I. INTRODUCTION

Numerically integrating fluid equations on a coarse grid relative to a fully resolved integration of a given problem (coarse-graining) is of both historical and current interest since it can significantly reduce the computational time required for a given simulation. With the advent of modern machine learning methods, a number of attempts have been made to develop improved coarse-grained models, including for 3D Eulerian and Navier-Stokes turbulence, buoyancy-driven, variable density turbulence, and molecular-level simulations.

In this paper, our focus will be machine learning an accurate local integrator for the coarse-grained, 1-D viscous Burgers’ equation. Burgers’ equation is the simplest fluid equation admitting shocks and is exactly integrable. Numerically integrating shocks is complicated by the Gibbs effect, where, when a discontinuity develops, unstable oscillations develop in a numerical simulation. This behavior may be corrected using shock capturing methods. Shock capturing methods rely on flux limiters - non-linear interpolations between high- and low-resolution integration schemes used in a numerical simulation to keep a shock solution monotonic, thereby eliminating spurious oscillations.

Harten provided a framework for constructing non-linear, monotonicity preserving flux limiters. At present, a large number of flux limiters have been defined and used in the literature. Different forms of flux limiter have been shown to have differing accuracy and convergence performance. And criteria have been studied for determining what parameter regions are appropriate for different classes of limiter.

Since Burgers’ equation is exactly integrable, it is straightforward to generate accurate, high-resolution data from which flux limiters may be tested using a machine learning approach. This setting also provides a good testing ground for developing methodology and evaluating which flux limiters are the best (for one particular equation, or potentially for multiple shock-forming equations), and for comparing existing flux limiters with those learned from data.

Below, we will introduce the Cole-Hopf solution to the inviscid Burgers’ equation, show how to use it to test DNS training data, and define our flux limiters. We will then present a cost function allowing us to compare predicted outcomes from a discretized flux limiter with high-resolution DNS training data. Next, we will optimize the cost function for a given set of data resulting in a regression analysis solvable with standard linear algebraic methods. We then apply this analysis to a training dataset to learn optimal flux limiters from coarse-grained data. We compare these optimized, coarse-grained flux limiters with a large set of other flux limiters and show that they outperform them over the set of coarse-grainings. We finally incorporate our regression analysis within a larger, hyper-parameter optimization framework to find an optimal discretization of the flux limiter over a range of parameters, including coarse-graining, number of discretized bins, and diffusion parameter. This demonstrates the ability to produce flux limiters that should be more broadly useful than standard limiters for general applications.

II. BACKGROUND

A. Exact Solution of Inviscid Burgers’ Equations

The viscous Burgers’ equation written in conservative form is

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left( \frac{u^2}{2} - \nu \frac{\partial u}{\partial x} \right) = 0 .$$

(1)

It may be solved exactly with the Cole-Hopf transform,

$$u = -2\nu \frac{1}{\phi} \frac{\partial \phi}{\partial x} .$$

(2)
For periodic boundary conditions (those considered here), the transform results in a diffusion equation

$$\frac{\partial \phi}{\partial t} = v \frac{\partial^2 \phi}{\partial x^2},$$

which may be solved with Fourier methods. Finally, the Cole-Hopf transform is inverted to recover the solution in the original coordinates.

We used a discretized version of the Cole-Hopf transform on a finely resolved lattice to generate simulation data. This approach works well for moderately sized lattices, but has instabilities for larger lattices when \( v \) becomes small\(^{22} \). Thus, for our flux limiter optimization procedure, below, we used a first-order, but very accurate scheme to generate our training and test data and validated the scheme, which is stable for any sized lattice, on simulations on smaller lattices by comparing with the discretized Cole-Hopf transform (See Appendix A). We call data generated in this way exact or high-resolution data. Note that this solution method can resolve shocks without the need for flux limiters and thus provides good data to test flux limiters in shock capturing integration schemes. For equations with no analytical solution, high-resolution data from finely resolved DNSs would need to be used.

### B. Flux Limiters

We use a semi-discrete scheme to integrate initial conditions from coarse-grained data selected at different resolutions from previously generated exact data. The integration scheme takes the general form

$$\frac{du_i}{dt} + \frac{1}{\Delta x_i} \left[ G \left( u_{i+\frac{1}{2}} \right) - G \left( u_{i-\frac{1}{2}} \right) \right] = 0 .$$

Here, \( i \) is a cell index, and \( G \left( u_{i-\frac{1}{2}} \right) \) and \( G \left( u_{i+\frac{1}{2}} \right) \) denote edge fluxes. The edge fluxes are non-linear interpolations between high- and low-resolution fluxes

$$G \left( u_{i+\frac{1}{2}} \right) = (1 - \phi(r_i)) f_{i+\frac{1}{2}}^{\text{low}} + \phi(r_i) f_{i+\frac{1}{2}}^{\text{high}},$$

$$G \left( u_{i-\frac{1}{2}} \right) = (1 - \phi(r_{i-1})) f_{i-\frac{1}{2}}^{\text{low}} + \phi(r_{i-1}) f_{i-\frac{1}{2}}^{\text{high}} .$$

Here, \( f_{i+\frac{1}{2}}^{\text{low}} \) indicates a low-resolution flux, typically first-order, which does not suffer from Gibbs phenomena, and \( f_{i+\frac{1}{2}}^{\text{high}} \) indicates a high-resolution flux to integrate smoother regions of a solution. The flux limiter, \( \phi(r_i) \), is a function of the flux ratio,

$$r_i = \frac{u_i - u_{i-1}}{u_{i+1} - u_i} .$$

A large number of flux limiters have been proposed and used. For instance, the van Leer flux limiter\(^{23} \) takes the form

$$\phi(r) = \frac{r + |r|}{1 + |r|} .$$

Note that this flux limiter and all of the flux limiters that we consider (see Fig. 1 and Tab. VII) here are (piecewise) continuous and zero for \( r \leq 0 \).
III. METHODS

