The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5–-9 June 2022, Oslo, Norway

USING CONSERVATION LAWS TO INFER DEEP LEARNING MODEL ACCURACY OF RICHTMYER-MESHKOV INSTABILITIES

CHARLES F. JEKEL[*], DANE M. STERBENTZ, SYLVIE AUBRY, YOUNGSOO CHOI, DANIEL A. WHITE and JONATHAN L. BELOF

Lawrence Livermore National Laboratory[†], PO Box 808, Livermore, CA,94551, USA

Key words: Deep learning, Full-field regression, Richtmyer-Meshkov instability, Inference error

Abstract. Richtmyer-Meshkov Instability (RMI) is a complicated phenomenon that occurs when a shockwave passes through a perturbed interface. Over a thousand hydrodynamic simulations were performed to study the formation of RMI for a parameterized high velocity impact. Deep learning was used to learn the temporal mapping of initial geometric perturbations to the full-field hydrodynamic solutions of density and velocity. The continuity equation was used to include physical information into the loss function, however only resulted in very minor improvements at the cost of additional training complexity. Predictions from the deep learning model appear to accurately capture temporal RMI formations for a variety of geometric conditions within the domain. First principle physical laws were investigated to infer the accuracy of the model's predictive capability. While the continuity equation appeared to show no correlation with the accuracy of the model, conservation of mass and momentum were weakly correlated with accuracy. Since conservation laws can be quickly calculated from the deep learning model, they may be useful in applications where a relative accuracy measure is needed.

1 INTRODUCTION

A Richtmyer-Meshkov instability (RMI) occurs when a shockwave amplifies perturbations at a material interface, causing large jet-like growths [1, 2, 3, 4]. This phenomenon is similar to Rayleigh–Taylor (RT) instabilities that occur at the interface of fluids with different densities. An example RMI is shown forming in Figure 1 at an interface, which results in a jet that deeply penetrates the neighboring material. The first image shows the initial interface at a high velocity impact, and subsequent images show time increments of 0.1 μs.

The use and understanding of RMI is important in niche applications. For example, experimentally measuring RMI formations is useful for calibrating high strain rate material models [5, 6]. Additionally, in inertial confinement fusion (ICF) experiments lasers are used to heat a

[*]Corresponding and presenting author. jekel1@llnl.gov https://jekel.me

[†]This manuscript has been authored by Lawrence Livermore National Security, LLC under Contract No. DE-AC52-07NA2 7344 with the US. Department of Energy. The United States Government retains, and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. LLNL-CONF-837041

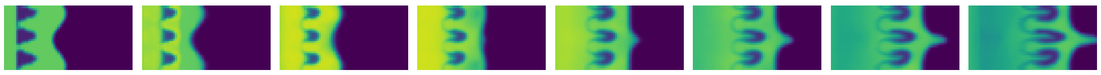
Figure 1: Snapshots of density in time increments of 0.1μs from left to right as an RMI forms.

fuel target with the hopes of starting a self-sustaining fusion reaction [7]. Unfortunately, RMI have been known to form within the ICF capsules. The RMI often destroy the fuel target before fusion ignition is achieved [8]. Thus, increasing our ability to design and control RMI could have profound impacts in fusion research.

Hydrodynamic simulations can be used to understand how initial conditions (e.g., geometric perturbations, shock velocities, materials) affect RMI growth. A designer may use optimization techniques on these simulations to select initial conditions for a desirable RMI formation [9, 10]. This requires both time and a large number of computationally expensive simulations.

This work investigates whether machine learning can be used to accurately learn how RMI form from varying initial conditions in hydrodynamic simulations. Predictions from machine learning models could be significantly cheaper than hydrodynamic simulations [11]. If the machine learning models are accurate in capturing RMI formations, then perhaps they may be useful aids for designers of complicated applications like ICF. Additionally, predictions from the model will be significantly faster than a full-fidelity hydrodynamic simulation.

Over a thousand hydrodynamic simulations were performed by varying three initial geometric parameters of a high velocity impact. The particular configuration leads to multiple RMI formations in a single simulation, where RMI formations interact with other neighboring RMI. A deep learning model was trained to predict the full density and velocity fields, using the the geometric and temporal components as inputs to the model. The model consists of the generator portion of Deep Convolutional Generative Adversarial Networks (DCGAN) [12] trained in regression.

