Imperial College London

Department of Earth Science and Engineering

MSc in Applied Computational Science and Engineering

Independent Research Project
Project Plan

Solving Partial Differential Equations with AI Libraries

by

IJEH, CHUKWUME

Email: chukwume.ijeh21@ic.ac.uk
GitHub username: acse-cgi21
Repository: https://github.com/ese-msc-2021/irp- cgi21

Supervisors:

Christopher Pain

Claire Heaney
Boyang Chen

July 2022

1 Introduction

Partial Differential Equations (PDE) play a crucial role in understanding, modelling and simulating physical phenomena across a diverse range of disciplines [14]. Some of these disciplines range from electromagnetism, heat conduction and diffusion, to quantum mechanics, wave propagation, general relativity and fluid dynamics [15].

Computational fluid dynamics (CFD) is a science concerned with predicting fluid-flow phenomena with the help of digital computers and conservation laws (such as mass, momentum, and energy) governing fluid motion [6]. One of such fluid-flow phenomena is that concerned with advection-diffusion. The advection-diffusion equation (ADE) is a PDE which is of great relevance in the society today. All around us, the importance of transport of particles is felt and seen. ADE describes this type of transport in many important processes in hydrogeology, mechanics, geology, and biology [16]. The importance of ADE can immediately be seen in the use of a variation of the equation in modelling infection risks of the COVID-19 virus in an enclosed space [10].

PDEs, including ADE, have previously been solved using varied traditional numerical methods [11], such as finite element methods (FEM) like time-space flux-corrected transport (FCT), finite difference methods (FDM) and finite volume methods (FVM) [3, 1, 2, 12, 16]. Some of these methods have been found very effective but however, they accrue high computational costs while also being extremely complex to code.

With the advancement in hardware technologies and computing power of artificial intelligence (AI) software, there has recently being a need for a paradigm shift in how PDEs are solved. The benefits of combining such AI software and traditional numerical methods has become a new focus because of speed and simplicity. For more context, Cerebras has recently produced a new 'AI' computer with approximately 1million cores capable of performing exascale computing. A single core of this AI computer delivers the wall-clock compute performance of many tens to hundreds of graphics processing units (GPU). If the combination of the new AI computers and AI software with traditional numerical methods can be harnessed, we can foresee a revolution in diverse disciplines, including computational physics.

2 Literature Review

Over the last two decades, AI has made an explosive impact in almost every industry, skyrocketing growth and aiding innovation of things once thought impossible. The extent of AI and machine learning (ML) is so broad range that over the last few years, researchers have been seeking possibilities of harnessing AI technologies with traditional methods of discretising PDEs. If there is a breakthrough in this area of using ML to discretize PDEs, there will be a massive increase in the number of developers capable of developing current CFD/nuclear code due to code simplicity and massive decrease in computational time. This breakthrough will further increase their parallel scalability.

Below, we look at some of the research already done in this area.

2.1 Deep Neural Network (DNN) vs standard FDM-based Numerical Solvers

[8] compared the use of a deep learning-based solver with that of some standard FDM-based numerical solvers by solving the 2D unsteady advection-diffusion equation (shown in equation 1). The following FDM-based solvers were considered: forward time central space method, upwind numerical method and fully implicit method. In 1, ϕ, D_x, D_y, v_x and v_y represent the fluid concentration, the coefficients of diffusion along the x and y-axes, the horizontal and vertical velocity, respectively.

$$\frac{\partial \phi}{\partial t} + v_x \frac{\partial \phi}{\partial x} + v_y \frac{\partial \phi}{\partial y} - D_x \frac{\partial^2 \phi}{\partial x^2} - D_y \frac{\partial^2 \phi}{\partial y^2} = 0 \quad (1)$$

It was compared using the following performance metrics: accuracy, stability and interpolation. DNN was found to be a promising alternative approach to FDM-based approaches to finding the solutions to PDEs.

2.2 Five-point Stencil CNN for solving reaction-diffusion type equations

[9] used a five-point stencil convolutional neural network (FCNN) for solving reaction-diffusion type equations, such as heat, Fisher's and Allen-Cahn equations, on structured grids. Their proposed model was trained using two consecutive evolutions of an equation. The model combines the use of finite difference schemes and convolutional neural networks (CNN).

