# EXPLORATORY PROJECT REPORT

# APPLICATION OF MACHINE LEARNING IN DIFFERENTIAL EQUATIONS

Exploratory project submitted in partial fulfillment of the course CSM-291 for
INTEGRATED DUAL DEGREE(B.TECH + M.TECH) in
***Mathematics and Computing***

Submitted by

*PRANAV S 20124053*
*NOORSHABA 20124031*
*ADITYA KR 20124003*
*PRIYANKA SONI 20124055*

**DEPARTMENT OF MATHEMATICAL SCIENCES**
**INDIAN INSTITUTE OF TECHNOLOGY**
**BANARAS HINDU UNIVERSITY**
**VARANASI - 221 005**

# 1. CERTIFICATE

It is certified that the project report entitled " Application Of Machine Learning In Differential Equation" is submitted by **Mr. Pranav S (20124053), Mr. Aditya Kr Gautam (20124003), Ms. Noorshaba (20124031)** and **Ms. Priyanka Soni (20124055)**. This project is a record of students own work carried out by themselves under my guidance and supervision in partial fulfillment of the course **CSM - 291**(Exploratory Project) of **Department of Mathematical Sciences, IIT(BHU), Varanasi**

**Supervisor**
**Prof V K Singh**
**IIT BHU**

.

# 2. ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to our Supervisor Prof V K Singh, IIT BHU for his patience, motivation and immense knowledge. Their guidance helped us in all the times of the project and the work done by all of us. Moreover we thank the Department of Mathematical Sciences IIT BHU for providing the opportunity to be a part of this project and for helping us learn various theories and technologies required for the completion of this project.

# 3. ABSTRACT

This exploratory project was done as a part of the course CSM-291 by the students of the Department of Mathematical Sciences IIT BHU in order to explore and understand the research opportunities and developments in the respective department.

In this project, we saw how to discover the underlying physical law expressed by partial differential equations from scattered data collected in space and time. We learnt various aspects related to machine learning and its application in solving Partial Differential Equations. In the project we reviewed two research papers.  They are as follows:-

1. Analysis of research paper by Maziar Raissi: This paper explored a deep learning approach for discovering nonlinear partial differential equations. Here we used two deep neural networks to approximate the unknown solution as well as the nonlinear dynamics. The first network acts as a prior on the unknown solution and essentially enables us to avoid numerical differentiations which are inherently ill-conditioned and unstable. The second network represents the nonlinear dynamics and helps us distill the mechanisms that govern the evolution of a given spatiotemporal data-set. Later the effectiveness of the approach was evaluated using several benchmark problems and the accuracy was confirmed.

2. Analysis of research paper by Hayden Schaeffer: This paper investigates the problem of learning an evolution equation directly from some given data. This work develops a learning algorithm to identify the terms in the underlying partial differential equations and to approximate the coefficients of the terms only using data. The algorithm uses sparse optimization in order to perform feature selection and parameter estimation. The features are data driven in the sense that they are constructed using nonlinear algebraic equations on the spatial derivatives of the data. The robustness of the method was then verified using several numerical methods.

We critically evaluated both these papers and analysed the method of approach undertaken in both of them. When the first paper dealt with the case when the form model is known the second paper dealt with the method to find the solution when the form model is unknown. Automatic differentiation and spectral method combined with sparse optimisation were used in the papers respectively to reach an accurate prediction of the solution using neural networks.

After critical evaluation and in depth learning regarding the topic we came to a conclusion depicted in the last part of this report. This project opened the door to the vast field of the application of ML in mathematical problems to us. More research can be done in this field and developments can be achieved in the near future.

# CONTENTS

# 4. INTRODUCTION

Mathematics and machine learning are like two sides of a coin. The world has been looking eagerly into the research interconnecting both of the fields for a long time. There exist a lot of mathematical models. From the flapping of wings by a butterfly to a dangerous hurricane or atomic explosion anything that exists around us can be expressed mathematically. Use of these mathematical expressions helps us to analyse certain things and predict the outcome and the future states. But in many real life cases these mathematical calculations become too complex to come up with a solution for a normal human being. This leads to the use of machine learning models to solve mathematical relations. It is pretty interesting to work with machine learning to solve mathematical problems as it itself originates from mathematics.

Machine Learning(ML) is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine Learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. In the present world Machine Learning and AI have developed that much that the countries having the greatest advancement in this field are now considered the strongest in the world. The day where machines talk, see and interact better than humans is so close. Machine learning and mathematics are considered as the best couples in the scientific world.