The machine learning model is treated as a black box, and it is not possible to directly assess the accuracy of predictions without running a high-fidelity hydrodynamic simulation. Basic first principle conservation laws are investigated to asses whether they may be used as proxies for prediction accuracy. Violations in the continuity equation, conservation of mass, and conservation of momentum can be calculated on the outputs of the black box. Therefore these quantities can be quickly calculated without a full-fidelity hydrodynamic simulation. The physical violations are then compared to the traditional L^1 norm. If these physical violations are correlated to error, then they may be used in adaptive sampling techniques based on an error indicator [13]. The paper goes on to describe the parameterized hydrodynamic simulations, machine learning model, training results, model predictions, and finally the comparison between physical violations and error.

2 HYDRODYNAMIC SIMULATIONS

The geometric initial conditions of a high velocity impact was parameterized to study complex RMI formations. A copper impactor is propelled toward a stationary copper target as shown in Figure 2. The target is allowed to move freely after the impact. The impactor hits the target at a velocity of 2.0 km/s. A sinusoidal perturbation was intentionally created on both sides of the target to seed initial RMI formations. The right perturbation is defined as

$$x = 0.1 \cos\left(\frac{20\pi}{9}\right) \tag{1}$$

and the impactor side of the target is defined as

$$x = b \cos\left(\frac{2\pi q y}{9} - s\pi\right) \tag{2}$$

where three variables (b, q, s) are varied. Lucite is filled into the perturbation to make for a flush interface between target and impactor. The three parameters change the amount of material removed from the impactor side of the target, but not the end-to-end width of the target. Figure 3 shows two different initial simulation domains by changing the three parameters defining the impactor side of the target.

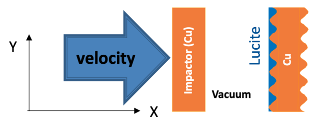

Figure 2: Experiment of copper impactor hitting copper target.

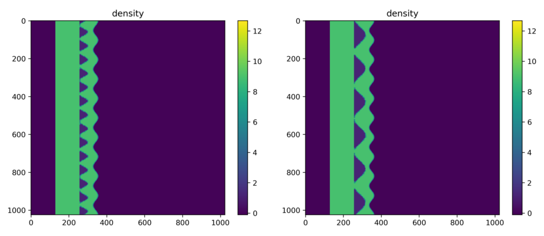

Figure 3: Two different initial conditions of the target. Left image has values of $b = 0.1714$, $q = 14.0862$, $s = 1.6492$ and right has $b = 0.2365$, $q = 5.7990$, $s = 2.8509$.

The impact was modeled using Arbitrary Lagrangian–Eulerian (ALE) hydrodynamics simulations. The ALE calculations were performed using BLAST[‡][14, 15] with second order elements. The simulations take approximately a half hour to run to completion on a single NVIDIA V100 GPU. Copper is modeled using a Steinberg-Guinan strength model [16]. The calculations start at the initial impact and end 7 μs after impact.

A random Latin hypercube sampling was performed using bounds shown in Table 1 on the three sinusoidal parameters defining the interface between target and impactor [17]. A total of 1626 simulations completed successfully. Merlin was used to asynchronously run simulations within a large HPC allocation on Lassen[*] [18].

For each simulation, the temporal full-field results were saved in 51 uniform increments from 0 μs to 7 μs. Field solutions for density ρ, velocity x, and velocity y were saved, which is only a partial view at the complex hydrodynamic state. These fields were evaluated on the high

[‡]https://computing.llnl.gov/projects/blast

[*]One node of Lassen contains one IBM Power9 with four NVIDIA V100 GPUs. https://hpc.llnl.gov/hardware/compute-platforms/lassen

Table 1: Sampling bounds on the geometric parameters defining the impactor-target interface.

Parameter	Lower bound	Upper bound
b	0.1	0.25
q	5.0	25.0
s	0.0	3.14

order elements using a fixed cartesian grid of 1024×1024. This results in a dataset of temporal full-field simulation results. The dataset consists of 1626 simulations, 3 fields, 51 time steps per field, on a 1024×1024 grid. There are 260,862,640,128 single precision floats in total taking up 0.96 TB on disk.

3 DEEP LEARNING MODEL

A machine learning problem can be described to learn the mapping from $[b, q, s, t]$ (our three sinusoidal parameters and time) to the three physical fields. The generator from the DCGAN [19] architecture was used to learn this relationship. The model uses deconvolutional layers (sometimes also called inverse convolutional, or transposed convolution [20]). Once trained, the deep learning model can be used to quickly visualize full-field solutions for any given time. Inference from a trained model may also be useful to quickly solve inverse problems (e.g., finding the temporal solution given a field, or partial solution), because the machine learning model will be many times faster than a hydrodynamic simulation.