A. A Shock Capturing Integration Method for Burgers’ Equation

For low- and high-resolution, we chose Lax–Friedrichs (LF)
\[ f^\text{low}_{i+\frac{1}{2}} = f^\text{LF}_{i+\frac{1}{2}} = \frac{1}{2} [F(u_i) + F(u_{i+1}) + \alpha \frac{\Delta x}{\Delta t} (u_{i+1} - u_i)]; \]
\[ \alpha = \max_u \frac{\partial F}{\partial u} \]  
(9)
and Lax–Wendroff (FW) fluxes
\[ f^\text{high}_{i+\frac{1}{2}} = f^\text{FW}_{i+\frac{1}{2}} = \frac{1}{2} [F(u_i) + F(u_{i+1}) + \frac{\Delta x}{\Delta t} \left( \frac{\partial F}{\partial u} (u_{i+\frac{1}{2}}) - F(u_i) \right)], \]
(10)
where \( F = \frac{\partial^2 u}{\partial x^2} - \nu \frac{\partial u}{\partial x} \) is the flux from Burgers’ equation.

B. Flux Limiter Discretization

We discretize the flux-limiter that we will optimize with machine learning methods, \( \phi(r) \), in piecewise linear segments, where the \( k \)th segment has the form,
\[ \phi_k(r) = \phi_0 + b_1 (r_2 - r_1) + b_2 (r_3 - r_2) + ... + b_k (r_r - r_{k-1}) + r_{k+1} + ... + r_K, \]
(14)
and \( r \in [r_k, r_{k+1}] \), \( k \in \{1, \ldots, K \} \), \( \phi_0 = 0 \), and \( b_i \) are slope coefficients. Note that for \( r \leq 0 \), \( \phi(r) = 0 \) and for \( r > r_K \), all terms in Eq. (14) are non-zero. Below, we use vector notation, \( b = [b_1, b_2, \ldots, b_k, b_{k+1}, \ldots, b_K]^T \) for slope coefficients. Eq. (14) can be rewritten as \( \phi_k(r) = b^T \Delta r_k \) with \( \Delta r_k \) defined as
\[ \Delta r_k = [r_r - r_1, r_3 - r_2, \ldots, r_{k-1} - r_k, 0, \ldots, 0]^T. \]
(15)

C. Learning an Optimal Discretized Flux Limiter

To optimize the discretized flux-limiter in Eq. (14), we define the mean squared error between \( N \) input-output pairs, \( \{o_i(\{u_i^*\}), g_i\} \):
\[ C = \frac{1}{2} \sum_{i=1}^N (o_i(\{u_i^*\}) - g_i)^2 \]  
(16)
as the cost. Here, \( g_i \) is the high-resolution flux velocity at the \( i \)-th grid position at time \( t_{n+1} \) and \( o_i \) is the shock-capturing method’s prediction of the fluid velocity at time \( t_{n+1} \) from data at the previous timestep. \( o_i \) is a functional of a subset of data points \( \{u_i^*\} = \{u_1^*, u_2^*, u_3^*, \ldots, u_N^*\} \) indicated relative to the \( i \)-th grid position at time step \( t_n \). Here, we used \( N_c = 6 \) data points at time \( t_n \) (see details in Appendix B) to predict a data point \( g_i \) at \( t_{n+1} \), i.e. \( \{u_i^*\} = \{u_{i-3}, u_{i-2}, u_{i-1}, u_i, u_{i+1}, u_{i+2}\} \). Thus, \( o_i(\{u_i^*\}) \) is the integration obtained with the flux-limiter method defined in Eqs. (4), (9), (10), (14) given a set of 6-points \( \{u_i^*\} \):
\[ o_i(\{u_i^*\}, t_{n+1}) = u_i(t_n) - \frac{\Delta t}{\Delta x} \Delta F(\{u_i^*\}, \{b_i\}, t_n). \]
(17)
Here, \( \Delta F(\{u_i^*\}, \{b_i\}, t_n) \), defined via Eqs. (12) and (13), is the difference of the two fluxes defined in Eq. (4). The minimum of the cost function, Eq. (16), can be computed exactly by finding the unique root, \( b \), of the equation \( \frac{\partial C}{\partial b} = 0 \), that is:
\[ \sum_{i=1}^N \left( u_i - g_i - \frac{\Delta t}{\Delta x} \Delta F^i \right) \left( - \frac{\Delta t}{\Delta x} \right) \Delta s_i \Delta F^i_{2,3} = 0. \]  
(18)
In Eq. (18), \( \Delta F^i = \Delta F(\{u_i^*\}, \{b_i\}, t_n) \) is defined via Eqs. (12) and (13). \( \Delta s_i = [\Delta r_i, \Delta r_{i+1}] \) is a \( K \times 2 \) matrix with \( \Delta r_i \) defined in Eq. (15). \( \Delta F^i_{2,3} = [\Delta F_2^i, \Delta F_3^i]^T \) with components \( \Delta F_2^i \) and \( \Delta F_3^i \) defined via Eq. (13).

Solving Eq. (18) reduces to solving a linear equation \( A \cdot b = C \) that yields \( b = A^{-1} \cdot C \). Here, \( A = \Delta r_F \cdot (\Delta r_F)^T \) and \( C = \frac{\Delta t}{\Delta x} \sum_{i=1}^N O_G^i \Delta r_F^i \), where \( \Delta r_F \) is a \( K \times N \) matrix with each column \( \Delta r_F^i \) a \( K \times 1 \) vector defined as \( \Delta r_F^i = (\Delta s_i)(\Delta F^i_{2,3}) \).
Finally, \( O_i^j = u_i - g_i - \frac{\Delta t}{\Delta x} \Delta F_i^j \). Note that \( \Delta F_i^j \) is defined via Eq. (13) and we recall that \( K \) is the size of the discretized flux limiter (i.e. the size of \( b \)). Hence, each matrix \( A \) (or \( C \)) is a function of \( N \) training data points.

