Stability Guided Neural Network For Hybrid Eulerian-Lagrangian Solvers

Parshad Sawant

18/5/2026

Abstract

Solver coupling induced instabilities is the primary problem faced by hybrid Eulerian-Langrangian solvers in vortex dominated flows. Current ML CFD methods target accuracy or speed, but numerical stability is handled by traditional correction methods which are typically threshold based or apply continuous corrections. This project proposes a Stability Guided Neural Network, a lightweight LSTM controller trained only on native solver failure trajectories, requiring no external DNS reference data. The SGNN monitors a 12-scalar diagnostic feature vector over a rolling 10-step window and predicts imminent numerical blow-up, enabling preemptive and intermittent application of classical stabilisation techniques only when necessary. Trained across four 2D periodic flow topologies (Kelvin–Helmholtz, Kolmogorov, Taylor–Green, and random vortex blobs), the controller achieves 0.98 AUC in blow-up prediction and extends stable simulation time by 1.56× to 2.24× across Reynolds numbers from 10,000 to 30,000, where the uncontrolled baseline solver fails deterministically. Because corrections are applied only when a risk threshold is crossed, the hybrid solver's conservation properties and spectral cascade are preserved during all stable phases. This proof-of-concept demonstrates that prediction of solver-specific instabilities is a tractable and computationally lightweight objective, complementary to existing accuracy-focused ML-CFD methods.

1 Introduction

Simulation of high-fidelity turbulent fluid flows, particularly at extended duration and high Reynolds numbers in custom hybrid solvers, is a fundamental problem in computational fluid mechanics (Pope, 2000; Sagaut, 2006). Hybrid Eulerian Lagrangian solvers offer an elegant advantage over traditional Eulerian approaches, by reducing numerical diffusion in the vortex dominated zones, (Cottet and Koumoutsakos, 2000) but are often limited by numerical instability before accuracy, especially at the coupling interface (Koumoutsakos, 2005).

The current trend in ML-CFD research is focused either on replacing traditional solvers with Physics-Informed Neural Networks (PINNs) (Raissi, Perdikaris and Karniadakis, 2019), or on increasing numerical accuracy via Solver-in-the-Loop (SiL) and learned closure methods (Um et al., 2020; Kochkov et al., 2021). This research proposes a Stability-Guided Neural Network (SGNN): an adaptive, learned, continuous neural netowrk positioned at the coupling interface of the hybrid solver. The SGNN is designed as a numerical stability controller. It identifies statistical precursors to solver blow-ups and intermittently applies classical stabilisation interventions only when necessary. Preliminary evidence suggests this approach may extend the stable integration time of under-resolved simulations without requiring the expensive high-resolution data typically needed for closure modelling. By adopting a diagnostic approach to ML-augmented CFD and focusing exclusively on globally aware stability, the SGNN offers distinct advantages. The self-supervised dataset required for training is computationally lightweight and highly robust across generalised flow regimes. Furthermore, because the SGNN operates intermittently, interference from flow-stabilisation filters is minimised, preserving higher overall physical fidelity.

Simulation of high-fidelity turbulent flows at high Reynolds numbers in hybrid Eulerian–Lagrangian solvers remains a fundamental challenge in computational fluid mechanics (Pope, 2000; Sagaut, 2006). Hybrid solvers have the complementary strengths of both frameworks. Eulerian methods efficiently resolve boundary layers and large-scale structure, while Lagrangian vortex methods reduce numerical diffusion in vortex-dominated regions. However, the bidirectional coupling interface introduces a class of instabilities not present in either solver alone, self-amplifying enstrophy feedback, Gibbs ringing from under-resolved particle, and aliasing at the particle-to-mesh projection step. These instabilities typically crash the solver before accuracy can be evaluated (Koumoutsakos, 2005; Cottet and Koumoutsakos, 2000).

The current trend in ML-augmented CFD addresses accuracy. Physics-Informed Neural Networks replace governing equations (Raissi, Perdikaris and Karniadakis, 2019), Solver-in-the-Loop methods learn continuous correction terms (Um et al., 2020), and learned interpolation methods accelerate spectral solvers by up to 80× while matching DNS accuracy (Kochkov et al., 2021). Adaptive stabilisation has received less attention. Closest to the present work, Ivagnes, Strazzullo and Rozza (2025) demonstrate that a reinforcement learning agent can adaptively select spectral filter parameters for turbulent

flows without DNS reference data, using physics-based reward signals alone. However, that framework applies corrections continuously.

This paper proposes a different approach. A Stability-Guided Neural Network (SGNN) designed as a **predictive, intermittent** numerical stability controller positioned at the coupling interface of the hybrid solver. The SGNN monitors temporal trajectories of solver diagnostics and identifies statistical precursors to blow-up events, applying classical stabilisation only when a learned risk threshold is crossed. Unlike Solver-in-the-Loop and closure methods, it is not trained to improve sub-grid accuracy, and requires no high-resolution DNS reference data. The training dataset is generated exclusively from the native solver's own failure trajectories. Because it operates as a conditional no-op during stable phases, it preserves the native solver's physical fidelity while extending its stability into regimes where the uncontrolled baseline deterministically fails. This framing, stability as an independent, learnable objective separate from accuracy, represents an unexplored direction in ML-CFD research.

2 Technical Context and Motivation

2.1 Technical Context and Research Gap

Numerical instability in unresolved fluid simulations are different from accuracy degradation, it is not a gradual loss of fidelity but an immediate, collapse of the numerical solution. In hybrid Eulerian Lagrangian solvers, this collapse is driven by the coupling interface instability and enstropy feedback. Neither traditional stabilising methods, nor learned accuracy correctors are designed to address this instability. Traditional heuristic methods are reactive, they cannot anticipate multi step blow up trajectories. Learned accuracy correctors require high resolution DNS data and apply continuous corrections that risk degrading physical accuracy during stable phases. Traditional method do not treat numerical instability as a temporally predictable, solver-specific phenomenon that can be identified from diagnostic precursors and corrected preemptively. The SGNN is designed to fill this gap. It is not a replacement for closure or accuracy models, but as a complementary stability layer that acts only when the solver's own trajectory indicates imminent failure.

2.2 Relationship to Standard Numerical Stability Approaches

Standard approaches to numerical stability in rely on Adaptive Mesh Refinement (Berger and Colella, 1989) and adaptive time stepping, while Vortex-in-Cell methods employ periodic remeshing and P3M corrections to manage particle density (Cottet and Koumoutsakos, 2000). These methods are reactive, responding to instantaneous field gradients or thresholds at a single timestep. This design is well suited for solvers whose instabilities are localised and instantaneously detectable. Hybrid Eulerian Lagrangian coupling instabilities are different. The coupling mechanism builds instability over multiple time steps before any instantaneous threshold corrects, by the time adaptive time stepping detects an anomaly, the solver is already within one or two steps of a divergence. In validation testing, the native hybrid solver diverged catastrophically between steps 2,200 and 3,200 even with standard stabilisation active, with way to recover the simulation once the instability was detected. The SGNN addresses this by learning the temporal trajectory toward failure rather than reacting to its instantaneous signature, identifying precursors up to 10 steps before the solver diverges catastrophically.

2.3 Relationship to Learned Accuracy Correctors

The current trends of ML CFD primarily focuses on accuracy by reducing numerical error or reconstructing sub grid physics (Kochkov et al., 2021; Um et al., 2020). The SGNN methodology is distinct by its self-supervised training protocol. It is trained only on numerical instability trajectories generated by the solver itself. By mapping statistical diagnostics to localised parameter interventions, it avoids the need for expensive DNS reference data (Raissi, Perdikaris and Karniadakis, 2019; Duraisamy, Iaccarino and Xiao, 2019).

The current trends in ML CFD uses neural networks to improve numerical accuracy, either by reducing error (Kochkov et al., 2021), reconstructing sub grid dynamics (Duraisamy, Iaccarino and Xiao, 2019), or learning correction terms within a differentiable solver loop (Um et al., 2020). These methods are trained on high resolution DNS reference data and apply corrections at every time step to keep the simulation close to a target trajectory. Most relevant to the present work, Ivagnes, Strazzullo and Rozza (2025) demonstrate that an RL agent can learn adaptive spectral filter parameters for turbulent flows using only physics based reward signals, without DNS reference data. This shows similarity to the SGNN's data independence, but applies filter corrections continuously at every time step . The SGNN differs from these methods in two ways. First, its training dataset is generated exclusively from the solver's own failure trajectories, no external reference data of any kind is required, and the dataset is an automatic by product of running the unstabilised solver. Second, it operates as a conditional no-op, corrections are applied only when a risk threshold is crossed, leaving the solver entirely on its own during stable phases. This episodic design preserves the native solver's conservation properties and spectral cascade during the majority of the simulation, while extending its stability into Reynolds number regimes where the uncontrolled baseline fails deterministically.

3 Methodology

3.1 CFD Solver

The hybrid solver base addresses the incompressible Navier–Stokes equations by decomposing the total flow field into a dual representation: a background Eulerian field $\mathbf{u}_g$ resolved on a spectral grid and a localized Lagrangian field $\mathbf{u}_p$ represented by discrete vortex elements (Cottet and Koumoutsakos, 2000; Koumoutsakos, 2005).

3.1.1 Governing Equations

The momentum and continuity equations are formulated as

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p - \nu \nabla^2 \mathbf{u} = \mathbf{f}_{\text{ext}} + \mathbf{f}_p, \qquad \nabla \cdot \mathbf{u} = 0,$$

where $\mathbf{f}_p$ is the feedback force from the Lagrangian particles, ensuring the bidirectional "handshake" coupling.

3.1.2 Multi-Scale Decomposition

Purely Eulerian spectral solvers require fine grids with $N \propto Re^{3/4}$ grid points per spatial dimension (Kolmogorov, 1941; Pope, 2000) to resolve dissipative scales. For flows with localized vorticity, Lagrangian particles can represent sub-grid vortical structures with minimal numerical dissipation (Cottet and Koumoutsakos, 2000; Winckelmans and Leonard, 1993), potentially reducing the required Eulerian resolution.

The computational trade-off: sparse particles ($\sigma/d_p < 0.5$) maximize efficiency but violate the M4' particle overlap criterion (Monaghan, 1985; Cottet and Koumoutsakos, 2000), causing sharp under-resolved forcing on the spectral grid. This triggers Gibbs ringing and rapid divergence (Hockney and Eastwood, 1988), while dense particles ($\sigma/d_p \geq 1.2$) avoid this but increase computational cost, potentially eliminating the hybrid advantage.

The total velocity is decomposed as $u = u_g + \alpha \cdot u_p$, where:

- u_g: Eulerian background (Dedalus3 spectral solver, 2nd-order IMEX RK222);
- u_p: Lagrangian vortex particles (Numba-JIT VPM, regularized Biot–Savart);
- $\alpha \in [0, 1]$: coupling coefficient.