Unlike the DCGAN work that uses both discriminator and generator, our work only uses the generator. Our data is perfectly labeled, meaning we have the small vector state that defines the results for any of our simulations. However, many GAN applications are unsupervised, and attempt to learn latent representations of the unknown vector state. Knowing the inputs to each simulation allows us to apply supervised regression to this problem. The objective of the machine learning model is to learn the mapping from this small vector state to the simulation results. As an added benefit, not having to train both discriminator and generator greatly simplifies training. Unfortunately using the generator only may result in less capability than a GAN model, since the inputs to our model may only be our parameterized initial conditions.

A separate generator model was used for each field, as shown in Figure 4. A single optimizer was used to train the three models simultaneously with a mean absolute error (L^1) loss function. The L^1 error in each physical field was normalized via the field's maximum observed range (e.g., $\max(\rho) - \min(\rho)$).

We attempted to use a single generator model with three image channels, rather than a separate generator model for each physical field. Our experiments showed that it was easy to train a model with multiple image channels per field if the physical fields were strongly correlated, then the predictions would be reasonably accurate for each field. However, performance deteriorated quickly when fields were poorly correlated. For example, a model with two image channels, one for density and one for velocity y, could only reasonably replicate one of the two fields. While using a separate model for each physical field is not necessarily the most efficient, it does allow for the learning of a wide variety of full-field solutions that do not require correlation in a single

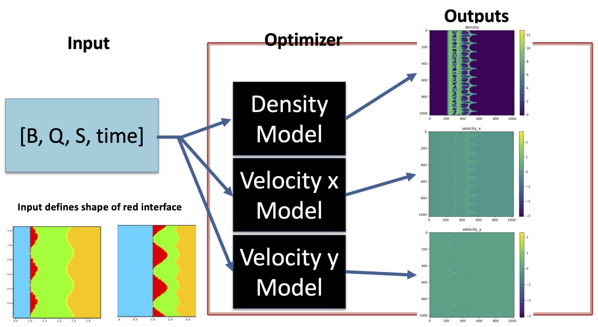

Figure 4: Overview of the deep learning model. The inputs define the sinusoidal shape of the copper target and the time after impact. Three generator models are trained in regression where each model independently learns one of the three fields. A single optimizer trains the models simultaneously. The output from each model is a 1024×1024 physical field.

training routine.

The model architecture is shown in Figure 5, again based on the generator from [19] with batch norm [21] and ReLU activations [22]. The initial kernel was 4×4, and the number of channels ranged from 512 to 32. The hyperbolic tangent function was used as the final activation, immediately followed by a linear transformation that maps $[-1, 1]$ to the range of the physical field (e.g., $[\min(\rho), \max(\rho)]$). This is used in the model because it is desirable for the outputs to be in real physical units in order to directly apply first principle conservation laws.

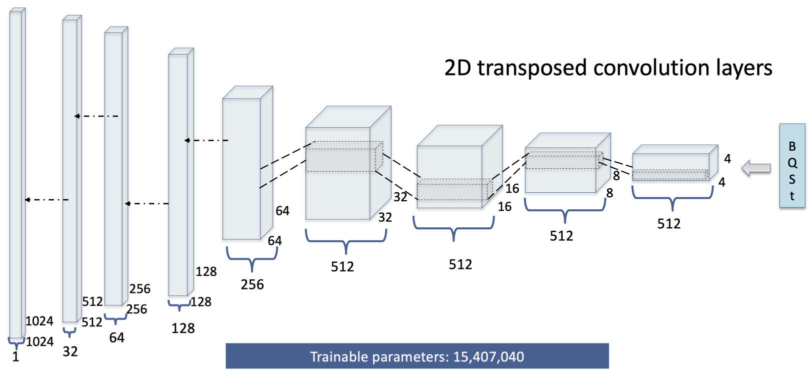

Figure 5: The convolution neural network architecture of the generator model starts with a 4×4 kernel. Each subsequent layer doubles pixels until the final 1024×1024 field is created.

The model is implemented using PyTorch [23]. Training is performed using Adam with a learning rate of 7e-4 [24]. It is possible to train the model in one hour with distributed data-parallel training [25, 26]. The models were trained for 500 epochs using 160 NVIDIA V100 GPUs on the Lassen HPC. Ten percent of the simulations were left-out of the dataset for testing.