We wish our estimates of each segment of the flux limiter, \( \phi_k(r) \), to have the same variance. Choosing this discretized space wisely is an important step. Our choice was to discretize the flux limiter such that each segment contained an equal number of training data points.

We used the above method with different coarse-grainings of the high-resolution dataset as training data in order to find optimal flux limiters, \( \phi_k(r) \), for a set of coarse-grainings.

### D. Hyperparameter Optimization of a Discretized Flux Limiter

The cost function defined in the previous section is intended to optimize a discretized flux-limiter for a given coarse-graining, \( CG \), number of bins, \( K \), and diffusion parameter, \( \mu \). We automated the generation of training and test data, the training of a discretized flux-limiter, and the testing of the learned limiter, to yield a function that produces the cost \( C \) as defined in Eq. (16) for a given \( CG \), \( K \), and \( \mu \). This approach leads to very good flux limiters for a particular set of parameters, however, the question remains as to whether we can find limiters that function well in a more general context.

To attack this issue, we extended our approach by optimizing over (hyper)parameters. By averaging the cost for a learned flux-limiter over all segments,

\[
\overline{C} = \frac{1}{K} \sum_{k=1}^{K} C_k / K.
\]  

(19)

In particular, we used an optimizer to produce a flux-limiter that minimizes \( \overline{C} \) within the region defined by \( \mu \in [0.005, 0.0248] \), \( CG \in [2, 10] \), and \( K \in [2, 38] \).

### IV. RESULTS

We generated 500 Burgers’ simulations with 500 random initial conditions that have in total 160M data points. We attained good convergence of training results even with just 80 simulations (i.e. using less than 30M data points). For a discretization into 20 segments, the solution to \( b \) was estimated with 1.5M data points per element and hence had a standard error of 0.0008. For this case, diagonalization was performed.
Here, the symmetric minmod limiter performs best, but still limiter performs better than all other limiters investigated. Here, \(CG\) is 2\(\times\) and \(r \in [0.0, 10.0]\). Note that the optimized limiter performs better than all other limiters investigated. Here, the symmetric minmod limiter performs best, but still reaches 10% greater error relative to the optimized limiter.

We plot optimized limiters, \(\phi_k(r)\) in Fig. 3(a). Also, included in the plot are van Leer and van Albada 2 limiters for reference. These two limiters are commonly-used limiters and bound the learned \(K = 20\) limiters. For this analysis, equal-variance bins (see Sec. III C) were computed individually for coarse grainings of 2\(\times\), 3\(\times\), 4\(\times\), and 8\(\times\) (transitioning from dotted to solid magenta lines in Fig. 3(a)). Note that the smallest coarse grained case (2\(\times\)) has the smallest first slope \(b_1\) in the first linear piece of \(\phi_k\), see Table 1. As we increase the coarse graining to a value larger than 2\(\times\), we see an increase in the slope of this first segment where \(b_1\) roughly tracks the value of the coarse-graining.

In Fig. 3(b), we show errors, \(E_{\text{int}}\), relative to the van Leer limiter, as a function of time, \(t\). Note that for 2\(\times\) through 8\(\times\) coarse grainings, the optimized limiters perform best, but for 10\(\times\) (not plotted in (a)), the discretized van Leer limiter performs better. Note that here, we compare with the continuous van Leer limiter.

| Coarse graining | First slope |
|-----------------|-------------|
| 2\(\times\)      | 2.33        |
| 3\(\times\)      | 3.11        |
| 4\(\times\)      | 4.02        |
| 8\(\times\)      | 7.28        |

Upon inspection of Fig. 3, the general form of the learned limiters is 1) an initial (small \(r\)) sharply sloped kink that gets sharper as coarse-graining increases, 2) a subsequent roughly linear region with low-amplitude fluctuations, and 3) a final (large \(r\)) kink that is less sharply sloped than the initial kink. In Fig. 4, we visualize this pattern by plotting the bin widths (Fig. 4(a)) and integrated slopes (Fig. 4(b)), \(\sum_{i=1}^{K} b_i\). The main patterns to point out here are that bin widths decrease for small \(r\) and increase for large \(r\), while the integrated slopes are virtually identical as a function of coarse-graining, with the exception that they scale roughly with the coarse-graining.

on the inverse of the square matrix \(A\) of size \(K \times K\) with maximal \(K\) being \(K = 20\) segments. Better standard errors were obtained with fewer segments, but at the cost of worse resolution of the limiter.

We validated our learning model on a subset of hold-out data which contained samples, \(\{u_i^c\}\), from about 20 simulations.

We compared our optimized flux limiter with a set of standard flux limiters (sFLs). We computed the spatially integrated relative error

\[
E_{\text{int}} = \sum_{i} \left( \frac{e_i^{\text{sFL}} - e_i^{\text{learned}}}{e_i^{\text{learned}}} \right),
\]

where the error at a given grid location, \(i\), for each flux limiter is

\[
e_i = \alpha_i - g_i.
\]

Note that this error, as defined, is positive when the learned limiter outperforms a given sFL. We investigated \(K = 2, 5,\) and 20 segment flux limiters. Optimized flux limiters for all values of \(K\) were better than the discretized van Leer limiter (i.e. the relative error was everywhere > 0 with an average improvement of about 10%).

In Fig. 2, we plot the integrated relative error of our optimized flux limiter with \(K = 20\) versus a set of eleven sFLs. Here, \(CG\) is 2\(\times\) and \(r \in [0.0, 10.0]\). Note that the optimized limiter performs better than all other limiters investigated. Here, the symmetric minmod limiter performs best, but still reaches 10% greater error relative to the optimized limiter.

We plot optimized limiters, \(\phi_k(r)\) in Fig. 3(a). Also, included in the plot are van Leer and van Albada 2 limiters for reference. These two limiters are commonly-used limiters and
Note also (in Tables II and III) that \( r_k \sim 1.0 \) when \( k \sim K/2 \), \( r_K \sim CG \), and \( b_1 \sim CG \).

