Splash in a Flash: Sharpness-aware minimization for efficient liquid splash simulation

Vishrut Jetly, Hikaru Ibayashi and Aiichiro Nakano

Department of Computer Science, University of Southern California

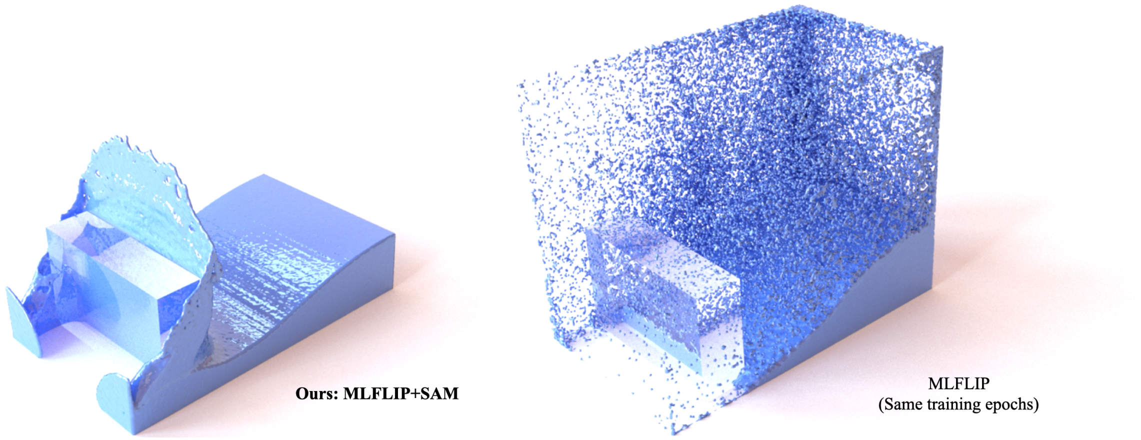

Figure 1: *Simulation results with the same number of training epochs. Our training scheme with sharpness-aware minimization quickly learns physical fluid dynamics (left) while the baseline result (right) with naive training scheme shows numerous unphysical liquid splashes.*

Abstract

We present sharpness-aware minimization (SAM) for fluid dynamics which can efficiently learn the plausible dynamics of liquid splashes. Due to its ability to achieve robust and generalizing solutions, SAM efficiently converges to a parameter set that predicts plausible dynamics of elusive liquid splashes. Our training scheme requires 6 times smaller number of epochs to converge and, 4 times shorter wall-clock time. Our result shows that sharpness of loss function has a close connection to the plausibility of fluid dynamics and suggests further applicability of SAM to machine learning based fluid simulation.

CCS Concepts

• ***Animation*** → *Fluid simulation;* • ***Methods and Applications*** → *Machine Learning; Neural Nets ; Optimization;* • ***Visualization*** → *Scientific Visualization;*

1. Introduction

Owing to the prevailing machine learning techniques, fluid simulation has witnessed drastic performance improvements in recent years [LJS*15, UB18]. However, due to large three-dimensional datasets, the training remains a bottleneck of machine learning based fluid dynamics (e.g. Table 1 in [KAT*19]). In this work, we introduce sharpness-aware minimization (SAM) [FKMN20] to machine learning based FLIP (MLFLIP) [UHT18] and show that SAM drastically reduces the training cost while capturing plausible behavior of liquid splashes (Fig. 1 and Table 1).

2. Sharpness-aware minimization

Sharpness-aware minimization (SAM) is a training scheme originally developed for image classification [FKMN20]. We here summarize the essential of SAM. Given training data $(\mathbf{x},\mathbf{y}) = \{(x_1,y_1),\ldots,(x_n,y_n)\}$, weight parameters w and loss function $\ell(\cdot,\cdot)$, training algorithms are designed to minimize the training loss, $L_w(\mathbf{x},\mathbf{y}) := \frac{1}{n}\sum_{i=0}^{n}\ell((x_i,y_i),w)$. Intuitively, sharpness is the curvature of loss surface $L_w(\mathbf{x},\mathbf{y})$, which is known to be highly correlated to the performances [JNM*19]. Specifically, SAM defines sharpness as follows

$$\max_{\|\varepsilon\|_2\le\rho}\{L_{w+\varepsilon}(\mathbf{x},\mathbf{y}) - L_w(\mathbf{x},\mathbf{y})\}, \tag{1}$$