Given that our model outputs the density, velocity x, and velocity y fields at any temporal solution, it is possible to include a physics based term in the loss function [27]. Considering the model is predicting the full density and velocity fields, a reasonable physics equation to consider is

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0 \tag{3}$$

which is known as the continuity equation [28]. Combining the general L^1 norm with the continuity equation, we arrived at the following loss function

$$L = \frac{1}{1}\sum_{i}^{n}|y_i - \hat{y}_i| + \lambda_c|\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u)| \tag{4}$$

where the left term represents the general L^1 norm, and the right term represents the L^1 penalty on the continuity equation violation. The λ_c term is a penalty parameter to help balance the two errors. General automatic differentiation is used to compute the temporal derivative in the continuity equation. The divergence is computed using using a Sobel operator (discrete spatial derivative) with Kornia [29].

4 TRAINING RESULTS AND PREDICTIONS

Training of the model was performed with and without the physics-based penalty on the continuity equation violation (i.e., $\lambda_c = 0$). Loss vs epoch curves are shown in Figure 6. The curves show the L^1 training error in density, velocity x, velocity y, and continuity equation. Despite not including the physical term in the loss function, initially the continuity equation error improves rapidly as the predictions improve. However, after approximately epoch 150 the continuity equation error gets worse as the other fields continue to improve. This is depicted in the left plot of Figure 6. When the physical error was included into the loss function, it was observed that both the physical field errors and continuity equation errors continue to improve with training. After training, the errors in density, velocity x, and velocity y are comparable with and without the continuity equation violation. However, the continuity equation violation shows over two orders of magnitude improvement when it was included into the loss function.

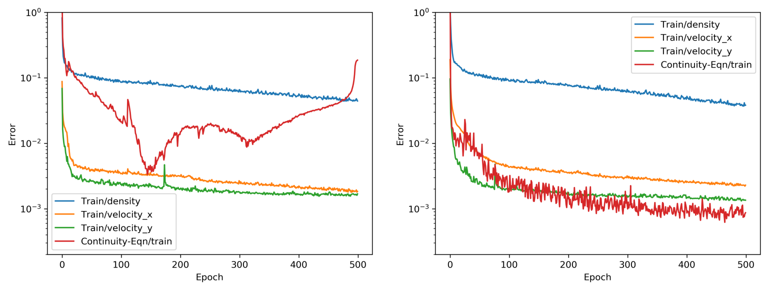

Figure 6: Training error for each physical field and continuity equation violation. Left shows no continuity equation penalty (i.e., $\lambda_c = 0$), right includes the physics informed penalty using $\lambda_c = 0.001$.

Finding an appropriate penalty parameter is crucial to successfully apply the physical constraint. If the penalty parameter was too large, then the deep learning model would quickly solve the continuity violation by trivially predicting zero velocity and zero density everywhere. Table 2 shows the L^1 error on the physical fields of the left-out testing data and different λ_c values. It is seen that as λ_c increases, the L^1 error on the physical predictions also increased. Despite improved continuity equation violations, the predictions in density and velocity actually worsened. While technically the most accurate model occurred with λ_c =1e-4, it was only marginally better than not including the continuity equation at all ($\lambda_c = 0$). It should also be noted that too large of a penalty parameter (i.e $\lambda_c \geq 0.1$) would result in predictions that were less accurate than not including the continuity equation at all.

Table 2: Table of MAE on the left-out testing fraction of the dataset for different λ_c penalty parameters.

LR	λ_c	MAE (L^1) Test
7e-4	0.0	0.005584
7e-4	1e-4	0.005537
7e-4	1e-3	0.005746
7e-4	1e-2	0.005927
7e-4	1e-1	0.006209

A continuation method could potentially work well to automatically adjust the penalty parameter during training. At early epochs the penalty could be ignored, and then gradually increased as training progresses. This was not attempted, as our current scope was to investigate whether the physics violations infer model inadequacy, however the continuation approach may help to avoid the manual tunning of a penalty parameter in future work.

The deep learning model's density predictions are compared to simulation data at four randomly selected points from the left-out set in Figure 7. Predictions from penalty parameters ranging from $\lambda_c = 0.0$ to $\lambda_c = 10^{-2}$ all appear quite similar to each other, and all reasonably agree with the simulation results. There is some noticeable fine detail loss when comparing model predictions to the simulation data, which is perhaps most noticeable in the second column where small holes form in the copper impactor. The predictions from a larger penalty parameter of $\lambda_c = 10^{-1}$ are noticeably worse than the smaller penalties. This agrees with the mean absolute error scores in Table 2, where too large of an emphasis on the model's ability to satisfy the continuity equation degrades performance.