Fig. 5 demonstrates the outcome of a flux limiter optimization in the hyperparameter subspace, \( \{CG, K, \mu \} \), defined in Sec. III D. For this optimization, we selected the Differential Evolution \(^{24}\) optimizer from mystic\(^ {25,26}\) with the coarse-graining and number of bins restricted to be integers. A population of twenty-two initial points \( \{CG, K, \mu \} \), chosen at random from within the set of valid solutions, was mutated at each iteration. Thus, each new generation contained twenty-two candidate solutions per iteration. The optimizer continued to generate new candidate solutions until the change in candidate solutions was less than \( 10^{-10} \) over 100 iterations. We used 50 simulations to build each training set, and 10 simulations for each test set, where each simulation was run for \( t = 800 \) steps. New training (and test) data is generated for each new combination of \( \{CG, K, \mu \} \) as part of the automated procedure.

In Fig. 5(a), we plot the learned flux limiters at select iterations as the hyperparameters converge. Note that as the learned limiters improve (with regard to the cost, in Fig. 5(c)), there is a convergence onto a positive linear slope at large \( r \), and a convergence of the mid-segment \( r_{K/2} \) of the limiter to roughly \( (r, \phi) = (1, 0.6) \). In comparing Fig. 5(b) and Fig. 5(c), it is apparent that the convergence of the cost is primarily driven by the convergence of the coarse-graining, \( CG \), and secondarily the diffusion parameter, \( \mu \), while the number of bins, \( K \), has little effect (also see Table IV). The lack of dependence of the cost on \( K \) makes sense, given that for all learned discretized flux limiters, the bulk of the line segments, \( r_i \), occur in the roughly linear region, \( r \in [0.5, 2.0] \), where the slope \( b_i \) exhibits only low-amplitude fluctuations. Essentially, as long as \( K \) is large enough to produce the initial and final kinked segments, any remaining extraneous segments appear to condense in the roughly linear region around \( r = 1 \).

It also appears that later in the convergence of the cost, where \( CG \) is fixed and \( \mu \) is still converging, that the value of \( \mu \) may primarily impact the shape of the limiter through \( \phi_K \) and the slope \( b_K \) for the last line segment. In early iterations, where \( CG \) and \( \mu \) are not nearly converged, we can see that at the mid-segment \( r_{K/2} = 1.0 \), but \( \phi_{K/2} \neq 0.6 \). As \( CG \) and then \( \mu \) converge, we find \( (r, \phi) = (1, 0.6) \).

The flux limiter that is learned in the hyperparameter optimization is given in Tables V and VI, and roughly follows the 'rules' established for \( r \) and \( b \) in Tables II and III. Specifically, learned limiters have an initial kinked region with a large slope, a final kinked region with a smaller slope, and a smaller roughly linear region around \( r_2 \sim 1.0 \) when \( k \sim K/2 \).
Additionally, we see that generally \( r_K \sim CG \) and \( b_1 \sim CG \).

V. DISCUSSION

Here, we presented a framework for the data-driven determination of optimal flux limiters for the coarse-grained Burgers’ equation. The framework consists of an internal optimizer computable with numerical linear algebraic methods, which can optionally be further optimized via a hyperparameter search over problem-specific parameters such as the number of bins in the limiter, the coarse-graining of the data, and the diffusion parameter (for Burgers’ equation this is the only physical parameter in the problem).

Similar, data-driven approaches to the integration of partial differential equations have been undertaken\(^{51}\). Here, by focusing on optimizing flux limiters, we attacked a central component of shock-capturing methods for integrating fluids. This allows our flux limiters to be essentially plug-and-play components for many existing numerical codes. We caution, however, that further work must still be done to understand how generally valuable our results are for fluid equations other than Burgers’. Information as to whether and how the regularities discovered in learned limiters change dependent on the target system may be useful for understanding the structure of shock-capturing codes more generally.

We chose to measure the quality of a discretized flux limiter by minimizing the mean-squared misfit for the limiter using 6 grid-point segments from the training data, averaged over all segments (as in Eq. (19)). We showed that under this condition, when searching for a generally optimal limiter across a range of hyperparameters, flux limiters should be designed with the following rules-of-thumb: they should have 1) a fixed point at \((r, \phi) = (0, 0)\), 2) a first segment with a slope of \(b_1 \sim CG\), 3) a second segment with a negative slope \(b_2 < 0\), followed by 4) a roughly linear region around \(r = 1\), \(\phi \in [0.5, 1.0]\), and \(K \sim K/2\), and 5) a larger final region where \(r_K \sim CG\). It is interesting to note that the \((r, \phi) = (1, 1)\) constraint, which is due to a requirement of second-order accuracy of the shock capturing scheme and Lipshitz continuity of \(\phi\)\(^{21}\), is not obeyed, yet these limiters, nonetheless, perform better than standard limiters.

A main advantage of our limiters seems to be the flexibility allowed by the multiple segments that make up their shape. As opposed to the bent or bowed shapes of standard limiters, multiple segment limiters allow a spring-like compression that, at least for Burgers’ equation, allows for an improvement in their performance.

In Fig. 6a, we show how our machine learned \(8\times\)-coarse-grained limiter fits with respect to the 2nd-order TVD region\(^{51}\). Full containment within the 2nd-order TVD region is a sufficient, but not necessary, condition to eliminate the possibility of Gibbs effects in 2nd-order shock capturing schemes (see discussion around Eqs. 2.15 and 2.16 in\(^{11}\)). To confirm numerically that our limiters do not exhibit a Gibbs effect, we plot the evolution of a sinusoidal initial condition for \(8\times\)-coarse-grained initial data. Note that the van Albada 2 limiter is partially outside of the 2nd-order TVD region in Fig. 6a, and exhibits a distortion of the sinusoidal form in Fig. 6b (particularly, away from the shock). Conversely, al-
Fixing $K$, and using a larger number of parallel cores, should be able to decrease the time to obtain results. As the limiter segments that condense around $r = 1$ are found to be very short, it may be interesting to attempt to learn if there is a relationship between the minimum number of bins required, the resolution of the limiter, and the resulting standard error. Another interesting study may be to discover the impact on the observed set of rules when $N_c \neq 6$, and if it has any impact on the shape or minimum number of bins.