where ρ is a hyper-parameter. SAM minimizes this sharpness along with the training loss $L_w(\mathbf{x},\mathbf{y})$ by the following update rule

$$w = w - \eta\left(\nabla_w L_w(\mathbf{x},\mathbf{y}) + \nabla_{w'} L_{w'}(\mathbf{x},\mathbf{y})|_{w'=w+\rho\frac{\nabla_w L_w(\mathbf{x},\mathbf{y})}{\|\nabla_w L_w(\mathbf{x},\mathbf{y})\|}}\right), \tag{2}$$

as a combination of the first-order gradients of loss function, i.e., $\nabla_w L_w(\mathbf{x},\mathbf{y})$. In image classification tasks, SAM showed drastic im-

DOI: 10.2312/egp.20221003

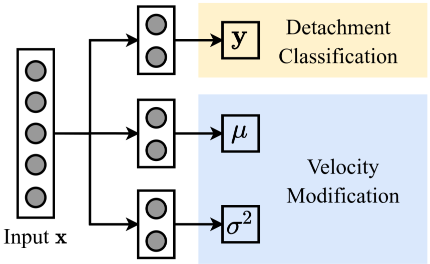

Figure 2: *Neural network layout used in this study. Detachment classifier determines whether a splash has been formed or not; and for the inputs which are classified as splash, velocity modification component of this network predicts the mean and the variance of the probability distribution for changes in the velocity.*

provement of generalization performance and robustness against noise [FKMN20]. Further, due to its off-the-shelf algorithm design, SAM has started spreading beyond its scope, including physics simulation [IiNR*21].

As a shortcoming, it is known that SAM has twice as heavy time complexity. But as we show in the following, SAM's suitable inductive bias overcomes this shortcoming in the training of fluid dynamics.

3. MLFLIP with SAM

MLFLIP is a data-driven splash generation model proposed in [UHT18], which adopts machine learning to predict the locations of fluid splashes instead of using high resolution FLIP scheme [ZB05]. MLFLIP procedure consists of several steps: synthetic data generation; feature engineering; and building suitable neural network architecture followed by training and evaluation. In the following subsections, we provide a detailed description of each step.

Data Generation The training data are generated through multiple high resolution FLIP simulations. These simulations are initialized with random values for the number of droplets and their positions and velocities to ensure sufficient variance in the generated data. From these simulations, we extract feature vector $\mathbf{x}$ consisting of 108 components having 27×3 velocity values and 27×1 level set values. In all, we have used 10^6 such samples from 16 simulations using a grid spacing of 5 mm, with even distribution of both splashing and non-splashing particles for neural network training.

Neural Network Architecture Input to our network is a feature vector $\mathbf{x}$, which has the information about the flow at a particular position. From this, the neural network will predict two components: 1) detachment classification, which determines whether the region will detach and form a splash or not; and 2) velocity modification, which determines the velocity change for a splash with respect to the fluid motion as shown in Figure 2. To estimate velocity modification, we predict both mean and variance of the velocities. The neural network has a separate 64-neuron hidden layer with 10% dropout for each of these prediction values, with *tanh* as non-linear activation function, followed by batch normalization layer.

	Epochs to converge	Wall-clock time
MLFLIP	320 epochs	2144 sec
MLFLIP + SAM (Ours)	**50** epochs	**569** sec

Table 1: *Performance comparison of ML based FLIP (MLFLIP) [UHT18] without and with SAM optimization. Our training scheme drastically improves convergences speed both in number of epochs and wall-clock time while it simulates plausible liquid splash effect (Fig. 1 left)*

These are in turn connected to the single output neuron. Activation for the detachment classifier output neuron is *sigmoid*.