The deep learning model has several advantages over the full hydrodynamic simulations. For instance, the model is capable of predicting any instance in time, while the hydrodynamic solutions need to be solved from the initial impact time. Using a single NVIDIA V100 GPU, inference from the model takes only 7 ms, while the full hydrodynamic simulation takes 30 minutes. The deep learning model can be run quickly on laptop CPU, where the dataset of simulation results and hydrodynamic code require HPC resources. These advantages do come at a cost of reduced accuracy and detail loss when compared to a full-fidelity hydrodynamic solution. The least accurate predictions were noted to occur in extrapolation at the corners of our sampling domain.

5 CORRELATIONS OF ERROR AND PHYSICAL VIOLATION

Scatter plots of the continuity equation violation vs L^1 error of the predictions in the left-out simulations is shown in Figure 8 for $\lambda_c = 0.0$ and λ_c =1e-3. There does not appear to be a correlation between the continuity equation violation and the error in the predictions of density and velocity. The results for all other λ_c values followed a similar pattern. This is unfortunate

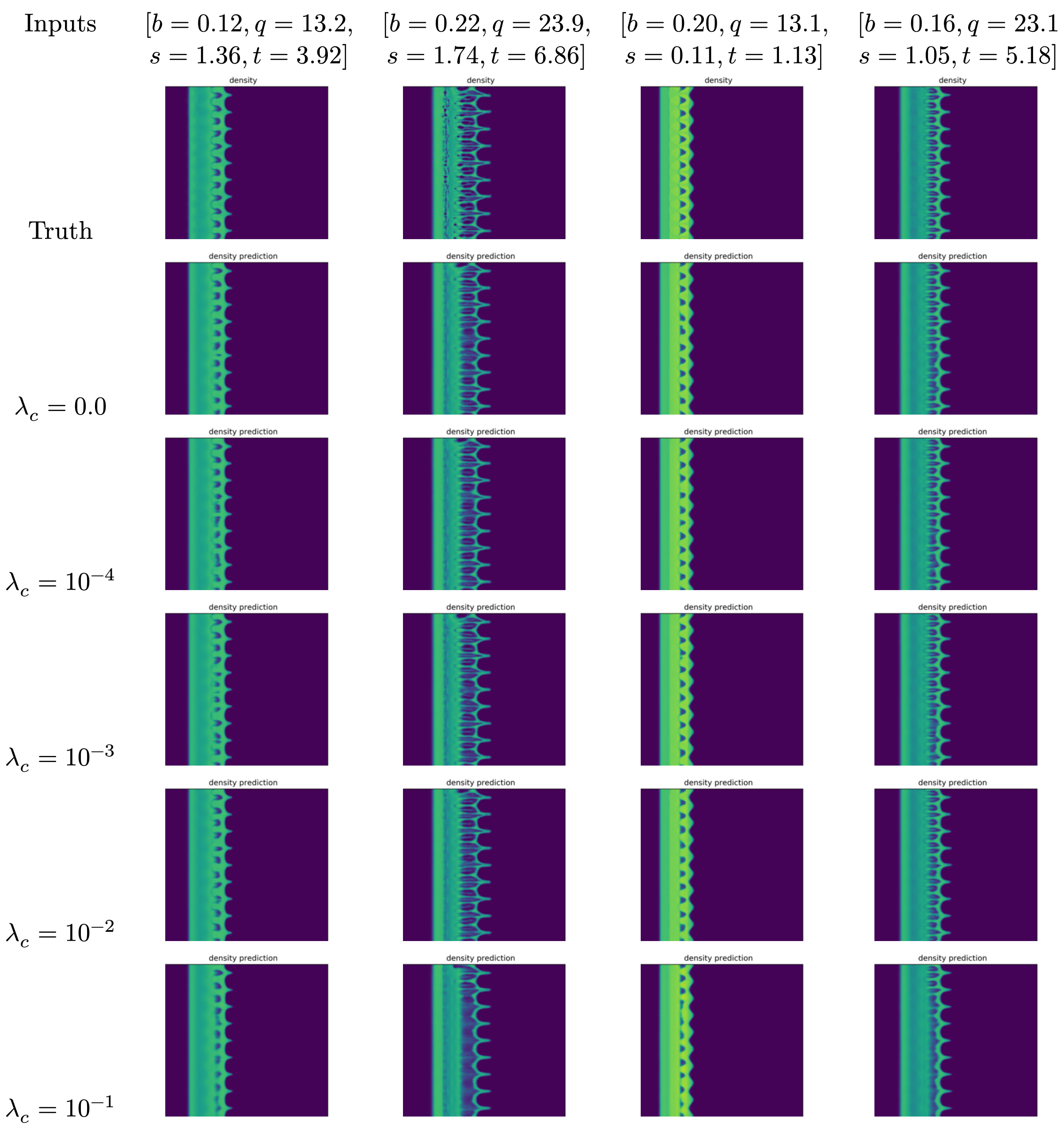

Figure 7: Figure of predictions and truth for four random points in the left-out set. Predictions are shown with λ_c penalty values ranging from 0.0 to 10^{-1}.

because if the two values were strongly correlated, then the continuity equation violation could perhaps be used to assess prediction accuracy. The correlation coefficient was 0.08 without the physics penalty and -0.06 with the penalty.

Since the simulation was computed on a closed domain (i.e., no inflow or outflow of mass), it is also possible to compute the temporal violation in conservation of mass and momentum.

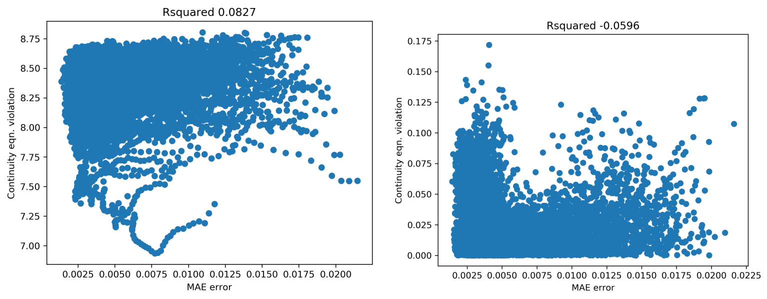