Another interesting observation is that while we used a diffusion parameter of $\nu = 0.01$ in our simulations, we found a learned diffusion parameter of $\mu = 0.02471$ in our hyperparameter optimization (see Table IV). The difference is roughly a factor of 2, which counter-balances the difference in coarse-graining between our simulations and the learned limiter. Similar changes in effective diffusion are seen in analytical approaches to coarse-graining\textsuperscript{12}. It may be prudent to determine if the diffusion parameter can always be determined similarly for other choices of CG.

Here, we chose to develop and test our machine learning approach to discovering improved flux limiters using 2nd order shock-capturing methods. State-of-the-art methods such as the piecewise-parabolic methods\textsuperscript{17,26} (PPM, 4th order) or (weighted) essentially non-oscillatory\textsuperscript{29-31} (ENO, WENO, 8th order) methods may also benefit from our machine learning approach.

Further, as the generation of training data contains some randomness, an improvement would be to modify Eq. (16) to

\[ \phi(r) = \begin{cases} 
\theta_0 & \text{if } r < 0.248, \\
\rho_0 & \text{if } 0.248 \leq r < 1, \\
\rho_0 & \text{if } 1 \leq r, 
\end{cases} \]

where $\phi(r)$ is the shape or minimum number of bins. Fixing $K$, and using a larger number of parallel cores, should be able to decrease the time to obtain results. As the limiter segments that condense around $r = 1$ are found to be very short, it may be interesting to attempt to learn if there is a relationship between the minimum number of bins required, the resolution of the limiter, and the resulting standard error. Another interesting study may be to discover the impact on the observed set of rules when $N_c \neq 6$, and if it has any impact on the shape or minimum number of bins.

Another interesting observation is that while we used a diffusion parameter of $\nu = 0.01$ in our simulations, we found a learned diffusion parameter of $\mu = 0.02471$ in our hyperparameter optimization (see Table IV). The difference is roughly a factor of 2, which counter-balances the difference in coarse-graining between our simulations and the learned limiter. Similar changes in effective diffusion are seen in analytical approaches to coarse-graining\textsuperscript{12}. It may be prudent to determine if the diffusion parameter can always be determined similarly for other choices of CG.

Here, we chose to develop and test our machine learning approach to discovering improved flux limiters using 2nd order shock-capturing methods. State-of-the-art methods such as the piecewise-parabolic methods\textsuperscript{17,26} (PPM, 4th order) or (weighted) essentially non-oscillatory\textsuperscript{29-31} (ENO, WENO, 8th order) methods may also benefit from our machine learning approach.
train flux limiters that minimize the expected mean squared error between input-output pairs:

$$C = \frac{1}{2} \sum_{i=1}^{N} \mathbb{E}(o_i(\{u_i^c\}) - g_i)^2.$$  \hfill (22)

In our study the impact of the randomness in the training data is mitigated by using a large number of simulations. Thus, we may be able to use significantly fewer simulations if we instead train the limiter on the expected mean-squared error. Additionally, our approach could be used to train flux limiters for robustness by minimizing $\overline{C}$, defined as:

$$\overline{C} = \sum_{k=1}^{K} \max C_k / K ,$$  \hfill (23)

for a range of $(CG, \mu)$, with $C_k$ defined as in Eq. (22).

VI. CONCLUSIONS

We developed a theoretical framework that uses machine learning to train a continuous piecewise linear flux limiter by minimizing the mean-squared misfit to 6 grid-point segments of high-resolution data, averaged over all segments. We demonstrated our framework by producing a limiter that minimizes the misfit given a specific coarse-graining, number of bins, and diffusion parameter. We then compared the learned limiter to a set of 11 common flux limiters, ranking the flux limiters by their log-error relative to high-resolution data. We found that our $2 \times$, $K = 20$ bin machine learned limiter produces less misfit everywhere, with respect to all 11 other limiters. Similarly, our $3 \times$, $4 \times$, and $8 \times$ limiters, with $K = 20$ bins, generally outperformed all the other flux limiters. Our learned limiters all produced an initial kinked region with a large slope, a final kinked region with a smaller slope, and a smaller roughly linear region around $r_k \sim 1.0$ when $k \sim K / 2$. Additionally, we see that generally $r_K \sim CG$ and $b_1 \sim CG$. A main advantage of our limiters seems to be the flexibility allowed by the multiple line segments, as opposed to the bent or bowed shapes of standard limiters, enabling a spring-like compression that, at least for Burgers’ equation, yields an improvement in their performance. We also extended our framework to demonstrate we can machine learn piecewise-linear flux limiters that outperform standard limiters across a range of hyperparameter values of coarse-graining, number of discretized bins, and diffusion parameter. We found that learned limiters for the hyperparameter case also appear to generally adhere to the same rules as the piecewise linear limiters trained at fixed parameter values. Our study demonstrates a new approach to produce flux limiters that should be more broadly useful than standard limiters for general applications.

VII. ACKNOWLEDGEMENTS

We thank Robert Chiodi for discovering and correcting an error in our numerical code, which led us to correct a portion of our study. We thank Daniel Livescu for helpful discussions. Research presented in this article was supported...
Appendix A: High-resolution solutions

In this Appendix, we discuss the solution obtained with a high-resolution, first-order scheme that we use to generate training and testing data, as compared to the discretized Cole-Hopf approach. We note again that the Cole-Hopf approach may exhibit instability when \( \nu \) is small. The explicit form of the high-dimensional solution to Eq. (4) was obtained numerically as:

\[
u_i(t_{n+1}) = u_i(t_n) + \Delta t \left( D v + G v^2 \right) \tag{A1}
\]

with \( D = \frac{1}{(4\Delta t)}[1,0,-1], \) \( v = [u_{j-1}, u_j, u_{j+1}]^T, \) and \( G = \frac{\nu}{(\Delta x^2)}[1,-2,1]. \) Here \( v^2 \) indicates an element-by-element function giving the square of each element. Our high-resolution solution is very accurate when \( \Delta t \) and \( \Delta x \) stay small. This is confirmed in Fig. 7(a) where the high-resolution solutions are, at worst, within \( 10^{-6} \) of the discretized Cole-Hopf solution for coarse-grainings of \( 4 \times \), and yet better for coarse-grainings of \( 2 \times \) and below. Here, we use \( \nu = 0.01 \) and \( \Delta t = 5 \times 10^{-4}, \Delta x = 5 \times 10^{-3} \) which are the values we used to generate our high-dimensional training dataset.

