Appendix

A. Illustrative Examples

Example 1 (Obtaining the Prolongation of an Infinitesimal Generator)**.** *We will work through an example of obtaining the prolongation of an infinitesimal generator of the heat equation: Consider $X \times U = \mathbb{R}^2 \times \mathbb{R}$ and the following infinitesimal generator, a symmetry of the heat equation:*

$$\begin{aligned}\mathbf{v} &= \xi_1(x,t,u)\partial_x + \xi_2(x,t,u)\partial_t + \phi(x,t,u)\partial_u \\ &= 2\nu t\partial_x - xu\partial_u\end{aligned}$$

where x, t denote the independent variables, u is the dependent variable and ν is a positive constant. By the prolongation formula, Eq. (4), the first prolongation in t is given by:

$$\begin{aligned}\phi^t &= D_t(\phi - \xi_1 u_x - \xi_2 u_t) + \xi_1 u_{xt} + \xi_2 u_{tt} \\ &= D_t(-xu - 2\nu t u_x) + 2\nu t u_{xt} \\ &= -xu_t - 2\nu u_x\end{aligned}$$

Example 2 (A Symmetry of the Heat Equation)**.** *As another illustrative example, we can consider the heat Eq. (2) and will show that the following vector field generates a symmetry group for this PDE: $\mathbf{v} = 2\nu t\partial_x - xu\partial_u$. We need to find the first prolongation $\phi^{(t)}$ and the second prolongation $\phi^{(xx)}$, where $\phi = -xu$. Using the prolongation formula given in Eq. (4), we get:*

$$\phi^{(t)} = -xu_t - 2\nu u_x \qquad \text{and} \qquad \phi^{(xx)} = -2u_x - xu_{xx}$$

Now:

$$\begin{aligned}\mathrm{pr}^{(2)}\mathbf{v}[\Delta] &= \phi^{(t)} - \nu\phi^{(xx)} \\ &= xu_t + 2\nu u_x - \nu\big(2\nu u_x - x\nu u_{xx}\big) \\ &= x\big(u_t - \nu u_{xx}\big)\end{aligned}$$

Clearly $\mathrm{pr}^{(2)}\mathbf{v}[\Delta] = 0$ when $\Delta = 0$, hence $\mathbf{v}$ is a symmetry of the heat equation.

B. Theory and Background

One Parameter Subgroups. The Lie point symmetry $\mathcal{G}$ of Δ can be multi-dimensional and complex. When the n-dimensional group $\mathcal{G}$ is *simply connected*, it is often represented in terms of a series of n *one-parameter transformations*, $g = g_1(\epsilon_1)g_2(\epsilon_2)\dots g_n(\epsilon_n)$, where $g_i : \mathbb{R} \to \mathcal{G}$, $i = 1, \dots, n$ and such that $g_i(\epsilon)g_i(\delta) = g_i(\epsilon + \delta)$. The ϵ's are the real parameters of the transformation, and each g_i is a continuous group homomorphism (*i.e.*, a smooth, group-structured map) from this parameter to the symmetry group.

Solution set of a PDE. The graph of all prolonged solutions is the set $\mathcal{S}_\Delta \subset X \times U^{(n)}$, and is defined as:

$$\mathcal{S}_\Delta = \{(\mathbf{x}, \mathbf{u}^{(n)}) : \Delta(\mathbf{x}, \mathbf{u}^{(n)}) = 0\} \tag{11}$$

In this new notation, we can say that $\mathbf{u}(\mathbf{x})$ is a solution of the PDE if:

$$\Gamma_u^{(n)} = \{(\mathbf{x}, \mathrm{pr}^{(n)}\mathbf{u}(\mathbf{x}))\} \subset \mathcal{S}_\Delta$$

where $\mathrm{pr}^{(n)}\mathbf{u}(\mathbf{x}) : X \to U^n$, is a vector-valued function whose entries represent derivatives of $\mathbf{u}$ wrt $\mathbf{x}$ up to order n.