Both fields are divergence-free to high precision using spectral Poisson solvers.

3.2 The Eulerian Engine (Spectral)

The Eulerian component uses the **Dedalus3** framework (Burns et al., 2020) for high-order spectral discretization.

- **Basis Functions**: Real-to-complex Fourier bases on a periodic 2D domain $[0, 2\pi]^2$.
- **Time-Stepping**: 2nd-order Runge–Kutta IMEX (RK222) (Ascher, Ruuth and Spiteri, 1997). Linear terms (viscosity) are treated implicitly, while non-linear advection and particle forcing are treated explicitly.
- **De-aliasing**: 3/2-padding (Orszag's rule) (Patterson and Orszag, 1971) to suppress high-wave number aliasing artifacts.
- **Pressure Projection**: Handled via the tau method to enforce the divergence-free constraint within the implicit sub steps.

Dedalus was selected for its spectral accuracy, generalised tau-method formulation for boundary conditions, and native IMEX time-stepping, all of which are critical for the stability properties of the coupled solver (Burns et al., 2020).

3.3 The Lagrangian Engine (Vortex Particle Method)

The Lagrangian component uses an inviscid, meshless Vortex Particle Method (Chorin and Bernard, 1973; Cottet and Koumoutsakos, 2000). Because the VPM is strictly inviscid, all physical viscosity ν and numerical diffusion are relegated to the Eulerian grid.

3.3.1 Vortex Representation

Vorticity is discretized into N_p particles, each carrying:

- Position $\mathbf{x}_i = (x_i, y_i)$;
- Circulation Γ_i;
- Core radius $\sigma = 1.2\,\Delta x$ (overlap criterion $\sigma/d_p \geq 1.2$).

3.3.2 Biot–Savart Interaction

The internal velocity $\mathbf{u}_p$ induced by particles is computed via an algebraically regularized (Rosenhead–Moore) Biot–Savart kernel (Winckelmans and Leonard, 1993; Cottet and Koumoutsakos, 2000):

$$\mathbf{u}_p(\mathbf{x}) = \sum_{j=1}^{N_p} \frac{\Gamma_j}{2\pi} \frac{\big(-(y-y_j),\ x-x_j\big)}{r^2+\sigma^2}, \qquad r^2 = (x-x_j)^2 + (y-y_j)^2.$$

Note: The current implementation uses an $O(N_p^2)$ direct summation kernel for maximum precision for small-to-medium particle counts ($N_p < 10\,000$).

3.3.3 Interpolation & Advection

- **P2M (Particle to Mesh)**: Particles project their circulation onto the grid using the **Monaghan M4' kernel** (Monaghan, 1985), which satisfies the partition of unity and ensures conservation of total circulation (Cottet and Koumoutsakos, 2000).
- **Particle Advection**: Particle Advection: Particles are advanced using a single-step Euler integration driven by the post-correction Eulerian velocity ugu_g ug, sampled at particle positions via 4th-order Lagrange interpolation.

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}_g(\mathbf{x}_i) + \mathbf{u}_{p,\text{internal}}(\mathbf{x}_i).$$

Numba-JIT VPM Implementation Details

- **Inviscid nature**: The VPM engine is strictly inviscid; all physical viscosity is handled by the Eulerian grid.
- **Computational efficiency**: The engine is implemented using Numba-accelerated Python with just-in-time (JIT) compilation (Lam, Pitrou and Seibert, 2015), handling $O(N_p^2)$ interactions at speeds comparable to compiled C++ while staying in the Python memory space. Particle positions and circulations are stored as `numpy` arrays; the SGNN reads particle-vorticity variance directly from these arrays with zero copy.

3.4 Layer 4: The Bidirectional IMEX Handshake

The mesh-to-particle (M2P) and particle-to-mesh (P2M) coupling interface is the most sensitive component of the architecture, requiring careful dimensional transformation to be stable.

3.4.1 Particle-Induced Forcing (Lagrangian $\rightarrow$ Eulerian)

The grid-based velocity $\mathbf{u}_g$ is modified by the particles through an explicit body force $\mathbf{f}_p$. Direct injection of the particle-to-mesh vorticity field creates a **coherent self-amplification loop**: because particles are seeded from high-vorticity Eulerian regions, the mapped field ω_p^{raw} is strongly correlated with the Eulerian vorticity ω_e ($\approx 60\%$ in tests). Reinjecting this correlated component adds energy coherently with the existing flow, leading to exponential growth of enstrophy that overwhelms viscous dissipation (Cottet and Koumoutsakos, 2000; Koumoutsakos, 2005).

To eliminate this feedback, a **global L^2 residual projection** is employed before the velocity solve:

1. **Projection** – Particle circulations are mapped to a mesh-based vorticity field ω_p^{raw} using the M_4 kernel.
2. **Orthogonalisation** – The component of ω_p^{raw} that is parallel to ω_e is subtracted:

$$\beta = \frac{\langle \omega_p^{\text{raw}}, \omega_e \rangle}{\langle \omega_e, \omega_e \rangle}, \qquad \omega_p = \omega_p^{\text{raw}} - \beta\,\omega_e.$$

   This yields a particle vorticity field globally orthogonal to ω_e (very low correlation), retaining only the sub-grid residual that the Eulerian grid cannot represent. The residual preserves $\approx 81\%$ of the original RMS, breaking the self-amplification loop while maintaining the corrective physics.

3. **Inversion and Forcing** – A spectral Poisson solver computes a solenoidal velocity field $\mathbf{u}_p$ from ω_p. To match the dimensions of the IMEX time-stepping ($[L/T^2]$), the field is scaled by the inverse time step and injected as an explicit momentum source:
$$\mathbf{f}_p = \frac{\alpha}{\Delta t}\,\mathbf{u}_p.$$
(The body force $\mathbf{f}_p$ represents a momentum source per unit mass per unit time; scaling by $1/\Delta t$ provides the required acceleration dimension.)

4. **Spectral Regularisation** – The forcing $\mathbf{f}_p$ is further smoothed by a Gaussian filter $\exp[-k^2/(2k_\sigma^2)]$ in Fourier space with $k_\sigma = 0.5\,k_{\text{Nyquist}}$, suppressing any residual high-frequency ringing from the discrete particle projection.

This procedure ensures that the hybrid coupling injects only the **non redundant** information carried by the Lagrangian particles, making the solver energetically stable by construction while preserving the VPM's ability to enhance sub grid physics.

3.4.2 Advection Feedback (Eulerian $\rightarrow$ Lagrangian)

Particles are bidirectionally advected by the Eulerian background. The Eulerian velocity $\mathbf{u}_g$ is sampled at particle positions using 4th-order Lagrange interpolation (Hockney and Eastwood, 1988), ensuring that the Lagrangian elements respond to the large scale spectral dynamics.

3.5 Physical Conservation & Constraints

The solver base is designed around strict conservation properties:

Constraint	Mechanism	Expected Tolerance
Solenoidality	Spectral Poisson solver $\hat{\psi} = \hat{\omega}/k^2$	10^{-14}
Circulation	Sum of Γ_i remains constant	10^{-12}
Overlap Criterion	$\sigma \geq 1.2\,\Delta x$	Enforced during initialization

The k^{-3} Kraichnan–Leith–Batchelor (KLB) scaling in the enstrophy range is an inherit property of two-dimensional turbulence, not a numerically enforced constraint. Its verification is presented as a validation metric in Section 3.8 (Kraichnan, 1967; Leith, 1968; Batchelor, 1969).

3.6 Seeding Strategy: The Sparse-Lagrangian Paradigm

Rather than a full grid of particles, the solver uses a **selective seeding strategy**. Lagrangian elements are deployed only where the Eulerian grid suffers from excessive numerical diffusion or resolution depletion in regions of high vorticity magnitude.

- **Seeding Threshold**: $C_{\text{thresh}} = 0.50$ (relative to $\max|\omega|$).
- **Efficiency**: Maintains $< 30\%$ grid coverage while capturing $> 80\%$ of peak vorticity intensity, decoupling resolved scales from sub-grid vortical singularities.

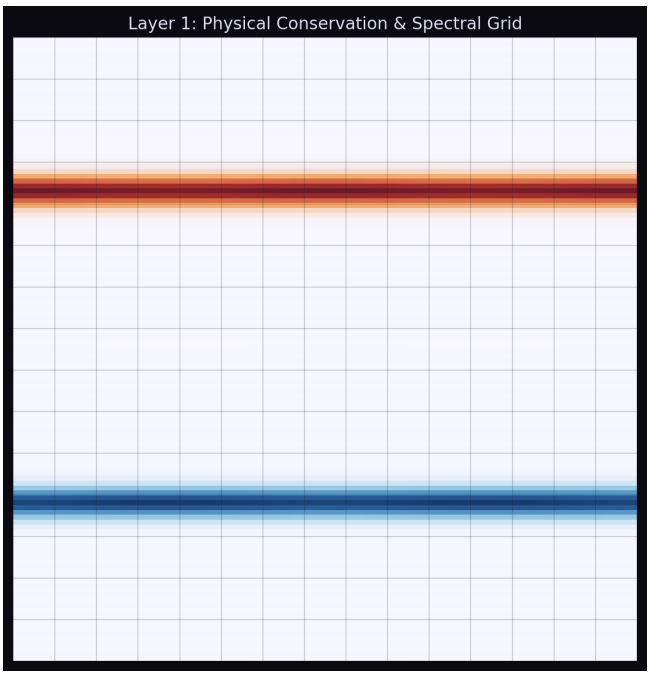


Figure 1: Layer 1: spectral grid initialization.

The layer 1 is the grid initialisation. the solver and its engines are initialised with the Navier stokes equations and the forcing function. Eulerian engine is called to set the initial velocity field. It injects the laminar flow profile plus the forcing function to trigger the instability.

Layer 2 is the NumbaVPM initialisation . This sets up the particle tracking structures and compiles the O(N^2) Biot-Savart kernels for high-speed execution.