Finding the solutions of differential equations are much necessary in several prediction cases such as predicting the trajectory of a cyclone or the change in the stock prices. But in most of these cases the differential equations that need to be solved become much more complex. An effective implementation of machine learning algorithms can make this task simple. Research has been going on in this field for a long time. Physics Informed Neural Networks (PINNs), Automatic differentiation, Sparse optimisation etc were developed as a result of these researches. But still this field is still in its infancy and needs much more research and improvements.

These facts led us to choose and explore “The Application of Machine Learning in Differential Equations” as our Exploratory project under the Department of Mathematical Science guided by Prof V K Singh.

Here we are discussing two research papers that are published in the topics related to the application of machine learning in solving differential equations. They are

1. Deep Hidden Physics Models: Deep learning of non linear Partial Differential Equations by Maziar Raissi.
2. Partial differential equation via data discovery and sparse optimization by Hayden Schaeffer.

# 5. DEEP HIDDEN PHYSICS MODELS: DEEP LEARNING OF NON-LINEAR PARTIAL DIFFERENTIAL EQUATIONS

This paper provided a deep learning approach for discovering nonlinear partial differential equations from scattered and potentially noisy observations in space and time. Here two neural networks were used to approximate the unknown solution as well as the nonlinear dynamics. The first network acts as a prior on the unknown solution and essentially enables us to avoid numerical differentiations which are inherently ill-conditioned and unstable. The second network represents the nonlinear dynamics and helps us distill the mechanisms that govern the evolution of a given spatiotemporal data-set. Later the algorithms were tested against several benchmark problems and it was confirmed that the proposed framework can learn and accurately learn the underlying dynamics and forecast the future states.

## 5.1. Evolution of the framework from physics informed deep learning method

Physics-informed deep learning method contains structured nonlinear regression models that can uncover the dynamic dependencies in a given set of spatio-temporal dataset and return a closed form model that can be subsequently used to forecast the future states. Also, we are using a richer class of function approximators to represent the nonlinear dynamics and consequently we do not have to commit to a particular family of basis functions.

Specifically, we consider nonlinear partial differential equations of the general form

$$u_t = N(t, x, u, u_x, u_{xx}, \ldots) \quad (1)$$

where N is a nonlinear function of time t, space x, solution u and its derivatives Here, the subscripts denote partial differentiation in either time t or space x. 2 Given a set of scattered and potentially noisy observations of the solution u, we are interested in learning the nonlinear function N and consequently discovering the hidden laws of physics that govern the evolution of the observed data.

## 5.2. Drawbacks of the above method

1. It relied on numerical differentiation to compute the derivatives involved in equation (1). Derivatives are taken either using finite differences for clean data or with polynomial interpolation in the presence of noise. Numerical approximations of derivatives are inherently ill-conditioned and unstable even in the absence of noise in the data. This is due to the introduction of truncation and round-off errors inflicted by the limited precision of computations and the chosen value of the step size for finite differencing. Thus, this approach requires far more data points than library functions. This need for using a large number of points lies more in the numerical evaluation of derivatives than in supplying sufficient data for the regression.
2. In applying the algorithm outlined above we assume that the chosen library is sufficiently rich to have a sparse representation of the time dynamics of the dataset. However, when applying this approach to a dataset where the dynamics are in fact unknown it is not unlikely that the basis chosen above is insufficient. Specially, in higher dimensions (i.e., for input x or output u) the required number of terms to include in the library increases exponentially. Moreover, an additional issue with this approach is that it can only estimate parameters appearing as coefficients. For example, this method cannot estimate parameters of a partial differential equation (e.g., the sine-Gordon equation) involving a term like $\sin(\alpha u(x))$ with $\alpha$ being the unknown parameter, even if we include sines and cosines in the dictionary of possible terms.

## 5.3. Methods to resolve the drawbacks

1. First drawback concerning numerical differentiation by assigning prior distributions in the forms of Gaussian processes or neural networks to the unknown solution u. Derivatives of the prior on u can now be evaluated at machine precision using symbolic or automatic differentiation. This removes the requirement for having or generating data on derivatives of the solution u. This is enabling as it allows us to work with noisy observations of the solution u, scattered in space and time. Moreover, this approach requires far fewer data points simply because, as explained above, the need for using a large number of data points was due to the numerical evaluation of derivatives.
2. The second drawback can be addressed in a similar fashion by approximating the nonlinear function N (see equation (1)) with a neural network. Representing the nonlinear function N by a deep neural network is the novelty of the current work. Deep neural networks are a richer family of function approximators and

consequently we do not have to commit to a particular class of basis functions such as polynomials or sines and cosines. This expressiveness comes at the cost of losing interpretability of the learned dynamics. However, there is nothing hindering the use of a particular class of basis functions in order obtain more interpretable equations

## 5.4. Solution Methodology

An efficient solution was proposed in the paper. Here approximated both the solution u and nonlinear function ***N*** with two deep neural networks and define a deep physics model ***f*** given by

$$\boldsymbol{f := u_t - N\,(t, x, u, u_x, u_{xx}, \ldots)} \qquad (2)$$

Here the derivatives of the neural network u were obtained with respect to time t and space x by applying the chain rule for differentiating compositions of functions using automatic differentiation.