Our neural network takes feature vector $\mathbf{x} = \{x_1, x_2, \ldots x_n\}$ and estimates the probability of it being a splash. Hence, we maximize the likelihood, i.e. minimize the negative log likelihood

$$L_d(\hat{\mathbf{y}}|\mathbf{x}) = -\sum_{i=1}^{n} \log P(\hat{y}_i|x_i) \quad (3)$$

where $\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n\}$ are the actual splash indicator values. We use the cross entropy loss for this classification part of our neural network. For the velocity change in the droplet, we assume it to follow a normal distribution relative to the mean flow of fluid.

$$f_v(\Delta\mathbf{v}_i|x_i) \sim \mathcal{N}(\Delta\mathbf{v}_i|\mu_i, \sigma_i^2) \quad (4)$$

where μ_i and σ_i^2 denote the mean and the variance respectively. Thus, we minimize the loss function L_v calculated as

$$L_v(\Delta\mathbf{v}|\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{d}\left[\frac{(\Delta\mathbf{v}_{i,j} - \mu_{i,j})^2}{\sigma_{i,j}^2} + \ln\sigma_{i,j}^2\right] \quad (5)$$

where j is the spatial index.

We apply SAM to the overall loss function $L_d + L_v := L_w$, using equation 2. The neural network is trained by minimizing this loss using Adam optimizer [KB14] with a learning rate of 10^{-4} and exponential decay rates of the first moment (β_1) and second moment (β_2) as 0.9 and 0.999 respectively. The models were implemented with Tensorflow (2.5.0) backend [AAB*15] having GPU support.

4. Results and Future direction

We trained two instances of the neural network: 1) MLFLIP, reproduced from [UHT18]; and 2) MLFLIP+SAM that utilizes SAM optimization for MLFLIP. Weights of both neural networks were initialized with normally distributed random values. The neural networks then iteratively learn to capture realistic behavior of the fluids and gradually converge to faithfully represent the underlying physics of droplet formation. The number of epochs and wall-clock time taken by both neural networks are shown in Table 1. Optimization with SAM helps the model to rapidly converge, achieving a speedup of 3.76x over MLFLIP while showing visual plausibility (Fig. 1 left). These promising results indicate the potential of SAM for fluid simulation and as a future direction, we will explore use of SAM for broader application of fluid dynamics such as pressure solvers and generative fluid models.

5. Acknowledgement

This work was supported by the National Science Foundation, CyberTraining award OAC 2118061.

References

[AAB*15] ABADI M., AGARWAL A., BARHAM P., BREVDO E., CHEN Z., CITRO C., CORRADO G. S., DAVIS A., DEAN J., DEVIN M., GHEMAWAT S., GOODFELLOW I., HARP A., IRVING G., ISARD M., JIA Y., JOZEFOWICZ R., KAISER L., KUDLUR M., LEVENBERG J., MANÉ D., MONGA R., MOORE S., MURRAY D., OLAH C., SCHUSTER M., SHLENS J., STEINER B., SUTSKEVER I., TALWAR K., TUCKER P., VANHOUCKE V., VASUDEVAN V., VIÉGAS F., VINYALS O., WARDEN P., WATTENBERG M., WICKE M., YU Y., ZHENG X.: TensorFlow: Large-scale machine learning on heterogeneous systems. https://www.tensorflow.org/, 2015. Software available from tensorflow.org. 2

[FKMN20] FORET P., KLEINER A., MOBAHI H., NEYSHABUR B.: Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412* (2020). 1, 2

[IiNR*21] IBAYASHI H., ICHI NOMURA K., RAJAK P., TAUFEQ MOHAMMED A. M., KRISHNAMOORTHY A., NAKANO A.: Sharpness-aware minimization for robust molecular dynamics simulations. *Proc NeurIPS MLPS* (2021). 2

[JNM*19] JIANG Y., NEYSHABUR B., MOBAHI H., KRISHNAN D., BENGIO S.: Fantastic generalization measures and where to find them. *arXiv preprint arXiv:1912.02178* (2019). 1

[KAT*19] KIM B., AZEVEDO V. C., THUEREY N., KIM T., GROSS M., SOLENTHALER B.: Deep fluids: A generative network for parameterized fluid simulations. In *Computer Graphics Forum* (2019), vol. 38, Wiley Online Library, pp. 59–70. 1

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 2

[LJS*15] LADICKỲ L., JEONG S., SOLENTHALER B., POLLEFEYS M., GROSS M.: Data-driven fluid simulations using regression forests. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 1–9. 1

[UB18] UMETANI N., BICKEL B.: Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics (TOG) 37*, 4 (2018), 1–10. 1

[UHT18] UM K., HU X., THUEREY N.: Liquid splash modeling with neural networks. In *Computer Graphics Forum* (2018), vol. 37, Wiley Online Library, pp. 171–182. 1, 2

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 965–972. 2