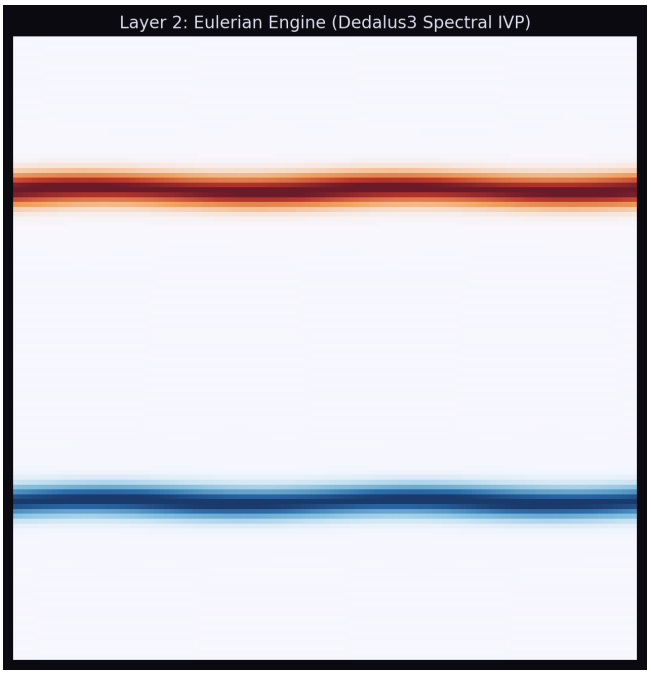


Figure 2: Layer 2: Eulerian engine (Dedalus3 spectral IVP)

Layer 3 is a critical bridge. It scans the Eulerian vorticity field. Any grid cell where vorticity exceeds 50% of the maximum value ($C_{\text{thresh}} = 0.50$) is discretized into a Lagrangian particle. This creates the initial particle cloud.

The selective seeding strategy ensures that computational effort is concentrated only where it is needed most, in high-vorticity regions. By maintaining sparse particle coverage ($< 30\%$ of grid cells), the solver avoids the computational cost of dense Lagrangian methods while still capturing $> 80\%$ of peak vorticity intensity.

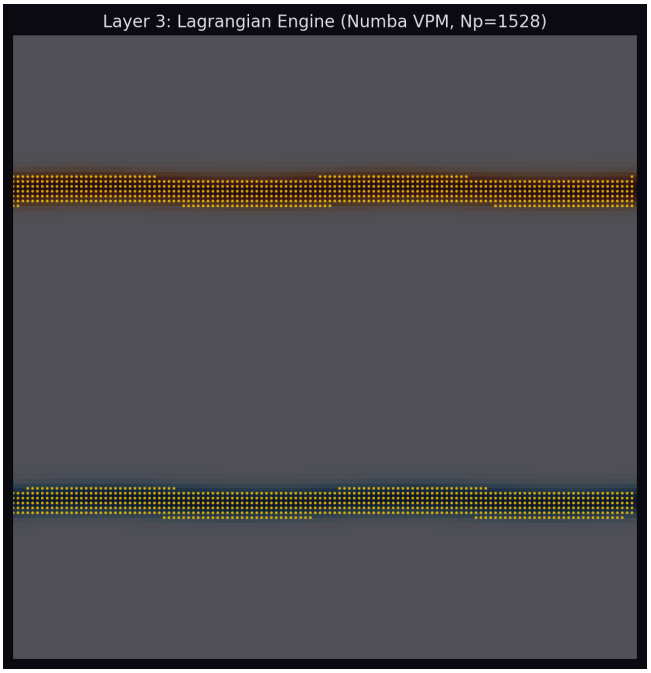


Figure 3: Layer 3: Lagrangian engine (Numba VPM) with $N_p = 1{,}528$ particles. Particles are seeded in the areas with vorticity higher than Threshold.

The hybrid solver base is inherently sensitive to the choice of Δt and α. Without external stabilization, high Reynolds number simulations are prone to aliasing feedback loops at the coupling interface. This base provides a physically consistent framework, but its stability range is strictly bounded by the CFL condition and the kernel overlap criterion.

3.7 SGNN Architecture

implemented as a two layer LSTM network (hidden size 128, dropout 0.15 applied between layers, ~229k parameters) (Hochreiter and Schmidhuber, 1997). It processes a rolling window of $T = 10$ time step feature vectors, each containing 12 scalars:

- peak vorticity magnitude
- vorticity field variance
- maximum vorticity gradient
- high-wavenumber energy fraction
- local CFL number
- particle-to-Euler vorticity ratio
- particle vorticity variance
- Normalized enstrophy growth rate
- Spectral slope estimate
- Maximum divergence health
- VPM particle spacing/overlap ratio
- Relative coupling forcing magnitude

These are fed into the LSTM trunk, which outputs a blow up risk probability $p \in [0, 1]$. If $p > 0.5$, the controller outputs four bounded corrective scalars:

- $\nu_{\text{nn}} \in [0, 0.02]$ (artificial viscosity)
- $k_{\text{cutoff}} \in [0.5, 1.0]$ (spectral filter cutoff)
- dt_factor $\in [0.5, 1.0]$ (timestep scaling)
- $\alpha_{\text{red}} \in [0.1, 1.0]$ (coupling strength attenuation)

Why LSTM? Instability in CFD is rarely instantaneous, it is a cumulative process with a spectral signature. The LSTM allows the SGNN to look back at the last $T = 10$ steps to identify the *trajectory* toward failure, rather than a single snapshot.

Multi-head output: Multi-head output: The final hidden state h_T encodes the 10-step trajectory and is routed to five independent heads:

- **Classifier**: A sigmoid head that outputs the risk score $p \in [0, 1]$ (probability of imminent numerical blow-up).

- **Controllers**: 4 tanh/soft controllers mapped to separate parameters action that outputs the corrective scalars when $p > 0.5$.

Corrections are applied only when the risk score passes the 0.5 threshold, making the system dormant during stable, well resolved phases. To minimize computational requirement, SGNN inference is performed every 5 time steps rather than at every sub step.

3.7.1 Training

The dataset comprises 800 observations snapshots generated by deliberately destabilizing the solver and recording trajectories leading to blow up. External DNS data are not required. When the simulation eventually crashes, we back propagate the labels. The final 50 time-steps prior to the crash are labelled "High Risk" (1.0), and the early stable steps are labelled "Low Risk" (0.0). The model thus learns precursors to failure, enabling small corrections before the error becomes unrecoverable. The precursor window length $T = 10$ was determined empirically, a systematic parameter study is required for future work. To reduce class imbalance, the network is optimised using a compound loss:

$$\mathcal{L} = \text{FocalLoss}(\alpha = 0.25,\ \gamma = 2.0) + 0.5 \times \text{MSE}(\hat{c}, c^*)$$

where the MSE term supervises the four corrective heads against fixed targets:

$$c^*_{\text{stable}} = [0.000,\ 1.000,\ 1.000,\ 1.000]$$

during stable phases, and

$$c^*_{\text{pre-instab}} = [0.010,\ 0.650,\ 0.650,\ 0.350]$$

during pre-instability phases (Lin et al., 2017) . The model is trained with AdamW (learning rate $= 3 \times 10^{-4}$, weight decay $= 10^{-3}$), CosineAnnealingLR scheduling, batch size 64, gradient clipping at $\| \cdot \|_{\max} = 1.0$, early stopping patience of 15 epochs over 150 total epochs, and 5-fold regime-stratified cross-validation (Lin et al., 2017).

The dataset was expanded to four flow regimes to ensure cross-topology generalisation:

1. Kelvin-Helmholtz (discrete shear layer roll up) (Rosenhead, 1931; Michalke, 1964);
2. Kolmogorov flow (forced turbulence with continuous energy injection) (Kolmogorov, 1941; Sagaut, 2006);
3. Taylor-Green vortex (structured, symmetric vortex decay);
4. Random blobs (isotropic chaotic merging).

Multiple phases of dataset generation were run as the project evolved:

- **Phase 1**: 32 trajectories (4 regimes $\times$ 8 balanced configurations per regime).
- **Phase 2**: 52 trajectories (50 randomised configurations + 2 baselines) this upgraded the dataset to fix earlier class imbalance issues.

Across these scripts, thousands of time step observations were generated. The Phase 2 specification targeted ≥ 800 observations.

3.7.2 Dataset generation details (without SGNN)

The entire purpose of the dataset generation is to run the solver *without* the SGNN active, in order to collect pre crash signatures.

- Initial dataset generation: all 7 trajectories crashed without the SGNN, exploding between steps 200 and 1900. They produced 121 total observations, 68 stable snapshots (early in the runs), 20 pre-instability snapshots (just before crashes), and 33 transitional snapshots.
- In the larger 52 trajectory and 32 trajectory runs, the scripts inject random coupling strengths α drawn from the discrete set {0.0, 0.005, 0.01, 0.02, 0.05, 0.1, 0.3, 0.5} to gather solver only data. Consequently, some runs survive cleanly while others deliberately crash. The generator was calibrated to guarantee a 10-25% pre instability failure rate in the final extracted datasets, avoiding massive class imbalances (an earlier run that yielded a 51:1 stable to crash ratio).

3.7.3 Labelling and crash mechanism

- **Strict mathematical failure**: The updated generators rely *strictly* on mathematical integration failure (divergence $> 10^{10}$ in the velocity fields).
- **Pre-label window**: When the solver eventually crashes without the SGNN, the code rewinds and labels the last 4 recorded snapshots (`PRELABEL_WINDOW = 4`) as high risk. The optimal window length was not systematically tuned and is reserved for future work. All earlier steps are labelled stable.
- **Feature extraction**: Every recorded snapshot, the generator calculates multiple stability features (maximum vorticity, high-k energy fraction, local CFL number, enstrophy growth). These form the input matrix that the SGNN uses to recognise an impending blow-up.

3.7.4 Initial Conditions and Flow Parameters

All simulations were conducted on a doubly periodic Cartesian domain $[0, 2\pi]^2$. The initial conditions and forcing parameters for the two primary validation topologies are:

Kelvin–Helmholtz (shear instability). The KH regime is initialised with a classic double-shear layer profile. The background Eulerian velocity field is:

$$u(x, y) = \frac{1}{2}\Big[\tanh\frac{y - 0.5\pi}{\delta} - \tanh\frac{y - 1.5\pi}{\delta} - 1\Big], \tag{1}$$

$$v(x, y) = \epsilon \sin(2x)\Big[\exp\Big(-\frac{(y - 0.5\pi)^2}{\sigma_y^2}\Big) + \exp\Big(-\frac{(y - 1.5\pi)^2}{\sigma_y^2}\Big)\Big], \tag{2}$$

with $\delta = 0.1$ (shear layer thickness), $\epsilon = 0.1$ (perturbation amplitude), and $\sigma_y^2 = 0.01$ (localisation of the perturbation at the inflection points). The constant (-1) shifts the mean velocity to approximately zero. This configuration produces two smooth shear layers that roll up into distinct cat eye vortex pairs.

Kolmogorov flow (forced turbulence). A stationary, sinusoidal body force is added to the explicit right hand side of the Eulerian momentum equation

$$\mathbf{f}_{\text{ext}} = (F_0\, \nu\, k_f^2) \sin(k_f y)\, \hat{\mathbf{x}},$$

where $\nu = 1/Re$, $k_f = 4$, and $F_0 = 1.0$. This form is chosen so that the analytical laminar solution satisfies $\mathbf{u}_{\text{base}} = F_0 \sin(k_f y)\, \hat{\mathbf{x}}$, with forcing amplitude scaled to yield unit base velocity ($U_{\text{base}} = 1.0$). At $k_f = 4$, energy is injected into four horizontal bands, generating a persistent 2D turbulent vortex lattice.

