Machine Learning in the Study of Differential Equations Related to Nuclear Engineering

M. Zafiris, K. Jegdic
University of Houston-Downtown, Houston, TX 77002

Keywords: Neural Network, Partial Differential Equations, Nuclear Engineering

I. Abstract

Most of differential equations that describe the real-world phenomena cannot be solved exactly and one must utilize an approximation method to obtain a solution, such as finite difference, finite volume, finite element numerical method, etc. The goal of this project is to utilize machine/deep learning methods in study of conservation laws that model multi-phase flows. Our project's emphasis is on relatively recent methods for solving differential equations that utilize machine learning and, specifically, neural networks, motivated by the universal approximation theorem.
Our approach consists of proposing a trial solution consisting of two parts the first part, satisfying the initial/boundary conditions, with no adjustable parameters, and the second part consists of a neural network that is trained to satisfy the differential equation by minimizing the squared error loss. We have found that the machine learning approximation methods yield more accurate and faster results than typical approximation methods.

II. Background

Differential equations model a variety of phenomena in sciences and engineering. The main focus of this project is on time-dependent partial differential equations known as conservation laws that are used to model multiphase flows. In particular, these equations are of interest to engineers who design nuclear reactor pipes to ensure the optimal flow of gasses and liquids.
We consider the differential equations stated in [3]. They are a simplified model for the flow of gasses and liquids in vertical pipes and are expressed as time-dependent one-dimensional conservation of mass laws in terms of volume fractions for the liquid and gas (αl and αg). Since αl + αg = 1, we state only the equation for αl and for simplicity of notation, we drop the subscript l:

$$\partial_t \alpha + h'(\alpha)\partial_x \alpha = 0$$

where α (x, t) is the unknown function denoting the fractional volume of liquid, t and x denote the temporal and spatial variables and h is the flux given by:

$$h(\alpha) = -\frac{\alpha \cdot (1-\alpha)^2}{\alpha \cdot (1-\alpha) \cdot I_g \cdot \mu_g + (1-\alpha)^2 \cdot I_l \cdot \mu_l + II \cdot \mu_l} \cdot \Delta\gamma$$

as well as the initial and boundary conditions,

$$\alpha(x,0) = \begin{cases} \alpha_{left}, & x = 0, \\ f(x), & 0 < x < 1, \\ \alpha_{right}, & x = 1, \end{cases}$$

Typically, these PDE solutions are approximated using numerical methods such as finite differences, finite volumes or finite elements [6]. However, we utilize the above stated background to generate solutions using neural network method stated in [5].

III. Solution Approximation Method

Method discussed in [5] results in an approximating solution that is given in a differentiable, closed analytic form. The main idea is to consider a trial solution written as a sum of two parts. The first part is constructed to satisfy the initial/boundary conditions, while the second part is given in terms of a neural network whose parameters are learned by minimizing the squared error loss function that imposes the governing differential equation. The trial solution is defined by

$$\alpha_{trial}(x,t) = A(x,t) + xt(1-x)N(x,t,\vec{p})$$

where the first term A(x, t) will be defined to incorporate initial and boundary conditions and the second term involves neural network with set of parameters denoted by p that will incorporate the differential equation. The appropriate choice for A(x, t) is

$$A(x,t) = (1-x)\cdot\alpha_{left} + x\cdot\alpha_{right} + (1-t)\cdot[f(x) - (1-x)f(0) - xf(1)]$$

The boundary conditions state what the fractional volume of liquid is at the entry of the pipe (when x = 0) and at the exit of the pipe (when x = 1), while the initial condition (when t = 0) specifies the initial distribution of liquid inside the pipe. Here, α-left and α-right are constants given in advance and f(x) will be a function that is given in advance. In our case, α-left = 1, α-right = 0, and f(x) which is compatible with alpha-l and alpha-r is a piecewise constant function as in [3].

Next, we need to define the neural network. We have two input variables x and t, one hidden layer with 20 nodes, and output N. Each node in the hidden layer is of the form,

$$z_i = w_{1i}x + w_{2i}t + u_i$$

We utilize the sigmoid activation function. Therefore, our neural network is defined as follows,

$$\sigma(z) = \frac{1}{1+e^{-z}}, \quad N(x,t) = \sum_{i=1}^{20} v_i\sigma(z_i)$$

In defining the minimization problem, we consider 400 points from the domain [0, 1] x [0, 1], obtained by considering 20 equidistant points in the interval [0, 1] in each variable t and x. We want to minimize the error by finding the optimal parameter set p satisfying

$$\min_{\vec{p}} \sum_{m,n} \left(\partial_t \alpha_{trial}(x_m,t^n) + h'(\alpha_{trial}(x_m,t^n))\cdot\partial_x\alpha_{trial}(x_m,t^n)\right)^2.$$

IV. Results

Our results yield accurate results, consistent to traditional methods, such as finite differences at a faster rate. This allowed for us to generate a surface, without computationally burdening our system. We are confident machine learning methods are also successful in approximating equations describing the multiphase flows.

Following the accuracy results utilizing the methods in [5], we wish to now implement a physics informed deep neural network approximation method discussed in [8]. This would allow for a more detailed and fine-tuned surface generation.

V. References

[1] Baydin, A. G., Pearlmutter, B. A., Radu,l A. A., and Siskind, J. M. 2018. Automatic differentiation in machine learning: a survey. Journal of Machine Learning Research 18, 1-43.

[2] Cybenko, G. 1989. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems, 2(4). 303-314.

[3] Dirdal, I. K. 2015. A Mathematical Model for Flow of Gas-Liquid Mixture in a Vertical Pipe. Master's Thesis. Faculty of Science and Technology, University of Stavanger.

[4] Fletcher, R. 1987. Practical Methods of Optimization. 2nd edition, John Wiley.

[5] Lagaris, I. E., Likas, A., and Fotiadis, D. I. 1998 Artificial neural networks for solving ordinary and partial differential equations. Mathematics, Physics, Medicine, Computer Science. Published in IEEE Trans. Neural Networks.

[6] LeVeque, R. J. 1992. Numerical Methods for Conservation Laws. Birkhauser Verlag.

[7] Liu, D. C., Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. Mathematical Programing 45, 503-528

[8] Raissi, M., Perdikaris, P., and Karniadakis, G. E. 2017. Physics informed deep learning (part I): data-driven solutions of nonlinear partial differential equations. arXiv: 1711.10561.