Appendix B: Customized Lax-Friedrichs flux

We explored methods to avoid computing the maximum over all \( u_i \) when calculating the value of the partial derivative w.r.t. \( u \) in Eq. (9). If possible, this would allow us to reduce the training input points in our machine learning method.

We found that replacing \( \alpha = \max(\{\frac{Dv}{\Delta x}\})_u \) with the constant \( \alpha = 0.6 \) gave us improved solutions relative to the full flux limiter when compared to our high-resolution data.

In Fig. 7(b) we plot the relative error between the custom flux limiter and high resolution data compared to the relative error between the full maximum flux limiter and high resolution data for a high resolution simulation with \( \Delta x = 2.5 \times 10^{-3}, \Delta t = 2.5 \times 10^{-4}, v = 0.005. \) In the plot, we show the average solution error over twenty simulations with random initial conditions drawn from a uniform distribution from \( v_{\min} = -1 \) to \( v_{\max} = 1. \) Note that the modified method gives an error that is always less than the standard Lax-Friedrichs method. Therefore, we use this customized flux, \( f_{\text{low}} \), with \( N_C = 6 \) points, which appears in Eq. (17) in our machine learning model.

Appendix C: Learned flux limiters

Our discretized flux ratio bin edges, \( r_i \), and machine learned coefficients, \( b_i \), for flux limiters corresponding to \( 2 \times 3 \times 4 \times 8 \) of coarse graining (see Fig. 3(a)) are given in Tables II and III. Note that bin width \( (r_{i+1} - r_i) \) decreases for small \( r \) and increases for large \( r. \) Also note that \( r_k \sim 1.0 \) when \( k \sim K/2, r_k \sim CG, \) and \( b_1 \sim CG. \)

Similarly, solved \( r_i \) and \( b_i \) in the optimization of a discretized machine learned limiter in the region defined by \( CG \in [2, 10], K \in [2, 38], \) and \( \mu \in [0.005, 0.0248] \) (see Fig. 5(a)) are given in Tables V and VI. The corresponding values of \( CG, K, \) and \( \mu \) are given in Table IV.

Appendix D: Standard flux limiters

See Table VII for the flux limiters used for comparison in Figs. 1 and 2.

1. D. Besnard, F. H. Harlow, R. M. Rauenzahn, and C. Zemach, “Turbulence transport equations for variable-density turbulence and their relationship to two-field models,” Tech. Rep. (Los Alamos National Lab., NM (United States), 1992).
2. J. J. Schenk, F. D. Livescu, R. A. Gore, R. M. Rauenzahn, and J. R. Ristorcelli, “Application of a second-order close loop control to mixing processes involving multicomponent miscible fluids,” Journal of Turbulence, N49 (2011).
3. Y. Zhou, C. Zhong, Z. Li, and F. Li, “Preconditioned jacobian-free newton–krylov fully implicit high order weno schemes and flux limiter methods for two-phase flow models,” Nuclear Engineering and Technology, No 54, 49–60 (2022).
4. N. Vallee, X. Alvarez-Farré, A. Gorobets, J. Castro, A. Oliva, and F. X. Trias, “On the implementation of flux limiters in algebraic frameworks,” Computer Physics Communications, 271, 108230 (2022).
5. R. Abedian, “A third-order weighted essentially non-oscillatory-flux limiter scheme for two-dimensional incompressible navier-stokes equations,” Computational Methods for Differential Equations (2022).
6. R. Viinola and S. L. Brunton, “Enhancing computational fluid dynamics with machine learning,” Nature Computational Science, 2, 358–366 (2022).
7. J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin, “Accelerating Eulerian fluid simulation with convolutional networks,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70 (JMLR. org, 2017) pp. 3424–3433.
8. A. Mohan, D. Daniel, M. Chertkov, and D. Livescu, “Compressed convolutional lstm: An efficient deep learning framework to model high fidelity 3d turbulence,” arXiv preprint arXiv:1903.00033 (2019).
9. A. Sedgwick, M. Santha, and D. Livescu, “Leveraging bayesian analysis to improve accuracy of approximate models,” Journal of Computational Physics, 394, 280–297 (2019).
10. H. Chan, M. J. Chenkara, B. Narayanan, T. D. Loeffler, C. Benmore, S. K. Gray, and S. K. Sankaranarayanan, “Machine learning coarse grained models for water,” Nature communications, 10, 1–14 (2019).
11. J. M. Burgers, “A mathematical model illustrating the theory of turbulence,” in Advances in applied mechanics, Vol. 1 (Elsevier, 1948) pp. 171–199.
12. H. Eberhard, “The partial differential equation \( u_t + uu_x = \mu u_{xx} \),” (1942).
13. D. Cole, “On a quasi-linear parabolic equation occurring in aerodynamics,” Quarterly of applied mathematics 9, 225–236 (1951).
14. H. Wilbraham, “On a certain periodic function,” The Cambridge and Dublin Mathematical Journal 3, 198–201 (1848).
15. S. Godunov, “A difference scheme for numerical computation of discontinuous solutions of fluid dynamics,” Mat. Sb 47, 271–306 (1959).
TABLE II. Discretized space $r$ for $2 \times, 3 \times, 4 \times$, and $8 \times$ of coarse graining.

| $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $b_6$ | $b_7$ | $b_8$ | $b_9$ | $b_{10}$ | $b_{11}$ | $b_{12}$ | $b_{13}$ | $b_{14}$ | $b_{15}$ | $b_{16}$ | $b_{17}$ | $b_{18}$ | $b_{19}$ | $b_{20}$ | $b_{21}$ |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 1    | 0.33 | -0.18 | 0.17 | -0.21 | 0.15 | 0.00 | 0.16 | -0.02 | 0.52 | 0.25 | 0.28 | -0.15 | 0.03 | 0.08 | 0.14 | 0.04 | 0.20 | 0.08 | 0.29 | 0.09 |
| 2    | 0.71 | -0.63 | 0.89 | -0.08 | 0.10 | 0.24 | 0.08 | 0.02 | 0.32 | 0.10 | 0.24 | 0.02 | 0.06 | -0.07 | 0.17 | 0.04 | 0.21 | 0.06 | 0.28 | 0.14 |
| 3    | 0.02 | -1.81 | 0.67 | -0.84 | 0.19 | 0.25 | 0.12 | -0.05 | 0.37 | 0.02 | 0.12 | -0.01 | 0.11 | 0.16 | 0.01 | 0.10 | 0.14 | 0.04 | 0.28 | 0.16 |
| 4    | 0.72 | -2.36 | 0.75 | -0.33 | 0.22 | -0.04 | 0.18 | 0.16 | 0.16 | 0.04 | 0.20 | 0.12 | 0.06 | 0.04 | 0.15 | 0.10 | 0.07 | 0.02 | 0.16 | 0.30 |

TABLE III. Line segment slopes $b$ obtained for $K=20$ with $2 \times, 3 \times, 4 \times$, and $8 \times$ of coarse graining.

| $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $b_6$ | $b_7$ | $b_8$ | $b_9$ | $b_{10}$ | $b_{11}$ | $b_{12}$ | $b_{13}$ | $b_{14}$ | $b_{15}$ | $b_{16}$ | $b_{17}$ | $b_{18}$ | $b_{19}$ | $b_{20}$ |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 1    | 0.4 | 20 | 0.02 | 0.348 | 0.000029491 | 2    | 3 | 33 | 0.02 | 0.366 | 0.00000547 | 3    | 34 | 0.02 | 0.367 | 0.0000393 | 6    | 43 | 0.02 | 0.479 | 0.00000387 |
| 1    | 20 | 0.02 | 0.247 | 0.000034 |
| 1    | 40 | 0.02 | 0.2474 | 0.00000352 |
| 1    | 100 | 0.02 | 0.2477 | 0.00000341 |

TABLE IV. Coarse graining, $CG$, number of bins, $K$, diffusion parameter, $\mu$, and associated cost, $\mathcal{C}$, at selected iterations in the optimization of a discretized machine learned limiter in the region defined by $CG \in [2, 10]$, $K \in [2, 38]$, and $\mu \in [0.005, 0.0248]$.

| $CG$ | $K$ | $\mu$ | $\mathcal{C}$ |
|-------|-----|-------|-------|
| 0    | 4x  | 20    | 0.02431 |
| 3    | 2x  | 33    | 0.02366 |
| 8    | 2x  | 34    | 0.02476 |
| 20   | 2x  | 38    | 0.02479 |
| 55   | 2x  | 37    | 0.02473 |
| 80   | 2x  | 35    | 0.02474 |
| 110  | 2x  | 36    | 0.02471 |

16B. Van Leer, “Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov’s method,” Journal of computational physics 32, 101–136 (1979).
17P. Colella and P. R. Woodward, “The piecewise parabolic method (ppm) for gas-dynamical simulations,” Journal of computational physics 54, 174–201 (1984).
18A. Harten, “High resolution schemes for hyperbolic conservation laws,” Journal of computer physics 135, 260–278 (1997).
19C.-W. Shu and S. Osher, “Efficient implementation of essentially non-oscillatory shock capturing schemes, 2,” (1988).
20D. Zhang, C. Jiang, D. Liang, and L. Cheng, “A review on tdv schemes and a refined flux-limiter for steady-state calculations,” Journal of Computational Physics 302, 114–154 (2015).
21P. K. Sweby, “High resolution schemes using flux limiters for hyperbolic conservation laws,” SIAM journal on numerical analysis 21, 995–1011 (1984).
22M. Maritz and S. Schoombie, “Exact analysis of nonlinear instability in a discrete burgers’ equation,” Journal of Computational Physics 97, 73–90 (1991).
23B. Van Leer, “Towards the ultimate conservative difference scheme. ii. monotonicity and conservation combined in a second-order scheme,” Journal of computational physics 14, 361–370 (1974).
24R. M. Storn and K. V. Price, “Differential evolution — a simple and efficient heuristic for global optimization over continuous spaces,” J. Global Optim. 11, 341–359 (1997).
25M. McKerns, P. Hung, and M. Aivazis, “mystic: highly-constrained non-convex optimization and u,” (2009-), https://uqfoundation.github.io/project/mystic.
26M. McKerns, L. Strand, T. J. Sullivan, A. Fang, and M. Aivazis, “Building a framework for predictive equations,” in Proceedings of the 10th Python in Science Conference, 2011 (2011) pp. 67–78, http://arxiv.org/pdf/1202.1056.
27Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner, “Learning data-driven discretizations for partial differential equations,” Proceedings of the National Academy of Sciences 116, 15344–15349 (2019).
28B. Tripathi, D. Espíndola, and G. Pinton, “Piecewise parabolic method for simulating one-dimensional shear shock wave propagation in tissue-mimicking phantoms,” Shock Waves 27, 879–888 (2017).
29A. Harten and S. Osher, “Uniformly high-order accurate nonoscillatory schemes. i,” in Upwind and High-Resolution Schemes (Springer, 1997) pp. 187–217.
30X.-D. Liu, S. Osher, and T. Chan, “Weighted essentially non-oscillatory schemes,” Journal of computational physics 115, 200–212 (1994).
31Y. Ha, C. H. Kim, Y. J. Lee, and J. Yoon, “Mapped weno schemes based on a new smoothness indicator for hamilton–jacobi equations,” Journal of Mathematical Analysis and Applications 394, 670–682 (2012).
32P. L. Roe, “Characteristic-based schemes for the euler equations,” Annual review of fluid mechanics 18, 337–365 (1986).
33B. Van Leer, “Towards the ultimate conservative difference scheme iii. upstream-centered finite-difference schemes for ideal compressible flow,” Journal of Computational Physics 23, 263–275 (1977).
34P. Gaskell and A. Lau, “Curvature-compensated convective transport: Smart, a new boundedness-preserving transport algorithm,” International Journal for numerical methods in fluids 8, 617–641 (1988).
35B. Koren, A robust upwind discretization method for advection, diffusion and source terms (Centrum voor Wiskunde en Informatica Amsterdam, 1993).
36N. Waterson and H. Deconinck, “A unified approach to the design and application of bounded higher-order convection schemes,” Numerical methods in laminar and turbulent flow, 9, 203–214 (1995).
37F.-S. Lien and M. Leschziner, “Upstream monotonic interpolation for scalar transport with application to complex turbulent flows,” International Journal for Numerical Methods in Fluids 19, 527–548 (1994).
38G. D. Van Albada, B. Van Leer, and W. Roberts, “A comparative study of computational methods in cosmic gas dynamics,” in Upwind and high-resolution schemes (Springer, 1997) pp. 95–103.
39M. Kermani, A. Gerber, and J. Stockie, “Thermodynamically based moisture prediction using roe’s scheme,” in 4th Conference of Iranian Aerospace Society, Amır Kabir University of Technology, Tehran, Iran, Vol. 3 (Citeseer, 2003).
Machine Learning Changes the Rules for Flux Limiters