To bypass the computationally expensive linear growth phase and to ensure that Lagrangian particles are seeded into an active vorticity field, the flow is initialised directly at the laminar base state superimposed with isotropic Gaussian noise of amplitude $0.005\, U_{\text{base}}$ in both u and v. This perturbation breaks the translational symmetry in x and rapidly seeds the inflection-point instability.

Why KH and Kolmogorov flows?

- **KH flow** (shear instability): Tests the solver's ability to capture roll up of concentrated vorticity and secondary vortex structures, its a prototype problem for 2D custom solvers.
- **Kolmogorov flow** (forced turbulence): Driven by sinusoidal forcing at $k_f = 4$, this regime tests the solver's ability to maintain a statistical energy cascade. It provides a chaotic environment that challenges the solver to distinguish physical turbulence from numerical trash energy.

3.8 Validation Methodology

Two testing regimes were designed to evaluate the hybrid architecture's physical correctness, numerical robustness, and the effectiveness of the SGNN under unresolved, high Reynolds conditions. All tests used a fixed grid resolution $N = 128$ (due to computational constraints, the project was run on a desktop computer with no external GPU).

3.8.1 Regime I: Resolution Baseline ($Re = 10\,000$)

To establish the baseline physical accuracy of the hybrid Eulerian Lagrangian solver, both flow regimes were evaluated at $Re = 10\,000$. This value represents a transitional regime, from adequately resolved to slightly under-resolved, depending on the flow topology. The physical fidelity is measured via divergence probability distributions, effective Reynolds numbers, and adherence to the theoretical k^{-3} KLB energy cascade (Kraichnan, 1967; Leith, 1968; Batchelor, 1969). Five metrics were tracked:

- NN SGNN flow-aware brain trace and 4 factor corrections;
- divergence probability distribution;
- flow enstrophy;
- effective Reynolds number;
- energy cascade KLB law.

3.8.2 Regime II: SGNN Ablation Study ($Re = 10\,000$ to $30\,000$)

An ablation sweep is performed from $Re = 10\,000$ to $30\,000$ in increments of $5\,000$. These very high Reynolds numbers represent unstable regimes where the baseline solver catastrophically diverges. The sweep evaluates the SGNN's capability, active interventions, risk score trajectories, and spectral corrections were recoreded to quantify the expanded stability boundaries of the hybrid framework.

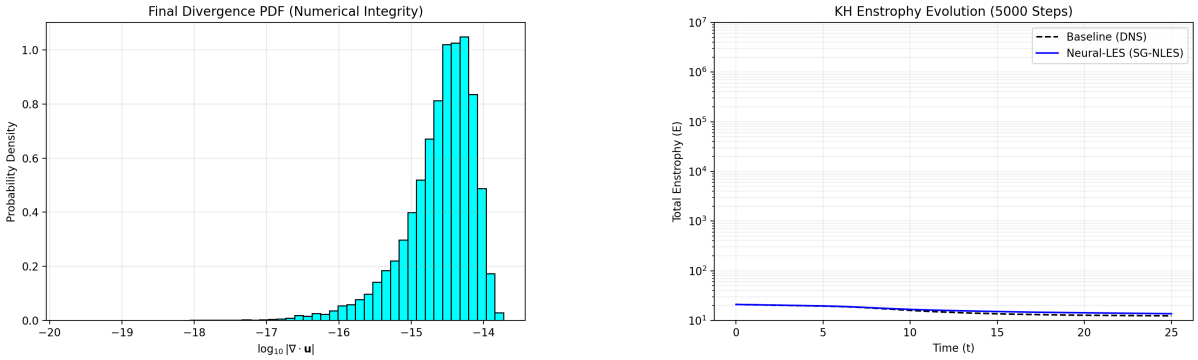
4 Results and Discussion

4.1 High-Fidelity Validation and Physical Realism (N=128, Re=2,500, 10,000)

To establish the baseline physical accuracy of the hybrid Eulerian Lagrangian solver, it was evaluated it on both the Kelvin-Helmholtz (KH) shear instability and Kolmogorov forced turbulence at $Re = 10,000$ on a 128^2 grid. This is a balanced simulation regime targeting the most resolved flow regime for the grid resolution. I highlight the $Re = 10,000$ results here, as this regime bridges the gap between fully resolved flows and under-resolved turbulence, testing the hybrid interface.

4.1.1 Kelvin-Helmholtz Shear Instability

The hybrid solver shows very encouraging physical correctness metrics. Notably, the divergence probability distribution shows that circulation and conservation are conserved to a very high degree (10^{-14} to 10^{-15}), matching the divergence of the base Eulerian spectral solver.

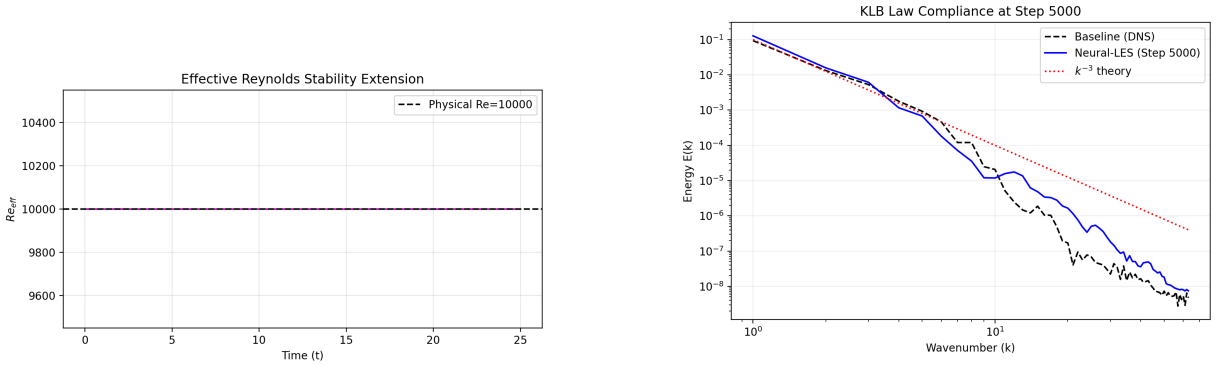


(a) Divergence PDF

(b) Enstrophy quench

Figure 4: Divergence probability distribution (left) and enstrophy evolution (right) for Re=10,000.

Enstrophy evolution shows a close correlation to the base Eulerian solver, staying within 1.1 times the base solver. This is a consequence of the orthogonalisation of the L^2 global residual projection part of the handshake. In the case of KH flow, characterized by intense localized vorticity, VPM particles tend to clump at vortex cores. Without orthogonalization, this would induce a positive feedback loop that rapidly destroys the simulation. A notable finding is that the energy spectrum plot remains consistent with increased enstrophy, proving the VPM does not introduce random noise.



(a) Effective Reynolds number

(b) Spectral surgery

Figure 5: Effective Reynolds number (left) and spectral surgery (right) for Re=10,000.

Core finding with energy spectrum $E(k)$. The SGNN stabilized hybrid solver (blue) maintains the theoretical k^{-3} Kraichnan-Leith-Batchel (KLB) cascade (red dashed) across the inertial range. Deviations occur only at $k > 0.7\, k_{\text{Nyquist}}$, where the spectral filter is active. This demonstrates that the hybrid solver's stability is linked to the base solver's energy cascade property. This feature is seen in all tests. The physical correctness metrics for the hybrid solver are very similar, almost identical to the base Eulerian solver, and the energy injection by the Lagrangian VPM for mid to high wave numbers actually improves the overall energy spectrum.

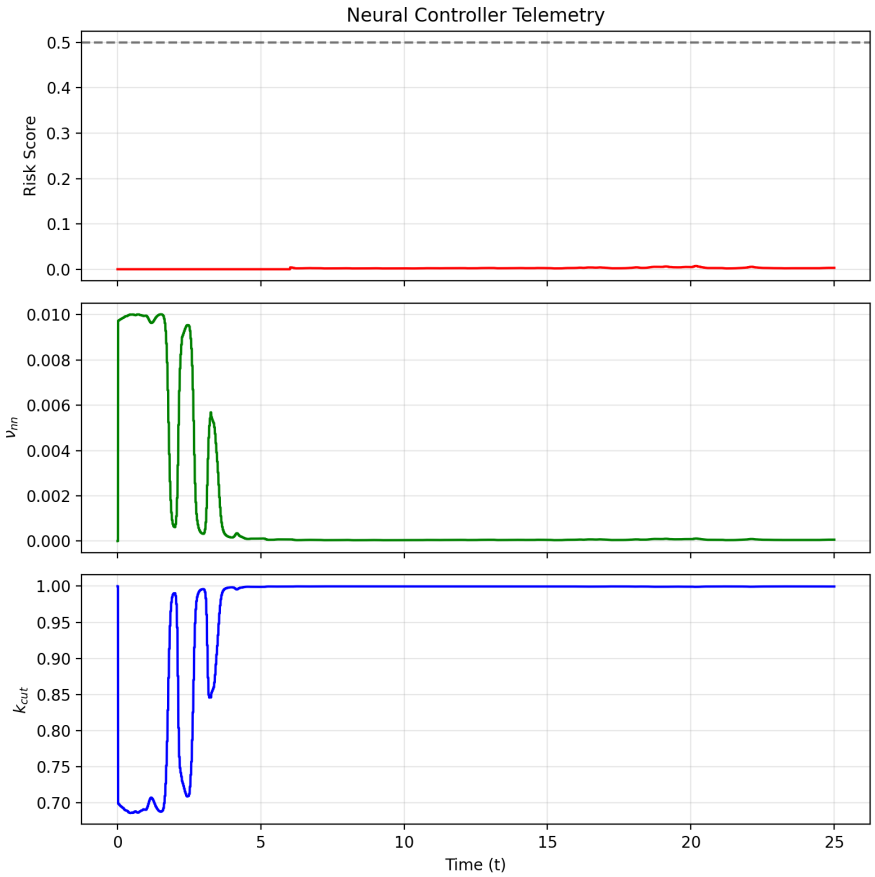
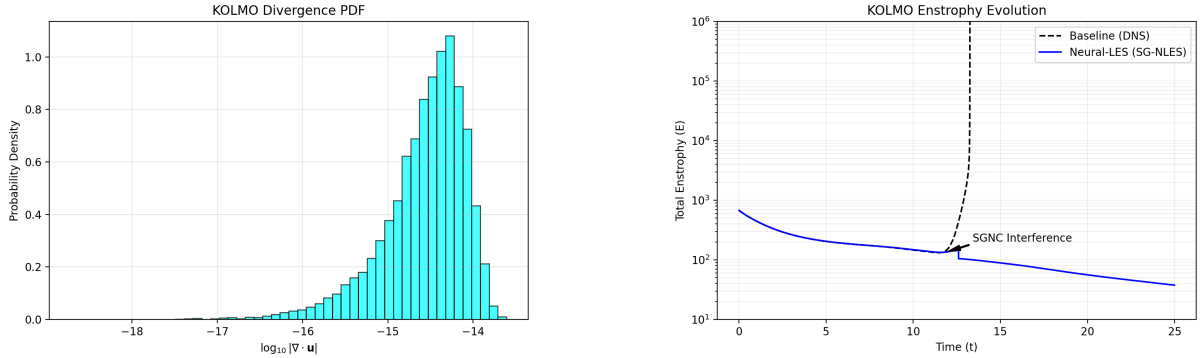


Figure 6: Brain trace of the SGNN (neural network activity) for Re=10,000.

