations.

E.3 Test Function

We considered a square domain defined by $(x, y) \in [-0.5, 0.5]^2$. We define the test function

$$\phi(\tilde{x}, \tilde{y}) = 1.0 + (\tilde{x}\tilde{y})^4 + \sum_{n=1}^{6} (\tilde{x}^n + \tilde{y}^n), \tag{26}$$

where $\tilde{x} = x - 0.1453$ and $\tilde{y} = y - 0.16401$. This choice of test function with pseudo-random offset ensues asymmetry in the function, to prevent the masking of errors, which could cancel for a symmetric function (e.g. Fourier modes) (King et al., 2020).

E.4 Taylor–Green Vortex

This appendix describes analytical solution used to compute the relative error of the TGV (Appendix E.4.1), and qualitative results (Appendix E.4.2).

E.4.1 Analytical Solution and Error Metric

The analytical solution of the TGV for incompressible flow is given by:

$$\begin{aligned} u(x,y,t) &= -\exp\left(-\frac{8\pi^2}{\mathrm{Re}}\,t^*\right)\cos(2\pi x)\sin(2\pi y), \\ v(x,y,t) &= \quad \exp\left(-\frac{8\pi^2}{\mathrm{Re}}\,t^*\right)\sin(2\pi x)\cos(2\pi y), \end{aligned} \tag{27}$$

on the spatial domain $(x, y) \in [0, 1]^2$ equipped with periodic boundary conditions. In the equation above, u and v denote the velocity components in the x- and y-directions, respectively, which we use as a reference. In the experiments carried in this section, we used a Reynolds number of 100 and Mach number of 0.1. In the weakly compressible regime, the numerical discretisation error dominates the deviation introduced by compressibility effects; therefore, the incompressible solution can be used as a reference (King & Lind, 2022).

The error between the analytical solution and the predicted solution obtained with the learned discretisation is quantified using a *relative* root mean squared error (relative L_2 error) based on the velocity magnitude. Specifically, defining the exact and numerical speed as

$$U_{\mathrm{ex}}(x_i, y_i, t) = \sqrt{u_{\mathrm{exact}}(x_i, y_i, t)^2 + v_{\mathrm{exact}}(x_i, y_i, t)^2}, \qquad U_{\mathrm{num}}(x_i, y_i, t) = \sqrt{u_i(t)^2 + v_i(t)^2},$$

where the error is given by

$$L_2(t) = \sqrt{\frac{\sum_{i=1}^{N} \left(U_{\mathrm{num}}(x_i, y_i, t) - U_{\mathrm{ex}}(x_i, y_i, t)\right)^2}{\sum_{i=1}^{N} \left(U_{\mathrm{ex}}(x_i, y_i, t_0)\right)^2}}.$$

E.4.2 Qualitative Results

Figure 12 and Figure 13 shows snapshots of the velocity and pressure, respectively, of the TGV at different time units.