2.3 Physics-informed geometry-adaptive convolutional neural networks (PhyGeoNet) for solving parameterized steady-state PDEs on irregular domain

CNNs are most effective on rectangular domains with uniform grids[5]. Because of this constraint, [5] proposed a CNN model aimed at learning solutions of parametric PDEs on irregular domains without any labelled data. In this CNN model, a physics-constrained model was introduced together with elliptic coordinate mapping. This approach was found to be more superior than the physics-informed neural networks (PINN) with fully-connected neural network (FC-NN) [13] in terms of accuracy and efficiency.

2.4 Neural network method for solving nonlinear fractional advection-diffusion equation with spatiotemporal variable-order

Here, a trained neural network method (NNM) is used to find the solution of the nonlinear fractional advection-diffusion equation. The network makes use of shifted Legendre orthogonal polynomials with adjustable coefficients. Results show that the method performed better than some FDMs in solving some nonlinear variable fractional order problems.

2.5 Graph convolutional networks applied to unstructured flow field data

Here, a graph convolutional neural network (GCNN) is used to predict properties from spatially unstructured data. The trained GCNN model, using inductive convolutional layers and adaptive pooling, achieved high accuracy with a Normalized Mean Squared Error below 0.01 [4].

3 Problem Description and Objectives

In a broad sense, nuclear modelling using AI at exascale level is capable of optimizing the following:

- Uncertainty quantification,
- Discretization and model error analysis,
- AI Physics modelling,
- Data assimilation,
- Validation,
- Adaptive targeted observations

More specifically, programming CFD/nuclear models using AI software enables interoperability between GPUs, CPUs and AI computers (e.g the energy efficient exascale Cerebras CS2) whilst exploiting community AI software.

This project, therefore, attempts to address these two main problems using AI software:

- CFD/nuclear codes can be computational expensive especially when attempting to run high-resolution code.
- CFD/nuclear codes can be extremely complex limiting the number of developers capable of writing such codes.

Our approach aims at implementing simple models using AI software. With this simplification, the number of developers capable of developing current CFD/nuclear codes will increase whilst speeding up the implementation of developments and their parallel scalability.

This new approach also makes possible simple development of digital twins by using the optimisation engine and sensitivities embedded in AI software. Digital twins are used to optimise systems, form error measures, assimilate data and quantify uncertainty in a relatively straightforward manner. With the use of digital twins, the major deficiency, i.e. formation of an error estimate, in current modelling approaches for safety critical systems in nuclear engineering and the environment is addressed.

4 Progress to Date and Future Plan

So far, a number of PDEs have been solved using CNNs on structured grids, such as:

- advection equation,
- diffusion equation,
- heat-diffusion equation,
- advection-diffusion equation.

Here, we will look at the heat-diffusion equation and the advection-diffusion equation. The architecture of the CNN used is shown in table 1 below.

Layer	Parameters	Settings
Convolution Layer	Kernel size	3
	Padding	'same'
	Stride	1

Table 1: CNN Parameters

4.1 Heat-Diffusion Equation

The 2-D heat-diffusion equation (shown in equation 2) below has been solved using CNNs. The equation is discretised using second-order central differencing scheme, shown in equation 3.

$$\frac{\partial T}{\partial t} = \Delta T, (\alpha = 1) \tag{2}$$

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} = \frac{1T_{i+1,j}^{n} + 1T_{i-1,j}^{n} + 1T_{i,j+1}^{n} + 1T_{i,j-1}^{n} - 4T_{i,j}^{n}}{\Delta x^2} \tag{3}$$

Here, T is temperature, t is time, α is the diffusion coefficient, i and j represent the cell along the x and y axes respectively and n refers to the time-step.