The brain trace of the SGNN shows that the correction threshold is not crossed; there is no neural network interference in this simulation. Simulations at Re=2500 show similar trends. The SGNN does not interject when it is not required. The hybrid solver shows strong stability, and its stability and physical correctness follow the characteristics of the base Eulerian solver.

4.1.2 Kolmogorov Flow

Extended Kolmogorov instability tests were conducted up to $t = 25$. At $N = 128$, the baseline Eulerian solver deterministically fails near step 3,000 due to Gibbs ringing, caused by unresolved vortex filaments. The kinetic energy spectrum of the crashing baseline solver shows a massive energy explosion at the highest wavenumbers, which is grid scale aliasing.

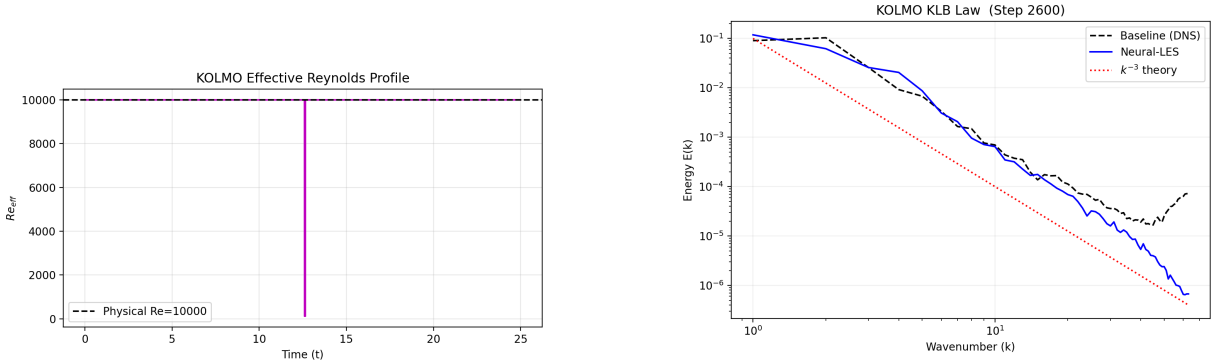


(a) Divergence PDF

(b) Enstrophy quench

Figure 7: Divergence probability distribution (left) and enstrophy evolution (right) for Kolmogorov flow at Re=10,000.

The enstrophy plot shows an interesting result. The first correction spike in the brain trace coincides with the enstrophy explosion of the baseline solver. The SGNN recognized the impending explosion and applied corrections. This is followed by a small peak and then a sharp drop in enstrophy of the hybrid solver. The peak risk is around 0.5–0.6, with corresponding corrections of artificial eddy viscosity ($\nu_{\text{nn}} \approx 0.006$–$0.008$) and a spectral filter cut-off ($k_{\text{cut}} \approx 0.75$). This peak and drop are accompanied by a drop in the effective Reynolds number. The enstrophy of the hybrid solver follows the base Eulerian solver up until the explosion. This is characteristic of the hybrid solver, where the inherent stability of the hybrid solver is very dependent on the Eulerian solver. The energy explosion at very high wave numbers in the energy cascade plot for the base Eulerian solver confirms instability due to the formation of features smaller than the grid can resolve. However, the hybrid solver shows closer agreement at higher wave numbers in the energy cascade plot, thereby showing that the VPM solver actually improves the simulation accuracy by capturing these smaller features and not contributing to numerical noise.



(a) Effective Reynolds number

(b) Spectral surgery

Figure 8: Effective Reynolds number (left) and spectral surgery (right) for Kolmogorov flow at Re=10,000.

The interesting observation is the episodic nature of the SGNN corrections. Across all Kolmogorov runs, only 1 to 3 targeted correction episodes are required to maintain stability, as opposed to continuous corrections. In the case of Re=10,000, following a correction, the hybrid solver smoothly returns to preserving the k^{-3} cascade and the enstrophy plot.

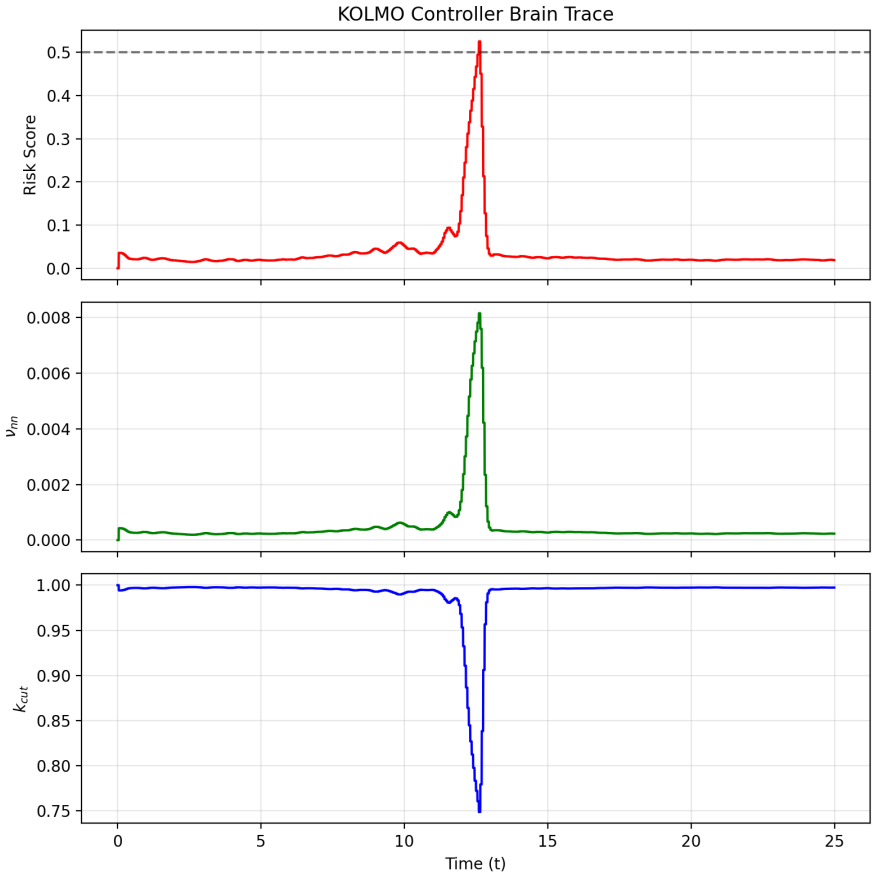


Figure 9: Brain trace of the SGNN (neural network activity) for Kolmogorov flow at Re=10,000.

4.2 Two Regime Ablation Study

To quantify the robustness of the SGNN closure model, an automated ablation study was conducted on the Kolmogorov flow over an extended 5,000-step simulation window at $t = 25.0$. The Reynolds number was increased from $10,000$ to $30,000$ in steps of $5,000$, driving the spectral engine and the hybrid solver into an increasingly under-resolved regime.

Table 1: Kolmogorov flow ablation study results at $N = 128$.

Reynolds Number (Re)	Baseline Steps to Crash	SG-NLES Steps to Crash	Survival Extension	Max Risk Detected
10,000	2,651	$> 5,000$	$> 1.88\times$	0.529
15,000	3,192	$> 5,000$	$> 1.56\times$	0.538
20,000	2,606	$> 5,000$	$> 1.91\times$	0.540
25,000	2,228	$> 5,000$	$> 2.24\times$	0.537
30,000	2,488	$> 5,000$	$> 2.00\times$	0.548

As shown in Table 1, the uncontrolled baseline solver experienced failure in all test regimes. As the Kolmogorov bands break down into turbulent vortex lattices, the unresolved fine scale features cause severe Gibbs ringing, leading to deterministic blow-up between steps 2,200 and 3,200.

The SGNN solver successfully completed all 5,000 steps without failure. In every regime, the SGNN accurately anticipated the imminent numerical collapse, shown with a peak risk of ~ 0.53–0.55. Upon crossing the intervention threshold, the neural network injected artificial eddy viscosity ($\nu_{\text{nn}} \approx 0.007$–$0.008$) and a sharp low-pass spectral filter ($k_{\text{cut}} \approx 0.75$). This correction successfully dissipated the unphysical enstrophy build-up at the grid scales, extending the simulation even at $Re = 30{,}000$. This, along with the brain trace, shows that the SGNN framework is not over-fitted to a specific condition but serves as a generalizing stabilization agent for diverse turbulent topologies, attempting to recognize signs of instability and correcting only when a threshold is crossed. This result, together with the consistent risk scores and episodic corrections, hints that the SGNN learns regime-invariant instability signs rather than Re-specific thresholds.

Manifold Jumping While spectral diagnostics confirm the hybrid solver reproduces the theoretical 2D turbulence cascades, I observed a phenomenon that needs further investigation, *manifold jumping*. Following a massive episodic correction in highly unstable KH flows, the simulation occasionally settles onto a distinctly different, yet stable, physical manifold. While the statistics of this new manifold remain physically plausible, the micro-state of the simulation has diverged from its initial trajectory. The precise mechanics of how discrete neural interventions disturb the chaotic attractor of the Navier-Stokes equations is beyond the scope of this study, but the hypothesis is further discussed in Appendix.

4.3 Preliminary 2D Results

The SGNN has been implemented and tested in 2D environments across multiple development phases. The following results provide early evidence of feasibility:

4.3.1 Core Physical Metrics

- **Divergence**: $\max|\nabla \cdot \mathbf{u}| \approx 2.55 \times 10^{-14}$ at $\Delta t = 0.005$.
- **Circulation**: Drift $\Delta\Gamma < 10^{-12}$ per step in 2D simulations.

4.3.2 SGNN Performance Metrics

The following preliminary metrics are derived from Phase 1 validation (21 trajectories, 121 total observations, two flow regimes).