| Table V. Solved discretized bin edge locations, $r_i$, in the optimization of a discretized machine learned limiter in the region defined by $CG \in [2, 10]$, $K \in [2, 38]$, and $\mu \in [0.005, 0.0248]$. The corresponding values of $CG$, $K$, and $\mu$ are found in Table IV.

| Table VI. Solved line segment slopes, $b_i$, in the optimization of a discretized machine learned limiter in the region defined by $CG \in [2, 10]$, $K \in [2, 38]$, and $\mu \in [0.005, 0.0248]$. The corresponding values of $CG$, $K$, and $\mu$ are found in Table IV.

| Table VII. Mathematical expressions for the standard flux limiters used for comparison in Figs. 1 and 2.

| $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ | $r_6$ | $r_7$ | $r_8$ | $r_9$ | $r_{10}$ | $r_{11}$ | $r_{12}$ | $r_{13}$ | $r_{14}$ | $r_{15}$ | $r_{16}$ | $r_{17}$ | $r_{18}$ | $r_{19}$ | $r_{20}$ | $r_{21}$ | $r_{22}$ |
|------|------|------|------|------|------|------|------|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 1.10 | 0.01 | 0.35 | 0.52 | 0.63 | 0.71 | 0.76 | 0.80 | 0.83 | 0.86 | 0.88 | 0.90 | 0.91 | 0.93 | 0.94 | 0.96 | 0.97 | 0.98 | 0.99 | 1.00 | 1.01 | 1.02 | 1.03 |

| $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $b_6$ | $b_7$ | $b_8$ | $b_9$ | $b_{10}$ | $b_{11}$ | $b_{12}$ | $b_{13}$ | $b_{14}$ | $b_{15}$ | $b_{16}$ | $b_{17}$ | $b_{18}$ | $b_{19}$ | $b_{20}$ | $b_{21}$ | $b_{22}$ |
|------|------|------|------|------|------|------|------|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 2.11 | 1.58 | 1.30 | 0.97 | 0.90 | 0.79 | 0.73 | 0.68 | 0.66 | 0.64 | 0.63 | 0.62 | 0.61 | 0.60 | 0.59 | 0.58 | 0.57 | 0.56 | 0.55 | 0.54 | 0.53 | 0.52 |

- **superbee**\(\text{\textsuperscript{32}}\): \(\phi_{\text{sb}}(r) = \max(0, \min(2r, 1), \min(r, 2))\)
- **monotonized central**\(\text{\textsuperscript{32}}\): \(\phi_{\text{mc}}(r) = \max(0, \min(2r, 0.5(1 + r), 2))\)
- **smart**\(\text{\textsuperscript{32}}\): \(\phi_{\text{sm}}(r) = \max(0, \min(2r, 1/4 + 3r/4, 4))\)
- **Koren**\(\text{\textsuperscript{32}}\): \(\phi_{\text{km}}(r) = \max(0, \min(2r, \min((1/3 + 2r/3), 2)))\)
- **van Leer**\(\text{\textsuperscript{32}}\): \(\phi_{\text{vL}}(r) = \frac{r + |r|}{1 + |r|}\)
- **HCUS**\(\text{\textsuperscript{36}}\): \(\phi_{\text{HCUS}}(r) = \frac{1}{1.5(r + |r|)}\)
- **ospre**\(\text{\textsuperscript{36}}\): \(\phi_{\text{ospre}}(r) = \frac{1}{1.5(r + |r|)}\)
- **UMIST**\(\text{\textsuperscript{37}}\): \(\phi_{\text{um}}(r) = \max(0, \min(2r, 1/4 + 3r/4, 3/4 + r/4, 2))\)
- **van Albada**\(\text{\textsuperscript{18}}\): \(\phi_{\text{va1}}(r) = \frac{r^2 + r}{r^2 + 1}\)
- **van Albada**\(\text{\textsuperscript{19}}\): \(\phi_{\text{va2}}(r) = \frac{r^2 + r}{r^2 + 1}\)
- **symmetric minmod**\(\text{\textsuperscript{32}}\): \(\phi_{\text{mm}}(r) = \max(0, \min(1, r))\)