Equation 3 results in a 3x3 CNN filter with corresponding weights shown in 4 below, with each forward pass acting as one-time step. The results from multiple forward passes is shown in figure 1.

$$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix} \tag{4}$$

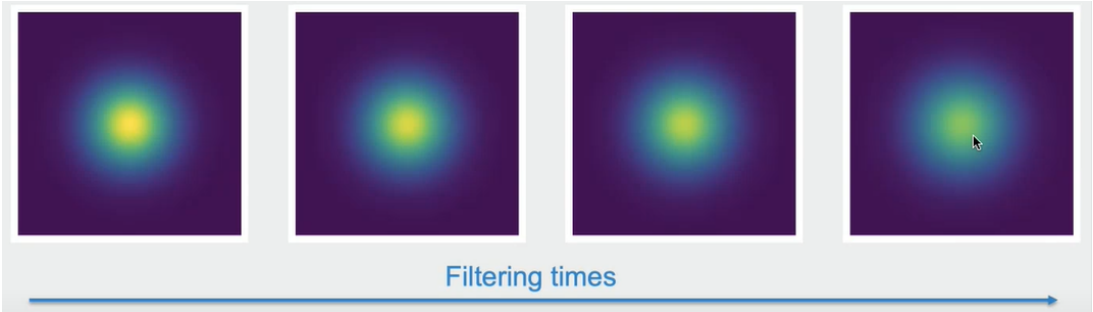

Figure 1: Discretisation of the Heat-Diffusion equation using CNNs

4.2 Advection-Diffusion Equation

In a similar manner described above, the advection-diffusion equation has also been solved using CNNs. The advection terms are discretized using central differencing while the diffusion terms are discretized using second-order central differencing. Equation 5 is the advection-diffusion governing equation, while equation 6 is the discretised equation.

$$\frac{\partial T}{\partial t} + u.\nabla T = \Delta T, (\alpha = 1) \tag{5}$$

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} + u\frac{1T_{i+1,j}^{n} - 1T_{i-1,j}^{n}}{2\Delta x} + v\frac{1T_{i,j+1}^{n} - 1T_{i,j-1}^{n}}{2\Delta y} = \frac{1T_{i+1,j}^{n} + 1T_{i-1,j}^{n} + 1T_{i,j+1}^{n} + 1T_{i,j-1}^{n} - 4T_{i,j}^{n}}{1\Delta x^2} \tag{6}$$

(7) below represents the CNN filter with weights corresponding to the advection terms while (8) is that of the diffusion terms.

$$\begin{matrix} 0 & 1/2 & 0 \\ -1/2 & -4 & 1/2 \\ 0 & -1/2 & 0 \end{matrix} \tag{7}$$

$$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix} \tag{8}$$

The results gotten from multiple forward passes using a combination of the advection and diffusion filters with weights is shown in figure 2 below.

4.3 Future Plan

In this project, we aim to solve the 2D advection-diffusion (shown in (9) below) equation on an unstructured grid. Modeling flow through unstructured grids are practical because flow in real life is irregular.

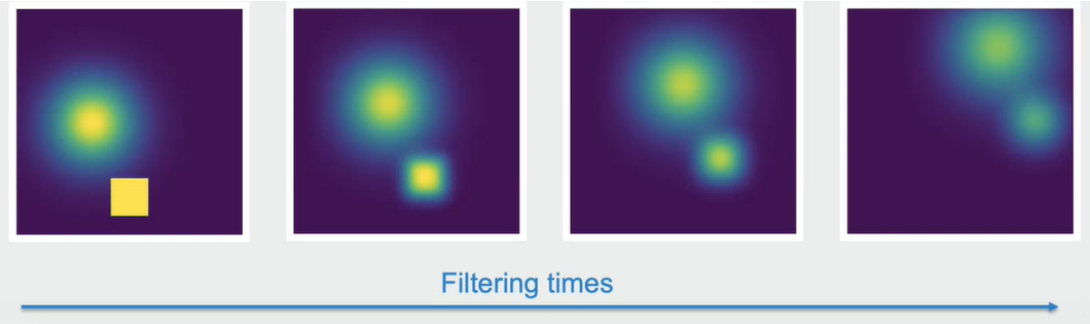

Figure 2: Discretisation of the Advection-Diffusion equation using CNNs

$$\frac{\partial T}{\partial t} + u.\nabla T = \Delta T \tag{9}$$

A number of neural networks have been adopted to solve PDEs on unstructured grids, such as PINN with FC-NN [13], PhyGeoNet [5], neural network method (NNM) [17] and GCNN [4] - a variant of graph convolutional network(GCN). Graph Neural Networks (GNNs) are deep-learning models designed to perform inference on data described as graphs [7]. There are various types of GNNs, such as GCN, graph attention network (GAT) and graph recurrent network (GRN) [7].