Prolonations. The coefficients for the partial derivatives are given below. See (Olver, 1986) for a derivation.

$$\phi_\alpha^{(J)} = D_J Q_\alpha + \sum_{i=1}^{p} \xi_i \frac{\partial \mathbf{u}_J^\alpha}{\partial x^i} \quad \text{where} \quad Q_\alpha = \phi_\alpha - \sum_{i=1}^{p} \xi_i \frac{\partial \mathbf{u}^\alpha}{\partial x^i} \tag{12}$$

C. Data Generation

To generate data, we use $\Omega = [0, L] = [0, 2\pi]$, discretized uniformly into 256 points and assume periodic spatial boundaries. We also use $[0, T] = [0, 16]$, discretized into 100 points. The viscosity coefficient is set to $\nu = 0.01$. Similar to (Brandstetter et al., 2022a) and (Bar-Sinai et al., 2019), we represent the initial condition functions by truncated Fourier series with coefficients A_k, l_k, ϕ_k sampled randomly, and $K = 10$: $u(t = 0, x) = f(x) = \sum_{i=1}^{K} A_k \sin(2\pi l_k x/L + \phi_k)$. These functions are sampled at $N_s = 200$ fixed points, which are used as input to e_{θ_1} in Eq. (8). We also sample a total of $N_l = 300$ points (including the 200 points sampled at $t = 0$), used to impose the data-fit loss, $\mathcal{L}_{\text{data-fit}}$.

D. Implementation

We model the two networks, g_{θ_1} and e_{θ_2} in Eq. (8) with MLPs consisting of 7 hidden layers of width 100. This choice was based on the previous research using PINN and DeepONets for solving Burgers' equation (Wang et al., 2021b). We used elu activation as differentiable activations are required for the PDE loss. The output of the embedding vectors from both networks is 100 dimensional. We note that while we acknowledge recent architectural improvements to PINNs (Krishnapriyan et al., 2021), our goal is to showcase the effectiveness of using symmetries, which is why we deploy MLPs for both networks.

For both the Heat equation and Burgers' equation experiments, we perform hyper-parameter tuning on the coefficients of the loss terms from the set $[0.1, \dots, 1, \dots, 10, \dots, 100, \dots 200]$. This is done separately for the baseline model and the model trained with symmetry loss as we varied the number of samples, N_r.

Algorithm 1 PINN with Lie Point Symmetry

inputs:
Δ: the PDE of order n,
$(\mathbf{v}_k)_{1:K}$: infinitesimal generators of symmetries of Δ
$\mathcal{D} = \left\{\left(\mathbf{x}_{1:N_r}, (\mathbf{x}, \mathbf{u})_{1:N_l}, \mathbf{u}_{1:N_s}\right)\right\}_{1:N_f}$: dataset for N_f different initial conditions
init: initialize parameters θ of network u_θ
for iteration **do**
Sample from $\mathcal{D}$
calculate $\mathcal{L}_{\text{sym}}$:
calculate $\mathbf{u}^{(n)}$ using auto diff for $\mathbf{x}_{1:N_r}$
calculate $\text{coef}(\text{pr}^{(n)}\mathbf{v}_k)$, using $\mathbf{x}_{1:N_r}$, $(\mathbf{u}^{(n)})_{1:N_r}$, auto diff and Eq. (4)
calculate $\mathcal{L}_{\text{sym}}$ using $\text{coef}(\text{pr}^{(n)}\mathbf{v}_k)$ and Δ and equation Eq. (9)
calculate $\mathcal{L}_{\text{PDE}}$:
using $\Delta, (\mathbf{u}^{(n)})_{1:N_r}$ and equation Eq. (6):
calculate $\mathcal{L}_{\text{data-fit}}$:
calculate $\hat{\mathbf{u}} = \mathbf{u}_\theta(\mathbf{x}, \mathbf{u}_{1:N_s})$ for $\mathbf{x}_{1:N_l}$
calculate $\mathcal{L}_{\text{data-fit}}$ using $\mathbf{u}_{1:N_l}$, $\hat{\mathbf{u}}_{1:N_l}$ and equation Eq. (7)
$\mathcal{L} = \alpha\mathcal{L}_{\text{PDE}} + \beta\mathcal{L}_{\text{data-fit}} + \gamma\mathcal{L}_{\text{sym}}$
$\theta \leftarrow \theta - \nabla_\theta\mathcal{L}$
end for
return u_θ

We also note that for Burgers' equation, cosine similarity for $\mathcal{L}_{\text{sym}}$ works better than the dot product. The results reported in Section 4 use cosine-similarity.