Figure 8: Continuity equation violation vs mean absolute error on the left-out simulations. Left plot shows no continuity equation penalty (i.e., $\lambda_c = 0$), right includes the physics informed penalty using $\lambda_c = 0.001$.

Thus for a given input condition $[b, q, s]$, the total mass in this model can be expressed as

$$m(t) = \frac{1}{n_y}\frac{1}{n_x}\sum_{i}^{n_y}\sum_{j}^{n_x}\rho_{i,j}(t) \tag{5}$$

because the volume is fixed. Additionally the total momentum can be expressed as

$$p(t) = \frac{1}{n_y}\frac{1}{n_x}\sum_{i}^{n_y}\sum_{j}^{n_x}\nabla\cdot\big(\rho_{i,j}(t)u_{i,j}(t)\big) \tag{6}$$

where u is the velocity field. An ideal machine learning model would preserve the conservation of mass and momentum to a comparable level of the hydrodynamic simulation.

The mass and momentum should be conserved quantities with respect to time. Thus, one way to assess how well the deep learning model conserves these quantities is to study the temporal variance in mass and momentum. The variance of a quantity $\psi(t)$ is expressed as

$$\mathrm{Var}(\psi(t)) = \frac{1}{n_t}\sum_{i}^{n_t}\Big(\psi(i) - \mathrm{Mean}(\psi(t))\Big)^2 \tag{7}$$

where the summation is over the 51 temporal realizations. A zero value for mass variance or momentum variance would be an indication that the model is conserving the quantity with respect to time.

The variance in conservation of mass is shown against the prediction error in Figure 9. Whether the physical penalty on the continuity equation was included in the loss function made little overall difference. Both cases resulted in weak correlations of 0.2.

The variance in conservation of total momentum is shown against the prediction error in Figure 10. There was very little difference when the the physical penalty was included into the loss function. Like with conservation of mass, it seems that using conservation of momentum is also a poor indicator of the prediction accuracy. The correlation coefficients were only around 0.15.