The methodology to achieve the project aim is by the use of the GCNN. GCNNs are the generalization of CNNs [4] but for operation on graphs, i.e. modeling using irregular data which is a limitation using CNNs. The GCNN produced will make use of an inductive convolution, which allows for prediction on meshes with varying resolutions. The GCNN will not be trained, instead, weights will be derived through discretizing the ADE.

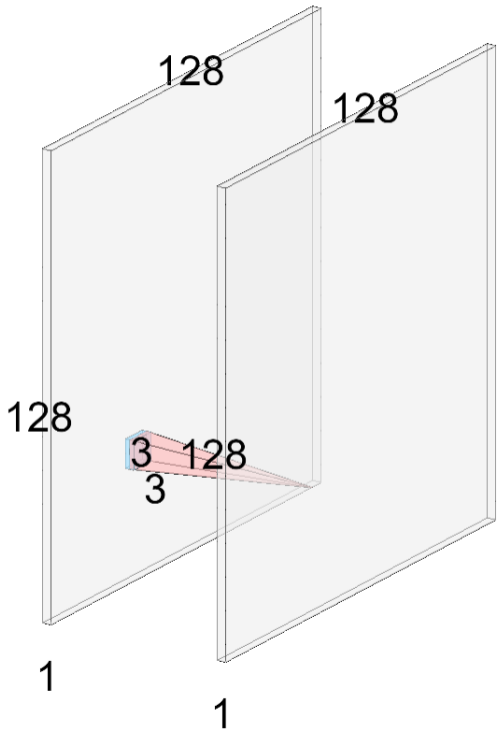

Figure 3: An example CNN using a 3x3 filter where input size and output size are the same. For a GCNN, the input and output will be of an irregular shape.

In this project, the following milestones are expected to be met using neural networks:

- Solution of the advection-diffusion equation using sparse layers on a structured and unstructured grid.
- Solution of the advection-diffusion equation using a GCNN on a structured grid and unstructured grid.

Figure 4 shows an image depiction of future plans and expected project timelines.