Algorithm 1 outlines the algorithm. Note that in Algorithm 1, we use the notation u_θ to denote the operator $\mathcal{O}_\theta$ and use N_s for the fixed sampled points from the initial condition function. Additionally, we will use $(\mathbf{x}l, \mathbf{u}^l)_{1:N_l}$ to include both N_0 and N_b. We will make the data and the code available on GitHub.

E. Results for Burgers' Equation

The second PDE we analyze is Burgers' Eq. (13), which combines diffusion (with thermal diffusivity ν) and nonlinear advection (wave motion). The nonlinearity of this equation makes it more complex, resulting in shock formation.

$$u_t = \nu u_{xx} - uu_x \tag{13}$$

Symmetries. Burgers' equation in the form described in Eq. (13) has a symmetry group spanned by the following 5-dimensional vector space.

$$\begin{aligned} \mathbf{v}_1 &= \partial_x & \mathbf{v}_4 &= x\partial_x + 2t\partial t - u\partial_u \\ \mathbf{v}_2 &= \partial_t & \mathbf{v}_5 &= tx\partial_x + t^2\partial_t - (x - tu)\partial_u \\ \mathbf{v}_3 &= t\partial_x + \partial_u \end{aligned} \tag{14}$$

However, only the last generator $\mathbf{v}_5$ results in a useful training signal. The first three generators give $\mathcal{L}_{\text{sym}} = 0$ and $\mathbf{v}_4$ gives $\mathcal{L}_{\text{sym}} = c\Delta = c\mathcal{L}_{\text{PDE}}$, for a constant c. As with the heat equation experiment, we can eliminate the PDE loss and only use symmetry and supervised loss for training.

Data. The data for Burgers' equation is obtained using the Fourier Spectral method with periodic spatial boundaries. Initial conditions are obtained similarly to the heat equation experiment, described in Section 4. We use $\nu = 0.1$ as the diffusion coefficient. The domain is $[0, L] = [0, 2\pi]$ and $[0, T] = [0, 2.475]$ discretized uniformly into 256 and 100 points respectively.

Training and Experiments. For Burgers' equation, we train the model on datasets of $N_f = 500$ initial conditions and $N_r = 5000, 25000$ and 100000 samples. We found that for $\mathcal{L}_{\text{sym}}$, cosine similarity works better in this case. The models' architectures are similar to those used for the heat equation described in Appendix D.

Results. Table 2 shows the average mean-squared errors on the test dataset for the two models as N_r increases. We can see that, even with one symmetry group useful for training, the model trained with $\mathcal{L}_{\text{sym}}$ performs better. The predictions on a single instance of the test dataset can also be seen in Fig. 3. Again, we see that the symmetry loss especially improves the model's performance for larger values of t, especially in low-data regime. We also note that the high standard deviations in Table 2 are because, compared to the heat equation, the behaviour of the solution (specifically shock formation) varies a lot based on the initial conditions.

Table 2. The average test set mean-squared error for Burgers' equation.

Number of Points (N_r)	No Symmetry	Symmetry
5000	0.041 ± 0.042	$\mathbf{0.034 \pm 0.039}$
25000	0.030 ± 0.038	$\mathbf{0.017 \pm 0.020}$
100000	0.018 ± 0.022	$\mathbf{0.013 \pm 0.020}$

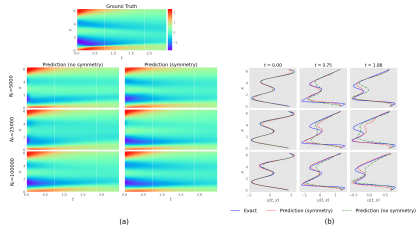

Figure 3. The effect of training the PDE solver for the Burgers' equation with and without the symmetry loss for one of the PDEs in the test dataset. (a) shows the ground truth solution and the predictions of the two models as the number of samples inside the domain increases from 5000 to 25000 and 100000.(b) shows the corresponding predictions and the ground truth solution at different time slices.