These weak correlations unfortunately indicate that variance in mass and momentum are poor indicators of the prediction accuracy. It was desired to have a cheap and reasonable assessment of prediction accuracy using only first principle physics. These types of error assessments could be

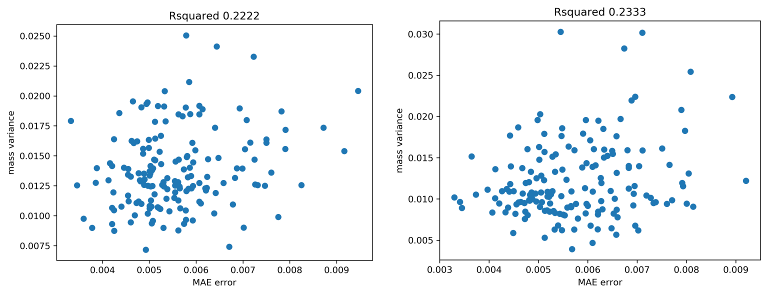

Figure 9: Conservation of mass reflected as the temporal variance vs mean absolute error on the left-out simulations. Left plot shows no continuity equation penalty (i.e., $\lambda_c = 0$), right includes the physics informed penalty using $\lambda_c = 0.001$.

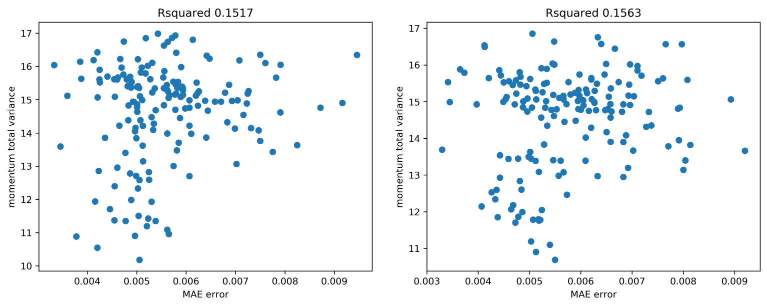

Figure 10: Conservation in total momentum reflected as the temporal variance vs mean absolute error on the left-out simulations. Left plot shows no continuity equation penalty (i.e., $\lambda_c = 0$), right includes the physics informed penalty using $\lambda_c = 0.001$.

computed without needing to run full hydrodynamic simulation. However, perhaps the physical violations may still be useful as an error indicator in an adaptive sampling strategy similar to gLaSDI [30], where larger physical violations could indicate regions of the design space requiring additional sampling.

6 CONCLUSION

Machine learning models can produce reasonably accurate RMI predictions, that are significantly cheaper than running high-fidelity hydrodynamic simulations. The continuity equation can be included into the loss function for training machine learning models that are based on predicting density and velocity fields. A penalty parameter is needed to balance the contribution from both the physical and accuracy terms in the loss function. While it is common belief that the inclusion of physical knowledge into machine learning models will result in increased accuracy, our results leave much to be desired. The inclusion of the continuity equation into the loss function did not offer much overall improvement in the accuracy of predictions, while significantly increasingly training complexity and overhead. It is unclear why satisfying the continuity equation did not improve predictions. Our application did have a large amount of physics-based data, thus perhaps the inclusion of physical terms like the continuity equation could only offer marginal improvement over data driven methods.

For certain applications, it is important to assess the accuracy of black box predictions.

Violations in first principle physical laws should be able to infer the accuracy of a machine learning model that predicts physics-based solutions. Unfortunately it does not appear that violations in the continuity equation, conservation of mass, or conservation of momentum can be used to strongly assess the accuracy of a model predicting RMI formations.

Acknowledgment

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and was supported by the LLNL-LDRD Program under Project No. 21-SI-006. The authors would like to thank Tom Stitt for his early contributions on learning Rayleigh-Taylor instabilities which inspired much of this work. The authors would like to thank Rob Rieben for his assistance with running hydrodynamic simulations.

References

[1] Q. Chen, L. Li, Y. Zhang, and B. Tian, “Effects of the atwood number on the richtmyer-meshkov instability in elastic-plastic media,” *Phys. Rev. E*, vol. 99, p. 053 102, 5 May 2019.

[2] H.-S. Park *et al.*, “Viscous rayleigh-taylor instability experiments at high pressure and strain rate,” *Phys. Rev. Lett.*, vol. 104, p. 135 504, 13 Apr. 2010. DOI: `10.1103/PhysRevLett.104.135504`.

[3] Z. Sternberger *et al.*, “A comparative study of rayleigh-taylor and richtmyer-meshkov instabilities in 2d and 3d in tantalum,” *AIP Conference Proceedings*, vol. 1793, no. 1, p. 110 006, 2017. DOI: `10.1063/1.4971669`.

[4] W. T. Buttler *et al.*, “Unstable richtmyer–meshkov growth of solid and liquid metals in vacuum,” *Journal of Fluid Mechanics*, vol. 703, pp. 60–84, 2012. DOI: `10.1017/jfm.2012.190`.