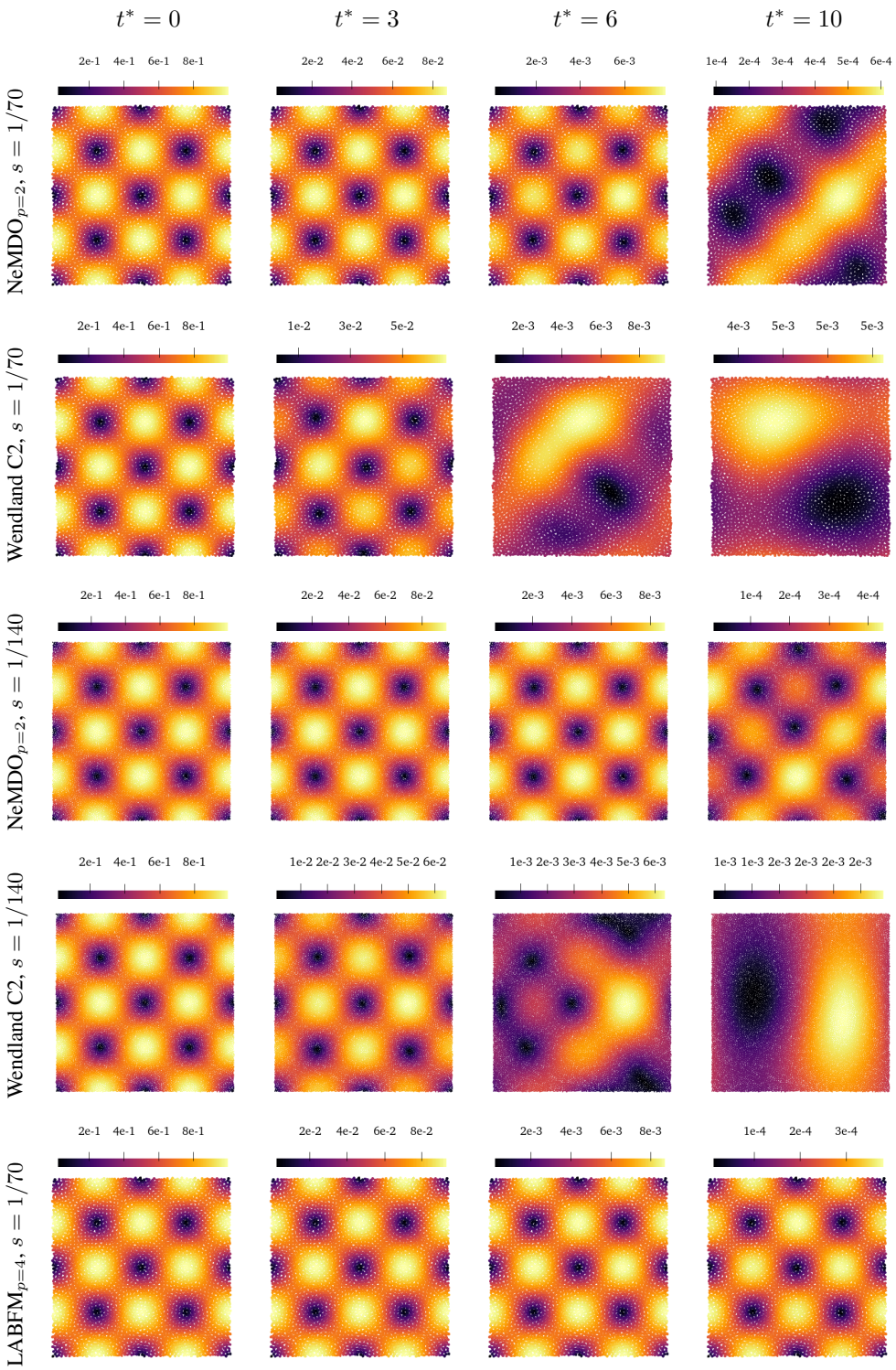


Figure 12: Velocity magnitude for the Taylor–Green vortex with different operators at different resolutions. Colour axis are rescaled at different time scales for better visualisation of the flow structures.

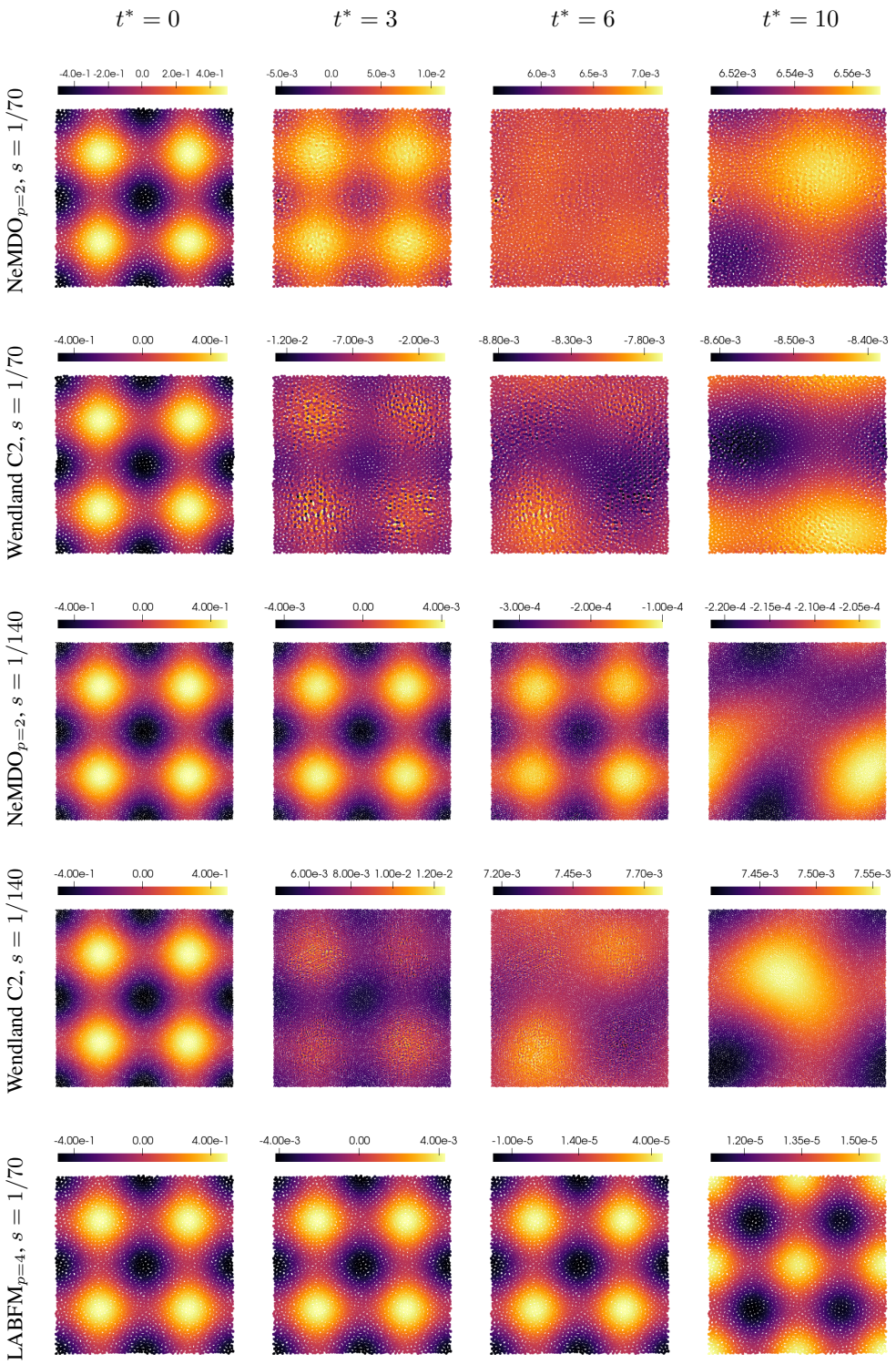


Figure 13: Relative pressure ($p - p_{ref}$) for the Taylor–Green vortex with different operators at different resolutions. Colour axis are rescaled at different time scales for better visualisation of the flow structures.

E.5 COMPUTATIONAL DETAILS

Providing a fully balanced performance comparison between fundamentally different approaches is inherently difficult. The estimates in Figure 4 should therefore be interpreted as indicative rather than absolute. All timings were obtained on a single Intel Core i9-14900F core (single thread). For NeMDO, we measure the wall-clock time of inference. For the SPH baselines, we measure the time required to compute weights with the kernels. For LABFM, we include the time to assemble and solve the resulting linear systems, and form the final stencil weights. The time measurements provided in Figure 4 represents the total time for each operator to predict the x-derivative weights for all nodes in a given resolution.

Most neural-network components in our framework are implemented using deep learning libraries optimised for GPU and batched execution, whereas small dense linear systems can be solved very efficiently on CPUs. As a result, the single-core CPU timings reported here are conservative for NeMDO and relatively favourable for LABFM. We do not report GPU or multi-threaded timings in this work; a systematic study of heterogeneous CPU–GPU execution and distributed inference is left for future work. The key takeaway from the present measurements is the relative ordering and scaling behaviour of the methods: NeMDO eliminates the per-stencil linear solves required by corrected-kernel schemes and exposes explicit architectural and stencil-size knobs to trade compute for accuracy within a single framework.