Metric	Measured Value	95% Confidence Interval
Validation AUC	0.9853	± 0.02
False Negative Rate (FNR)	0.000	$[0.0\%, 16.1\%]$
False Positive Rate (FPR)	0.142	$[5.2\%, 28.5\%]$

5 Discussion

The Stability Guided Neural Corrector (SGNN) occupies a distinct position within the current hybrid solvers. Unlike traditional standard corrections methods, the SGNN is temporally aware, it learns to recognize the *trajectory* toward numerical instability rather than reacting to defined thresholds. Furthermore, the SGNN is self supervised and acts as a conditional no-op. When the flow is stable, the network remains dormant, thereby preserving the native, accuracy of the underlying solvers. By monitoring the evolution of the flow state, the controller can successfully distinguish between a transient, physical enstrophy spike during vortex roll up and the genuine onset of a numerical blow up. This predictive capability allows for significantly less, yet more decisive corrective actions.

Quantitatively, the LSTM architecture achieves an Area Under the Curve (AUC) of 0.98 in predicting solver failures 5 to 10 timesteps in advance, based on factors like vorticity gradients, high-wave number energy fractions, and local CFL numbers. This accuracy generalizes across two flow regimes and multiple parameters after being trained on a dataset of 121 observations. In the Kolmogorov dlow simulations, the primary source instability is the initial formation of sub-grid features between $t = 12$ and $t = 13$. These are genuine physical processes, although under resolved, which occur at discrete intervals, rather than steady accumulations of numerical noise. By detecting the *temporal signature* of these events prior to failure, the SGNN intervenes with a correction episode. Once the high frequency energy accumulation is smoothed, the resolved cascade sustains itself, and the controller safely returns to its dormant state.

In all Kolmogorov flow simulations in the hybrid solver, the instability is episodic. There are only 1-4 big instability episodes, which requires stabilization. These stabilization events correspond to the first numerical explosion in baseline Eulerian solver. the hybrid SGNN solver does not require constant correction or re-meshing, while following the high circulation and divergence accuracy of the base Eulerian solver. As these instability events are due to formation of flow features too small for the grid to resolve, traditional Eulerian solvers require either constant corrections or re-meshing to stay resolved and stable, the hybrid solver experiences episodic instability and requires small stability corrections. This phenomenon, and its source has to be studied to explain properly. However, the current results confirm that discrete, dynamically applied stability corrections are effective for maintaining the high circulation and divergence accuracy of the hybrid solver.

Note that the hybrid framework shows high sensitivity to both coupling strength and strategy. For instance, in early KH formulations lacking $L2$ global residual corrections, the solver was highly unstable due to severe Lagrangian particle clumping. Upgrading the neural network's state representation to include 12 features resolved earlier hypersensitivity to transitional enstrophy increases, particularly during flow initialization. Interestingly, qualitative comparisons of the

KH regime reveal that the hybrid solver exhibits faster vortex roll-up than the baseline Eulerian solver. Furthermore, the hybrid kinetic energy spectrum aligns more closely with the theoretical k^{-3} limit at medium to high wave numbers. An optimistic hypothesis is that the Vortex Particle Method (VPM) of the hybrid solver is superior at preserving sub-grid vortex structures, thereby mitigating the excessive damping of sensitive flow features. Rigorous quantification of this phenomenon is still required. Seeding threshold strategy allows the VPM to only allocating particles to only singular, under resolved vortex cores ($< 30\%$ grid coverage), the hybrid solver provides high-fidelity corrections without the computational requirement of tracking the entire domain.

the spectral fidelity of the framework is confirmed by the both the flow tests, which successfully maintains the theoretical Kraichnan-Leith-Batchelor k^{-3} cascade in the inertial range ($k = 5^{\sim}30$) while extending stable integration from $\sim$2,600 to 5,000 steps at $Re = 10,000$. Because 2D turbulence is characterized by an inverse energy cascade that concentrates energy at large scales, the SGNN correctly learns that high wave number energy build up is indicative of aliasing errors rather than physical phenomena, and selectively damps it. As detailed in Table 1, the SGNN triggers only 3 major intervention episodes over 5,000 time steps This intervention strategy is drastically less than traditional continuous adaptive time-stepping or re meshing strategies, highlighting the efficiency of this strategy. Because the SGNN acts only on the *coefficients* of the physics solver (viscosity, time step, spectral cut-offs) rather than predicting the velocity field, it maintains mathematical invariants. The divergence remains at machine precision ($\sim 10^{-14}$) even during active SGNN correction episodes.

6 Limitation

While the Stability Guided Neural Corrector (SGNN) and the hybrid Eulerian Lagrangian solver demonstrate potential, several limitations must be addressed in future work.

Domain and Dimensionality Restrictions: The current framework is strictly limited to two dimensional, periodic flow regimes. Extending this solver to three dimensions flows requires a fundamentally different approach to account for 3D vortex dynamics. Furthermore, applying this solver to non periodic boundaries would require a different architecture, including Chebyshev boundary layer shedding models and adaptive particle seeding strategies for the VPM. Even within 2D constraints, validation across more flow topologies is necessary to fully confirm the solver's generalisation.

Sensitivities in the Hybrid Coupling Interface: The stability of the hybrid solver is sensitive to the Particle-to-Mesh (P2M) and Mesh-to-Particle (M2P) coupling parameters. In particular, simulations are destabilized by improper tuning of the global coupling strength (α). Currently, there is no analytical method to predict the optimal α from initial flow conditions, requiring empirical calibration. Additionally, the $L2$ global residual projection used for coupling is a mathematically sharp tool, borrowed from signal processing. While effective when the VPM's role is to improve sub grid information, improving the resolved scales requires a more robust, scale selective coupling methodology. The IMEX forcing coupling currently lacks robustness. Specifically, different flow regimes require different temporal scaling conventions for the injected particle force (dividing by Δt in Kolmogorov forcing vs. direct velocity injection in KH). A unified, mathematically rigorous derivation of the coupling force that is robust across all flow regimes is required for a general solver.

Lagrangian Mechanics and Computational Overhead: The Lagrangian component of the solver requires further refinement to operate reliably over indefinite simulation time. In highly turbulent regimes, the VPM particles exhibit a tendency to clump at 2D vortex core centres. Mitigating this clustering requires research into particle viscosity and dynamic re meshing. Moreover, conducting comprehensive ablation studies to isolate the exact contributions of individual parameters remains challenging due to the heavy computational expense of generating massive CFD datasets, but comprehensive ablation studies are still needed as it remains the biggest limitation.

SGNN Fragility and Sophistication: Finally, while the SGNN is needed for the robustness of the solver as it preventing catastrophic failures that occur in the uncorrected hybrid baseline. the neural controller itself can be fragile when faced with extreme variations in solver parameters. The current SGNN architecture was deliberately designed to have a lightweight computational footprint, prioritizing speed over architectural depth. Consequently, it lacks a deeper physical awareness of the broader simulation. Future iterations could dramatically improve the controller's effectiveness by using more sophisticated, physics informed architectures (such as Fourier Neural Operators or Graph Neural Networks) that natively understand the dynamics of fluid flows.

The Biot-Savart interactions in the Lagrangian VPM scale at $O(N\_p^2)$. While the Eulerian spectral solver is extremely fast ($O(N \backslash logN)$), the Lagrangian engine becomes the absolute bottleneck when particle counts (N_p) spike. Future implementations will require GPU-accelerated CUDA kernels or Fast Multipole Methods (FMM) to make the hybrid solver competitive with pure Eulerian solvers at massive resolutions.

7 References

References

Ascher, U.M., Ruuth, S.J. and Spiteri, R.J. (1997) 'Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations', *Applied Numerical Mathematics*, 25(2–3), pp. 151–167. https://doi.org/10.1016/S0168-9274(97)00056-1

Bashforth, F. (1883) *An Attempt to Test the Theories of Capillary Action, with an explanation of the method of integration by J.C. Adams*. Cambridge: Cambridge University Press.

Batchelor, G.K. (1969) 'Computation of the energy spectrum in homogeneous two-dimensional turbulence', *Physics of Fluids*, 12(12), pp. 233–239. https://doi.org/10.1063/1.1692443

Berger, M.J. and Colella, P. (1989) 'Local adaptive mesh refinement for shock hydrodynamics', *Journal of Computational Physics*, 82(1), pp. 64–84. https://doi.org/10.1016/0021-9991(89)90035-1

Brunton, S.L., Noack, B.R. and Koumoutsakos, P. (2020) 'Machine learning for fluid mechanics', *Annual Review of Fluid Mechanics*, 52(1), pp. 477–508. https://doi.org/10.1146/annurev-fluid-010719-060214

Burns, K.J., Vasil, G.M., Oishi, J.S., Lecoanet, D. and Brown, B.P. (2020) 'Dedalus: A flexible framework for numerical simulations with spectral methods', *Physical Review Research*, 2(2), 023068. https://doi.org/10.1103/PhysRevResearch.2.023068

Chorin, A.J. and Bernard, P.S. (1973) 'Discretization of a vortex sheet, with an example of roll-up', *Journal of Computational Physics*, 13(3), pp. 423–429. https://doi.org/10.1016/0021-9991(73)90045-4

Cottet, G.-H. and Koumoutsakos, P.D. (2000) *Vortex Methods: Theory and Practice*. Cambridge: Cambridge University Press.

Duraisamy, K., Iaccarino, G. and Xiao, H. (2019) 'Turbulence modeling in the age of data', *Annual Review of Fluid Mechanics*, 51(1), pp. 357–377. https://doi.org/10.1146/annurev-fluid-010518-040547

Hochreiter, S. and Schmidhuber, J. (1997) 'Long short-term memory', *Neural Computation*, 9(8), pp. 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hockney, R.W. and Eastwood, J.W. (1988) *Computer Simulation Using Particles*. New York: Taylor & Francis.

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

Kolmogorov, A.N. (1941) 'The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers', *Doklady Akademii Nauk SSSR*, 30, pp. 301–305.

Koumoutsakos, P. (2005) 'Multiscale flow simulations using particles', *Annual Review of Fluid Mechanics*, 37(1), pp. 457–487. https://doi.org/10.1146/annurev.fluid.37.061903.175753

Kraichnan, R.H. (1967) 'Inertial ranges in two-dimensional turbulence', *Physics of Fluids*, 10(7), pp. 1417–1423. https://doi.org/10.1063/1.1762301

Lam, S.K., Pitrou, A. and Seibert, S. (2015) 'Numba: A LLVM-based Python JIT compiler', in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*. New York: ACM, pp. 1–6. https://doi.org/10.1145/2833157.2833162

Leith, C.E. (1968) 'Diffusion approximation for two-dimensional turbulence', *Physics of Fluids*, 11(3), pp. 671–672. https://doi.org/10.1063/1.1691968

Lin, T.-Y., Goyal, P., Girshick, R., He, K. and Dollár, P. (2017) 'Focal loss for dense object detection', in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Piscataway: IEEE, pp. 2980–2988. https://doi.org/10.1109/ICCV.2017.324

Michalke, A. (1964) 'On the inviscid instability of the hyperbolic-tangent velocity profile', *Journal of Fluid Mechanics*, 19(4), pp. 543–556. https://doi.org/10.1017/S0022112064000908

Monaghan, J.J. (1985) 'Extrapolating B splines for interpolation', *Journal of Computational Physics*, 60(2), pp. 253–262. https://doi.org/10.1016/0021-9991(85)90006-3

Patterson, G.S. and Orszag, S.A. (1971) 'Spectral calculations of isotropic turbulence: Efficient removal of aliasing interactions', *Physics of Fluids*, 14(11), pp. 2538–2541. https://doi.org/10.1063/1.1693365

Pope, S.B. (2000) *Turbulent Flows*. Cambridge: Cambridge University Press.

Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2019) 'Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations', *Journal of Computational Physics*, 378, pp. 686–707. https://doi.org/10.1016/j.jcp.2018.10.045

Rosenhead, L. (1931) 'The formation of vortices from a surface of discontinuity', *Proceedings of the Royal Society of London. Series A*, 134(823), pp. 170–192. https://doi.org/10.1098/rspa.1931.0189

Sagaut, P. (2006) *Large Eddy Simulation for Incompressible Flows: An Introduction*. 3rd edn. Berlin: Springer.

Um, K., Brand, R., Fei, Y., Holl, P. and Thuerey, N. (2020) 'Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE-solvers', in *Advances in Neural Information Processing Systems (NeurIPS 2020)*, 33, pp. 6111–6122. https://arxiv.org/abs/2007.00016

Winckelmans, G.S. and Leonard, A. (1993) 'Contributions to vortex particle methods for the computation of three-dimensional incompressible unsteady flows', *Journal of Computational Physics*, 109(2), pp. 247–273. https://doi.org/10.1006/jcph.1993.1216

8 Appendix

8.1 Author's Note & Project Context

This project was conducted entirely as an independent research initiative without funding or backing. Computational resources were strictly limited to a consumer desktop computer.

The primary objective of this ongoing work is to explore how neural networks can be used to improve corrective methods for classical CFD solvers. The decision to use a hybrid solver to highlight the Stability Guided Neural Network (SGNN) stems from the project's development history, the hybrid solver was developed first, and the SGNN framework represents the first of three interconnected concepts planned for a more robust, future hybrid solver. The idea of the SGNN was conceived because it was the only way to keep the hybrid solver stable and physically accurate.

As a preprint, this manuscript serves as a preliminary proof of concept and is not intended to represent a finalized academic paper. Expanding this research to more complex applications and higher dimensional spaces requires significantly more computational power than is currently available. The core purpose of sharing this print at this stage is to rapidly dissect the idea and invite review, criticism, and feedback, to establish the merit of the concept and determine whether it is worth pursuing further.

8.2 Manifold Jumping

8.2.1 Observed phenomenon

In several high Reynolds KH simulations (Re $\geq$ 50,000) with badly tuned coupling parameters, large SGNN correction episodes were followed by the simulation settling onto a flow state with measurably different vortex pairing topology. The post-jump states passed all physical validation criteria: divergence remained at machine precision ($\sim 10^{-14}$), the k^{-3} KLB cascade was preserved in the inertial range, and enstrophy stabilised within physically plausible bounds. However, the vortex core positions, pairing sequence, and large-scale circulation topology had irreversibly diverged from both the pre-correction trajectory and the baseline Eulerian solver. It is to be noted that during the correction windows, there are non spikes in enstropy and the energy cascade, but their return to correct metrics is equally quick.

Manifold jumping was observed exclusively in KH simulations and not in Kolmogorov flow. Kolmogorov flow would usually crash if ran on inconsistent simulation parameter. This is physically consistent, KH roll-up proceeds via a discrete vortex pairing sequence whose outcome depends sensitively on the phase relationship between neighbouring vortex pairs at the moment of first roll-up (Rosenhead, 1931; Michalke, 1964). A large spectral correction during this window may select a different pairing branch. Kolmogorov flow, driven by continuous sinusoidal forcing, has no analogous discrete branching point.

The post-jump states were not unphysical. They satisfied all conservation properties and spectral scaling laws. The most optimisitic interpretation is that the SGNN correction delivered a finite-amplitude perturbation large enough to re-route the simulation toward a different, but equally valid, long-time flow state of the same governing equations. The precise mechanism is not established and constitutes an open question.

8.2.2 Why are Neural Correctors Prone To Manifold Jumping?

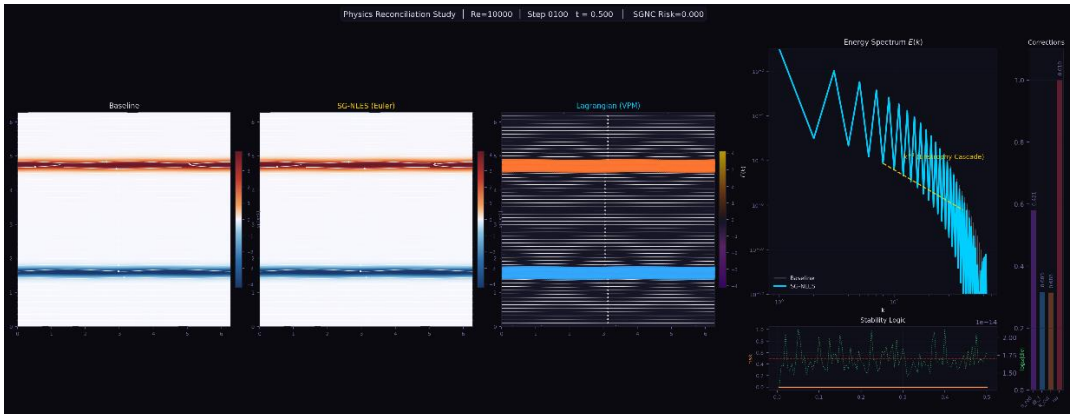
The SGNN was designed to prevent numerical blow-up, not to preserve trajectory fidelity. Its corrective scalars act globally on the vorticity field. A large correction ($\nu_{\text{nn}} \sim 0.007$, $k_{\text{cut}} \sim 0.75$) removes roughly 25% of the spectral content in a single timestep, which in a chaotic, sensitive system may be sufficient to permanently alter the long-time trajectory. This is a known risk of any global corrective intervention in a chaotic PDE system and is not unique to neural controllers. This is just a hypothesis so far. Whether an analogous effect arises in accuracy-focused neural correctors or learned subgrid models is an open question the global nature of spectral interventions suggests it cannot be ruled out.

A correction strong enough to prevent blow-up is not always also strong enough to alter the trajectory. In fact this phenomenon is extremely rare and happens at deliberately unstable flow regimes, which are just stable enough to not outright crash. A spatially localised corrector that targets only the high-k tail while preserving the phase relationships of the dominant vortex cores may stabilise the solver without redirecting the long-time state.

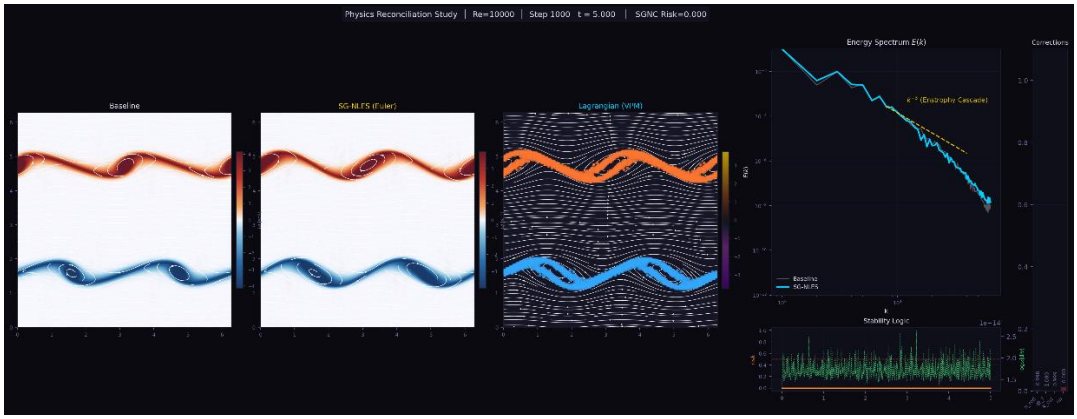
8.3 Simulation snapshots

8.3.1 KH Instability - Flow Snapshots

The following five equally spaced snapshots illustrate the temporal evolution of the Kelvin-Helmholtz instability, showing the roll-up of the initial shear layer into distinct cat-eye vortex pairs and the subsequent development of secondary structures.