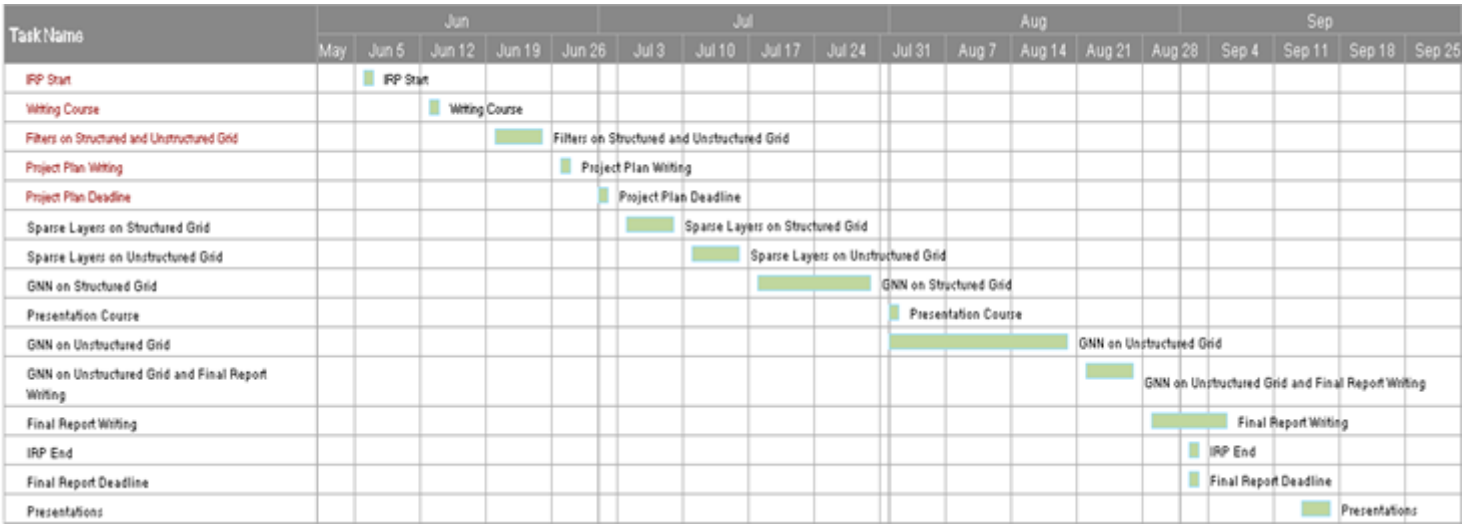

Figure 4: Gantt Chart displaying expected Project Timelines

References

[1] A. Ebrahimijahan, M. Dehghan, and M. Abbaszadeh. Compact local integrated radial basis functions (Integrated RBF) method for solving system of non–linear advection-diffusion-reaction equations to prevent the groundwater contamination. *Engineering Analysis with Boundary Elements*, 121:50–64, 2020.

[2] Y. Esmaeelzade Aghdam, H. Mesgarani, G. M. Moremedi, and M. Khoshkhahtinat. High-accuracy numerical scheme for solving the space-time fractional advection-diffusion equation with convergence analysis. *Alexandria Engineering Journal*, 61(1):217–225, 2022.

[3] D. Feng, I. Neuweiler, U. Nackenhorst, and T. Wick. A time-space flux-corrected transport finite element formulation for solving multi-dimensional advection-diffusion-reaction equations. *Journal of Computational Physics*, 396:31–53, 2019.

[4] Ogoke Francis, Meidani Kazem, Hashemi Amirreza, and Farimani Amir Barati. Graph convolutional networks applied to unstructured flow field data. *Machine Learning: Science and Technology*, 2(4), 2021.

[5] H. Gao, L. Sun, and J. X. Wang. PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. *Journal of Computational Physics*, 428(110079), 2021.

[6] Howard H. Hu. Computational fluid dynamics, 2012.

[7] Zhou J., G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[8] Salman A. K., A. Pouyaei, Y. Choi, Y. Lops, and A. Sayeed. Deep learning solver for solving advection-diffusion equation in comparison to finite difference methods. *SSRN Electronic Journal*, 2022.

[9] Y. Kim and Y. Choi. Fcnn: Five-point stencil cnn for solving reaction-diffusion equations, 2022.

[10] Z. Lau, I. M. Griffiths, A. English, and K. Kaouri. Predicting the spatially varying infection risk in indoor spaces using an efficient airborne transmission model, 2020.

[11] C. Pierre El Soueidy, A. younès, and P. Ackerer. Solving the advection-diffusion equation on unstructured meshes with discontinuous/mixed finite elements and a local time stepping procedure. *International Journal for Numerical Methods in Engineering*, 79(9):1068–1093, 2009.

[12] A. Preuss, J. Lipoth, and R. J. Spiteri. When and how to split? A comparison of two IMEX splitting techniques for solving advection–diffusion–reaction equations. *Journal of Computational and Applied Mathematics*, 414(114418), 2022.

[13] M. Raissi, Perdikaris P., and Karniadakis G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, 2019.

[14] P. Ren, C. Rao, Y. Liu, J. X. Wang, and H. Sun. PhyCRNet: Physics-informed convolutional-recurrent network for solving spatiotemporal PDEs. *Computer Methods in Applied Mechanics and Engineering*, 389(114399), 2022.

[15] G. Stephenson Stephenson and G. (Geoffrey) Stephenson. Partial differential equations for scientists and engineers, 1985.

[16] Sun Y., A. S. Jayaraman, and G. S. Chirikjian. Approximate solutions of the advection-diffusion equation for spatially variable flows. *Physics of Fluids*, 34(3), 2022.

[17] Wu Z., S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks.