Adversarial Noise Injection for Learned Turbulence Simulations

Jingtong Su
Center for Data Science
New York University
js12196@nyu.edu

Julia Kempe
Center for Data Science and
Courant Institute of Mathematical Sciences
New York University
kempe@nyu.edu

Drummond Fielding
Center for Computational Astrophysics
Flatiron Institute
drummondfielding@gmail.com

Nikolaos Tsilivis
Center for Data Science
New York University
nt2231@nyu.edu

Miles Cranmer
Department of Astrophysical Sciences
Princeton University
mcranmer@princeton.edu

Shirley Ho
Center for Computational Astrophysics
Flatiron Institute
shirleyho@flatironinstitute.org

Abstract

Machine learning is a powerful way to learn effective dynamics of physical simulations, and has seen great interest from the community in recent years. Recent work has shown that deep neural networks trained in an end-to-end manner seem capable of learning to predict turbulent dynamics on coarse grids more accurately than classical solvers. All these works point out that adding Gaussian noise to the input during training is indispensable to improve the stability and roll-out performance of learned simulators, as an alternative to training through multiple steps. In this work we bring in insights from *robust* machine learning and propose to inject *adversarial* noise to move machine learning systems a step further towards improving generalization in ML-assisted physical simulations. We advocate that training our models on these worst case perturbation instead of model-agnostic Gaussian noise might lead to better rollout and hope that adversarial noise injection becomes a standard tool for ML-based simulations. We show experimentally in the 2D-setting that for certain classes of turbulence adversarial noise can help stabilize model rollouts, maintain a lower loss and preserve other physical properties such as energy. In addition, we identify a potentially more challenging task, *driven* 2D-turbulence and show that while none of the noise-based attempts significantly improve rollout, adversarial noise helps.

1 Introduction

Simulating the dynamics of large-scale spatial and temporal data is a difficult problem in science and engineering. A representative task is predicting turbulent fluid dynamics, as turbulence is remarkably difficult to model due to its multiscale and chaotic nature: infinitesimal changes to the input may drastically alter the future state. In the last decades many outstanding numerical solvers have been designed to solve problems governed by complex partial differential equations (PDEs). Yet the price

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

to pay to obtain a high precision solution is a loss in efficiency of these tools. Thus, obtaining a better accuracy-speed trade-off is a key topic in the research community.

Recently, the use of machine learning models, especially deep learning models, has assisted in many broad and profound advances in the natural sciences. In particular, deep learning has been impressively promising in numerical simulations. Some research works use physical information to assist in neural network modeling; e.g., Wang et al. [2020] incorporate physical knowledge into neural networks to perform turbulence simulations. Sanchez-Gonzalez et al. [2018, 2020] demonstrate graph networks can effectively model the information exchange between interacting particles and obtain realistic simulation results. Kochkov et al. [2021] combine neural networks with a numerical solver to simulate turbulence. Pfaff et al. [2020] utilize graph networks to adapt the mesh-based physics model to learn dynamics for plastics. Closest to our work, Stachenfeld et al. [2022] show deep neural networks can learn to simulate turbulence at low resolution conditions with results comparable to those of high-resolution PDE solvers.

A noteworthy common thread in the above pure ML-based methods is the observation that injecting Gaussian noise into the input during training is the key to long rollout generalization for models which are trained with single steps only (rather than trained through a rollout). In this work, we aim to establish an alternative, arguably more principled avenue to noise injection for physics, bringing in insights from robust training in computer vision where one also injects informative noise to improve the quality of representations learned by the networks [Salman et al., 2020]. In particular, while Gaussian noise is generic and model-independent, adversarial noise is derived from the model itself in a worst-case fashion, and this specificity might achieve a better trade-off between robust rollout generalization and accuracy.

While widespread in computer vision, to our knowledge there are only a couple works addressing the application of adversarial noise to assist the natural sciences, Cubuk and Schoenholz [2020] and Schwalbe-Koda et al. [2021]. Both address the task of learning the energy field given a molecule configuration using neural nets, so called NN potentials. Cubuk and Schoenholz [2020] identify the existence of adversarial directions along which the NN potentials can suffer from larger loss. Schwalbe-Koda et al. [2021] build on this idea for an ensemble of NN potentials. Instead of sampling from the given molecule distribution, they adversarially attack the variance of the ensemble prediction, to get adversarial configurations. Then they sample these adversarial examples and train their models on them. This is distinct from our work: their adversarial noise is a tool to improve the sampling quality, and we inject it directly to the inputs to improve rollout generalization.

Our contributions are as follows:

1. We introduce the idea of adversarially robust machine learning to simulators for physical problems, and in particular the simulation of turbulence trajectories. To the best of our knowledge we are the first to consider ideas from the field of trained adversarial robustness for machine learning models to stabilize simulations of physical systems.
2. We conduct a set of 2D turbulence experiments based on prior work in Stachenfeld et al. [2022] to demonstrate the benefit of adversarial noise for ML simulations of astrophysical turbulence to stablization of rollout. Our results paint an encouraging picture.
3. In revisiting prior work, we identify a difficult simulation problem, namely *driven* turbulence. We present evidence that neither models trained with Gaussian noise nor with our adversarial noise can significantly stabilize long model rollouts in this regime, although adversarial noise helps more.

2 Task and Data Description

Here, we restrict ourselves to two-dimensional turbulence. Following Stachenfeld et al. [2022], we study decaying turbulence, in which an isothermal gas is initialized with a uniform density and random velocity field on a periodic domain and is allowed to evolve freely from there. We also look at driven turbulence, where energy is injected in the system, not considered in Stachenfeld et al. [2022].

The goal is to predict the turbulence configuration over a long rollout trajectory, given a specific input. Formally, for each time step $t \in \{0, 1, \ldots, T\}$, the turbulence configuration is described by a 3-dimensional tensor: $X_t \in \mathbb{R}^{C \times H \times W}$, and the task is to predict the entire trajectory $\{X_0, X_1, \cdots, X_T\}$, given X_0. Here, H and W denote the grid size on the two spatial axes, in our

case, 32. C denotes the number of features, each of which corresponds to a physical variable that we would like to predict. For general turbulence simulation, these quantities include density, velocity, and pressure. We focus on the *isothermal* process, as is common in astrophysical turbulence. In this case, $C = 3$, and the variables are the x-velocity, y-velocity and density. The associated *energy field* for this isothermal process is the kinetic energy, defined point-wise as

$$E = \frac{1}{2}\rho(v_x^2 + v_y^2), \tag{1}$$

for v_i the velocity and ρ the mass density.

We generate the data from a state-of-the-art PDE solver, Athena++ [Stone et al., 2020]. We produce high-resolution data with grid size 64x64 at a time interval of $1\,\mathrm{ms}$ ("simulation step"), and downsample them to 32x32. We sample 27 trajectories in total with evenly spaced temporal axis t from $0\,\mathrm{s}$ to $1.05\,\mathrm{s}$.[1] 24 of them are used for training, while the remaining 3 are left as a holdout set.

3 Model and Training

Following previous works, we adopt the end-to-end training pipeline to obtain our deep simulators. Following [Stachenfeld et al., 2022], we use the Adam optimizer [Kingma and Ba, 2014] with initial learning rate 10^{-4}, and train the model for 320 epochs with batch size 32. We apply temporal downsampling and fix the "model step" size Δt to $32\,\mathrm{ms}$, which is 32 times coarser than the Athena++ ground truth, i.e., one model step corresponds to 32 simulation steps.

Encode-U-Net-Decode (U-Net). U-net has been a popular benchmark and shown competitive performance in numerous previous works [Li et al., 2020, Pathak et al., 2020, Wang et al., 2020]. We use 1x1 convolution as the structure of the encoder and the decoder. The intermediate U-Net is adopted from Ronneberger et al. [2015].

Loss. We normalize the inputs X_t to be zero-mean and unit variance before training. The model receives (normalized) X_t as input, and predicts (normalized) $X_{t+\Delta t}$. We use the *Mean Square Error (MSE)* loss to optimize parameters. For independent diagnostic purposes at testing we will also compute the *Energy Field Root MSE Loss (ERMSE)*, a quantity that we do not optimize for and which measures distance to another physical ground truth. It is calculated over the pixel-wise energy field E of Eq. (1), then taking the square-root to maintain the correct unit.

Noise Injection. As pointed out by Sanchez-Gonzalez et al. [2018], Pfaff et al. [2020], Stachenfeld et al. [2022], injecting Gaussian noise to the input during training is the key to long rollout generalization. Specifically, instead of optimizing $\mathcal{L}(f(X_t;\theta), X_{t+\Delta t})$, they optimize

$$\mathcal{L}(f(X_t + \mathcal{N}(0, \sigma^2 I);\theta)), X_{t+\Delta t}). \tag{2}$$

Our approach. In this work, we propose using a different injection scheme than the ones previously considered. In computer vision, it has been observed that models trained to classify natural images are not robust to small, adversarial perturbations of the input. The most common approach to defend against malicious inputs is *adversarial training (AT)* [Madry et al., 2018]. Concretely, during training of a neural network $f(\cdot;\theta)$ with parameters θ, for each training epoch and each training sample x we first calculate its worst case perturbation δ within a certain radius of the input, and then train on $x_{adv} = x + \delta$ instead, updating the data again at the next epoch. To calculate the worst case perturbation for a sample (x, y) of a model with loss function ℓ we optimize

$$\delta = \arg\max_{||\delta|| \leq \epsilon} \ell(f(x + \delta;\theta), y), \tag{3}$$

with an iterative procedure called *projected gradient descent* [Kurakin et al., 2017] over 10 iterations. For training the machine learning model on the turbulence task with MSE loss we thus produce adversarial perturbations at every training step by iteratively optimizing[2]

$$\text{Adv\_Noise} = \arg\max_{||\delta|| \leq \epsilon} \mathcal{L}_{\text{MSE}}(f(X_t + \delta;\theta), X_{t+\Delta t}). \tag{4}$$

Here the norm can be taken as $||\cdot||_2$ or $||\cdot||_\infty$, as is standard for computer vision tasks.

[1] Following Stachenfeld et al. [2022], we discard the first 50ms of data due to the unstable nature of Athena++. We use data from $0.05\,\mathrm{s} \sim 1.05\,\mathrm{s}$ to train our model.

[2] In Stachenfeld et al. [2022], the residual $X_{t+\Delta t} - X_t$ is used as the target and noise is added to the input X_t and subtracted from the target $X_{t+\Delta t} - X_t$, which is equivalent to not changing $X_{t+\Delta t}$ in our case.

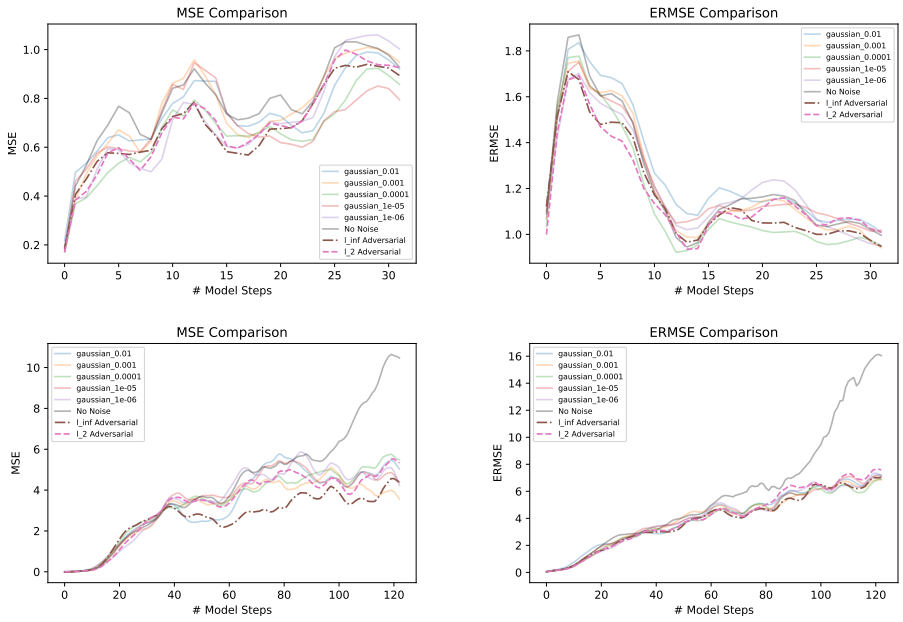

Figure 1: Test MSE and ERMSE losses, averaged over the 3 test trajectories. **Top row: Decaying Turbulence.** Models trained with adversarial noise show lower losses, especially for ERMSE, at the beginning. The noise magnitude (ϵ in Eq. (4)) of the ℓ_2 noise is 10^{-6}, and for ℓ_∞ is 10^{-4}. Note that decaying turbulence evolves towards a stationary uniform-density fluid, which could explain that ERMSE decays after a while. **Bottom row: Driven Turbulence.** Models trained with our Adversarial Noise show lower losses for longer rollouts. The noise magnitude (ϵ in Eq.(4)) of the ℓ_2 noise is 10^{-5}, and for ℓ_∞ is 10^{-4}.

4 Experiments

4.1 Adversarial Noise helps Simulating Decaying Turbulence

For both noise models we use a small grid search over the noise parameter ϵ. The first row of Figure 1 shows pixel-wise MSE and ERMSE test loss of models, averaged over the 3 test trajectories, trained with/without noise, where we have picked the best adversarial model for each ℓ_2 and ℓ_∞ noise respectively. Models trained with adversarial noise, both ℓ_2 and ℓ_∞, exhibit lower loss values, especially for ERMSE at early rollout, suggesting better energy preservation. To provide a visualization (Figure 2) we select the best Gaussian model to show several snapshots produced by our models on one of the test trajectories, trained with/without noise. Models trained with Gaussian noise and our adversarial noise can maintain quality on par with the ground-truth simulator.

4.2 Driven Turbulence

The bottom row of Figure 1 shows test MSE and ERMSE loss of models trained with/without noise on driven turbulence along 3 driven test trajectories, showing a larger number of model testing steps, since here the fluid does not become stationary. Increased loss values, as well as the visulization in Figure 3 show that driven turbulence is a more difficult task to simulate. We find that adversarial noise can somewhat contribute to rollout stabilization, while leaving significant improvements as a future challenge.

5 Conclusion

We have introduced tools from robust machine learning to increase accuracy and efficiency via learned low-resolution physical simulations that aim to replace high-resolution engineered ones. Our experimental results for 2D turbulence on the benefits of adversarial noise are encouraging, even for

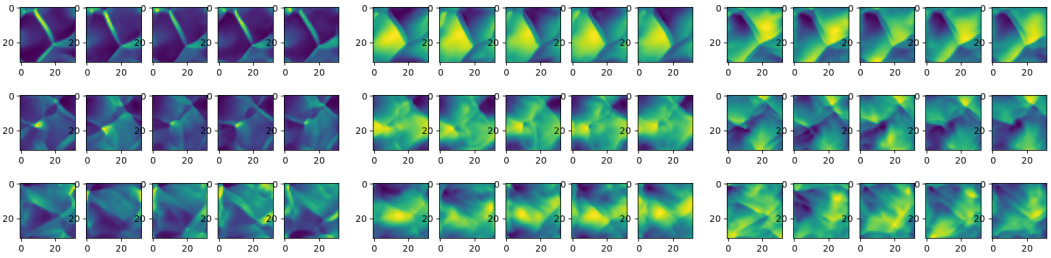

Figure 2: Decaying Turbulence Simulation Snapshots. Within each subfigure, from left to right: Ground Truth; vanilla model trained without noise; model trained with Gaussian noise; model trained with the ℓ_2 adversarial noise; and model trained with the ℓ_∞ adversarial noise. **Left**: Density. **Middle**: v_x. **Right**: v_y. From **Top** to **Bottom**: configuration at model step 10, 21, and 32 (simulation step 320, 672, and 1024.) The σ^2 in Eq. (2) of the Gaussian noise is 10^{-4}.

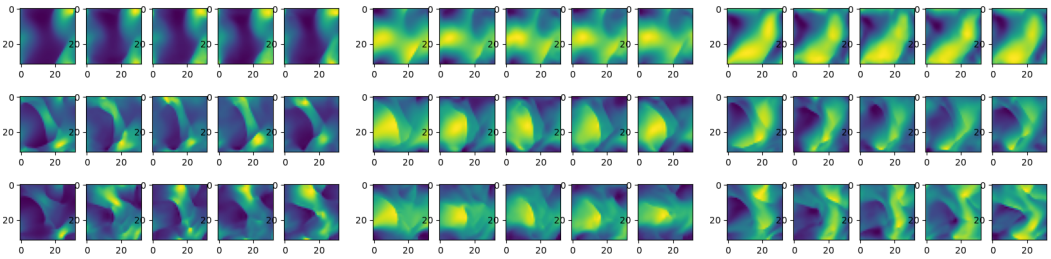

Figure 3: Driven Turbulence Simulation Snapshots. Within each subfigure, from left to right, we illustrate: Ground Truth; vanilla model trained without noise; model trained with Gaussian noise; model trained with the ℓ_2 adversarial noise; and model trained with the ℓ_∞ adversarial noise. **Left**: Density. **Middle**: v_x. **Right**: v_y. From **Top** to **Bottom**: configuration at model step 10, 20, and 30 (simulator step 320, 640, and 960.) The σ^2 in Eq. (2) of the Gaussian noise is 10^{-6}.

the more challenging setting of driven turbulence, and we hope that further studies in the 3D setting will demonstrate even higher benefits.

Broader Impact

Our work is unlikely have potential and direct ethical consequences as we discuss turbulence simulations in a pure physics context. For future societal consequences, since we empirically test how adversarial noise can assist simulation, we not only show it is beneficial for this specific task, but also bring a new tool to the physics community in general.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

References

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1457–1466, 2020.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning*, pages 4470–4479. PMLR, 2018.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pages 8459–8468. PMLR, 2020.

Dmitrii Kochkov, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2020.

Kimberly Stachenfeld, Drummond B Fielding, Dmitrii Kochkov, Miles Cranmer, Tobias Pfaff, Jonathan Godwin, Can Cui, Shirley Ho, Peter Battaglia, and Alvaro Sanchez-Gonzalez. Learned coarse models for efficient turbulence simulation. *International Conference on Learning Representations*, 2022.

Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *Advances in Neural Information Processing Systems*, 33:3533–3545, 2020.

Ekin D Cubuk and Samuel S Schoenholz. Adversarial forces of physical models. In *3rd NeurIPS workshop on Machine Learning and the Physical Sciences*, 2020.

Daniel Schwalbe-Koda, Aik Rui Tan, and Rafael Gómez-Bombarelli. Differentiable sampling of molecular geometries with uncertainty-based adversarial attacks. *Nature communications*, 12(1): 1–12, 2021.

James M Stone, Kengo Tomida, Christopher J White, and Kyle G Felker. The athena++ adaptive mesh refinement framework: Design and magnetohydrodynamic solvers. *The Astrophysical Journal Supplement Series*, 249(1):4, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Jaideep Pathak, Mustafa Mustafa, Karthik Kashinath, Emmanuel Motheau, Thorsten Kurth, and Marcus Day. Using machine learning to augment coarse-grid computational fluid dynamics simulations. *arXiv preprint arXiv:2010.00072*, 2020.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*, 2018.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.