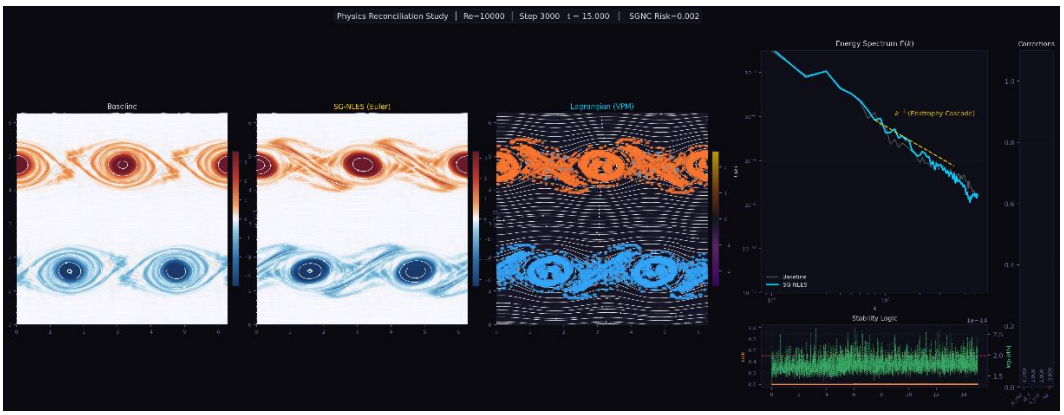


Frame 100

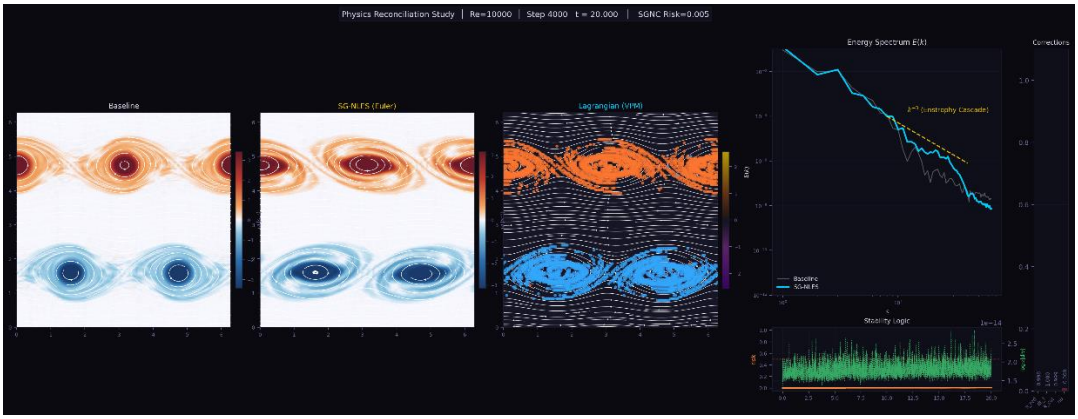


Frame 1000

Frame 2000



Frame 3000



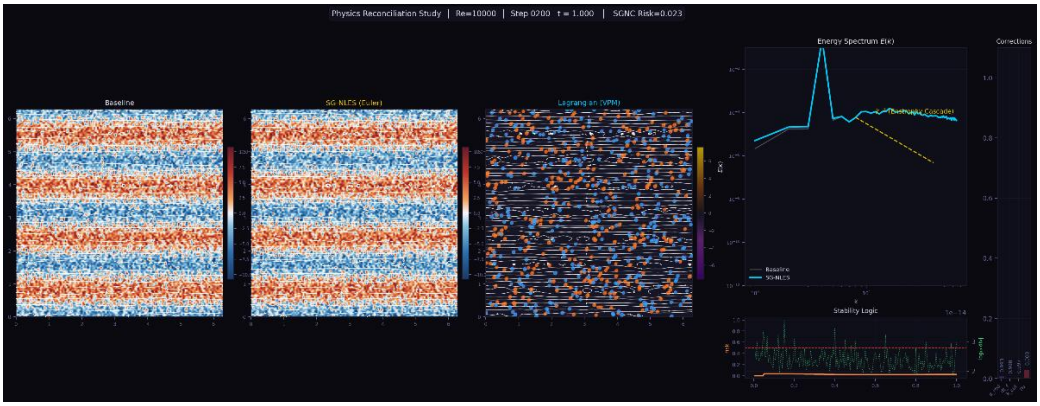
Frame 4000

Frame 5000

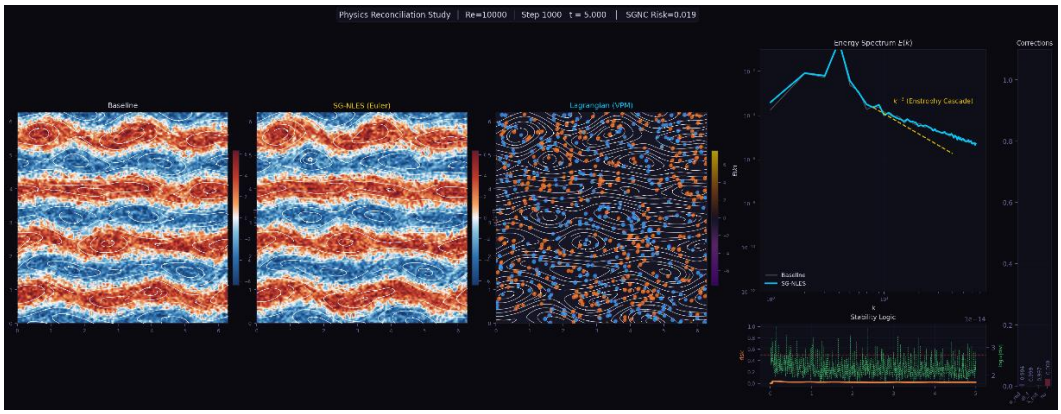
As mentioned before, the hybrid solver evolves marginally quicker than the Eulerian solver from 2000 steps onward. This disparity needs to be further researched to confidently explain this phenomenon. This also shows one of the current limitations, particles clumping towards the centre of the eyes. Remeshing helps mitigate this slightly.

8.3.2 Kolmogorov Flow - Flow Snapshots

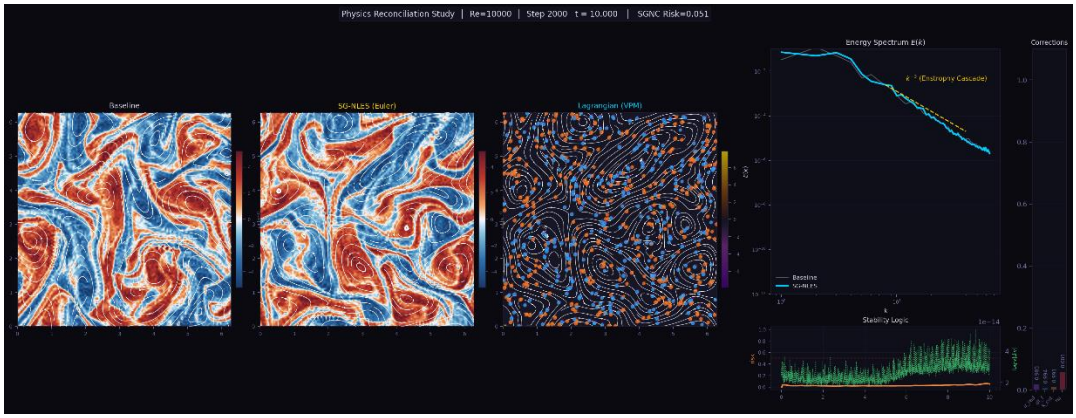
The following five equally spaced snapshots show the temporal evolution of the Kolmogorov flow, from the initial laminar forced state through the development of vortex lattices to fully developed forced turbulence.



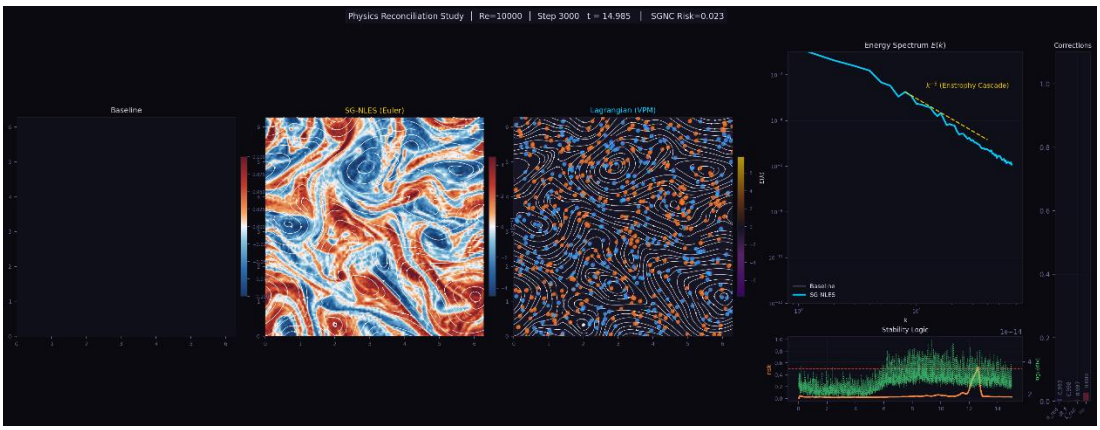
Frame 200



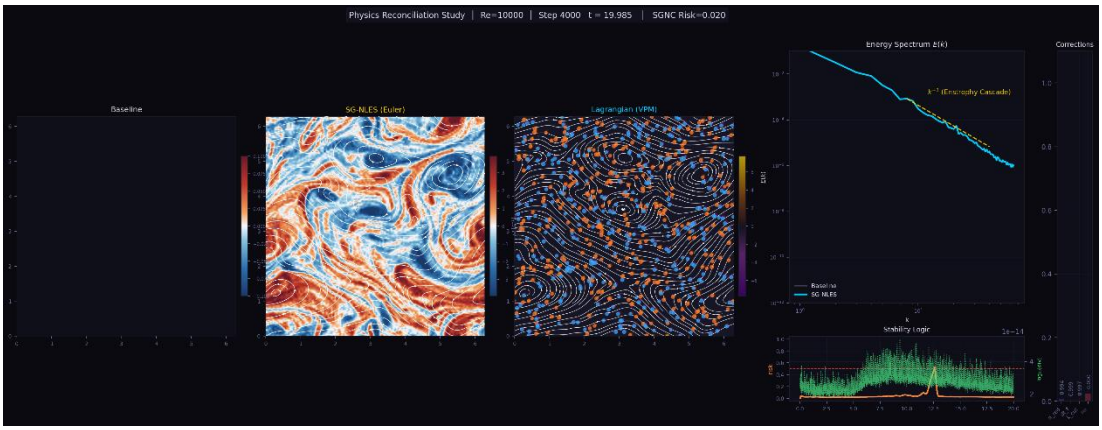
Frame 1000



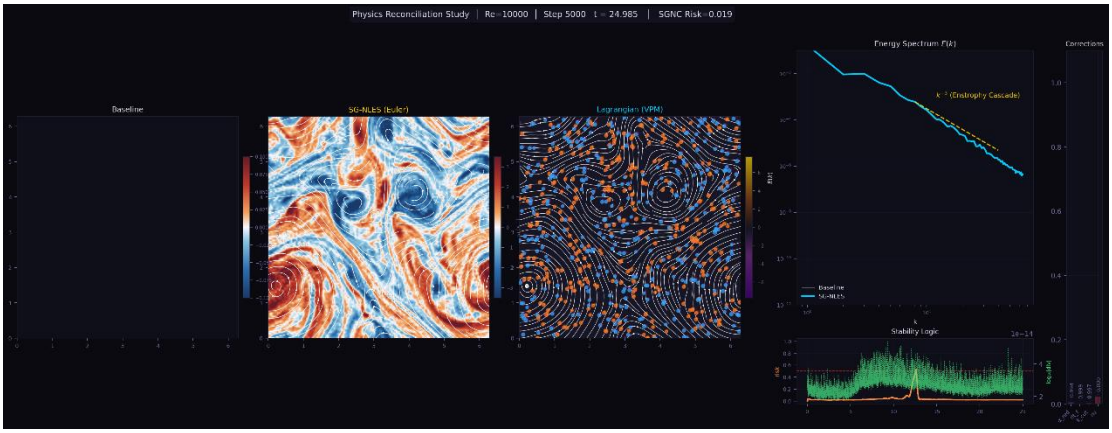
Frame 2000



Frame 3000



Frame 4000



Frame 5000

This frame snapshot shows the evolution and the eventual failure of the baseline solver and the continuation of the hybrid solver. One thing to notice is the different evolution phase of the hybrid solver from the Eulerian solver. They evolve differently but largely follow the same metrics.