[5] G. Dimonte *et al.*, “Use of the richtmyer-meshkov instability to infer yield stress at high-energy densities,” *Phys. Rev. Lett.*, vol. 107, p. 264 502, 26 Dec. 2011. DOI: `10.1103/PhysRevLett.107.264502`.

[6] J. L. Belof *et al.*, “Rayleigh-taylor strength experiments of the pressure-induced $\alpha \rightarrow \varepsilon \rightarrow \alpha$‘ phase transition in iron,” *AIP Conference Proceedings*, vol. 1426, no. 1, pp. 1521–1524, 2012. DOI: `10.1063/1.3686572`.

[7] A. B. Zylstra *et al.*, “Burning plasma achieved in inertial fusion,” *Nature*, vol. 601, no. 7894, pp. 542–548, 2022. DOI: `10.1038/s41586-021-04281-w`.

[8] T. Desjardins *et al.*, “A platform for thin-layer richtmyer-meshkov at omega and the nif,” *High Energy Density Physics*, vol. 33, p. 100 705, 2019, ISSN: 1574-1818. DOI: `https://doi.org/10.1016/j.hedp.2019.100705`.

[9] D. Sterbentz, C. Jekel, D. White, S. Aubry, and J. Belof, “Design optimization for richtmyer—meshkov instability suppression,” *Bulletin of the American Physical Society*, 2022.

[10] D. Sterbentz, C. Jekel, D. White, S. Aubry, and J. Belof, “Design optimization for richtmyer—meshkov instability suppression at shock-compressed material interfaces,” *Physics of Fluids*, 2022.

[11] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. DOI: `10.1038/nature14539`.

[12] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv:1511.06434*, 2015.

[13] Y. Choi, G. Boncoraglio, S. Anderson, D. Amsallem, and C. Farhat, “Gradient-based constrained optimization using a database of linear reduced-order models,” *Journal of Computational Physics*, vol. 423, p. 109 787, 2020, ISSN: 0021-9991. DOI: `https://doi.org/10.1016/j.jcp.2020.109787`.

[14] R. W. Anderson, V. A. Dobrev, T. V. Kolev, R. N. Rieben, and V. Z. Tomov, “High-order multimaterial ale hydrodynamics,” *SIAM Journal on Scientific Computing*, vol. 40, no. 1, B32–B58, 2018. DOI: `10.1137/17M1116453`.

[15] R. Rieben, "The marbl multi-physics code," in *Exascale Computing Project Annual Meeting*, Feb. 2020. DOI: 10.13140/RG.2.2.12326.14403.

[16] D. J. Steinberg, S. G. Cochran, and M. W. Guinan, "A constitutive model for metals applicable at high-strain rate," *Journal of Applied Physics*, vol. 51, no. 3, pp. 1498–1504, 1980. DOI: 10.1063/1.327799.

[17] F. A. Viana, G. Venter, and V. Balabanov, "An algorithm for fast optimal latin hypercube design of experiments," *International journal for numerical methods in engineering*, vol. 82, no. 2, pp. 135–156, 2010.

[18] J. L. Peterson *et al.*, "Enabling machine learning-ready hpc ensembles with merlin," *Future Generation Computer Systems*, vol. 131, pp. 255–268, 2022.

[19] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016.

[20] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, e3, 2016.

[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, PMLR, 2015, pp. 448–456.

[22] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Icml*, 2010.

[23] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] P. Goyal *et al.*, "Accurate, large minibatch sgd: Training imagenet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[26] S. A. Jacobs, N. Dryden, R. Pearce, and B. Van Essen, "Towards scalable parallel training of deep neural networks," in *Proceedings of the Machine Learning on HPC Environments*, ser. MLHPC'17, Denver, CO, USA: Association for Computing Machinery, 2017, ISBN: 9781450351379. DOI: 10.1145/3146347.3146353.

[27] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.10.045.

[28] R. M. Colombo, M. Herty, and M. Mercier, "Control of the continuity equation with a non local flow," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 17, no. 2, pp. 353–379, 2011.

[29] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski, "Kornia: An open source differentiable computer vision library for pytorch," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 3674–3683.

[30] X. He, Y. Choi, W. D. Fries, J. Belof, and J.-S. Chen, "Glasdi: Parametric physics-informed greedy latent space dynamics identification," *arXiv preprint arXiv:2204.12005*, 2022.