---

### How does Automatic differentiation help here?

Automatic differentiation is different from, and in several aspects superior to, numerical or symbolic differentiation; two commonly encountered techniques of computing derivatives.

Automatic differentiation relies on the fact that all numerical computations are ultimately compositions of a finite set of elementary operations for which derivatives are known. Combining the derivatives of the constituent operations through the chain rule gives the derivative of the overall composition. This allows accurate evaluation of derivatives at machine precision with ideal asymptotic efficiency and only a small constant factor of overhead.

---

Parameters of the neural networks u and N can be learned by minimizing the sum of squared errors

$$\sum_{i=1}^{N}\left(\left|u(t^i, x^i) - u^i\right|^2 + \left|f(t^i, x^i)\right|^2\right) \qquad (3)$$

Here the term $\left|u\left(t^i, x^i\right) - u^i\right|^2$ tries to fit the data by adjusting the parameters of the neural network u while the term $\left|f\left(t^i, x^i\right)\right|^2$ learns the parameters of the network N by trying to satisfy the partial differential equation (1) at the collocation points ($t^i$ , $x^i$). Training the parameters of the neural networks u and N can be performed simultaneously by minimizing the sum of squared error (3) or in a sequential fashion by training u first and N second.

Here the learned function N is a black box function; i.e We do not know its functional form. So to solve the equations we have to resort to modern black box solvers such as PINNS - Physics Informed Neural Networks. The steps involved in PINNs as solvers are similar to equations (1), (2), and (3) with the nonlinear function N being known and the data residing on the boundary of the domain.

The proposed framework provides a universal treatment of nonlinear partial differential equations of fundamentally different nature. This generality will be demonstrated by applying the algorithm to a wide range of canonical problems spanning a number of scientific domains. In the paper they proved the generality by applying the algorithm to the Burgers', Korteweg-de Vries (KdV), Kuramoto-Sivashinsky, Nonlinear Schrödinger, and Navier-Stokes equations. In the report we are taking the Burger's equations case to elaborate the method.

## 5.5. Evaluation of the model using Burger's equation

In one space dimension Burger's equation is given as

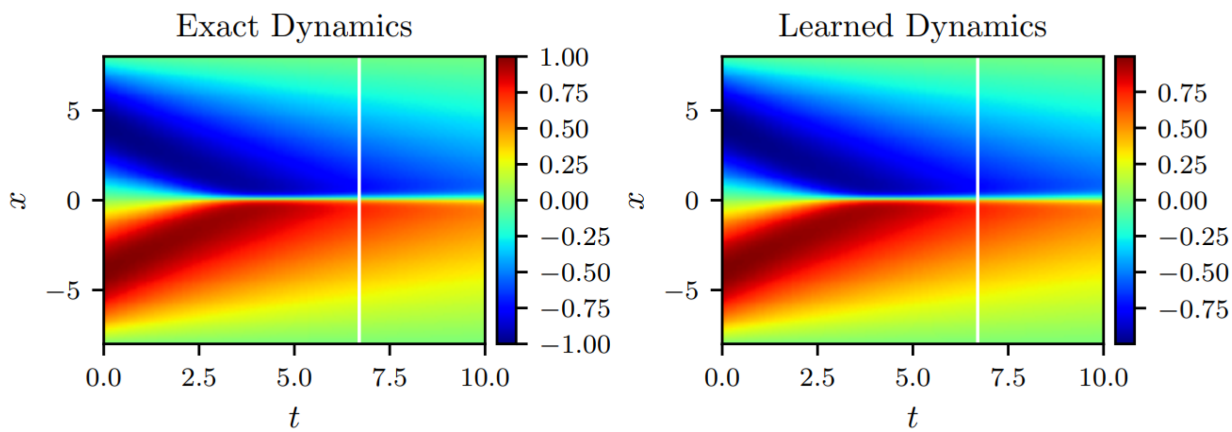
$$u_t = -uu_x + 0.1u_{xx} \tag{4}$$

To obtain a set of training methods and the test data the Burger's equation was stimulated using conventional spectral methods. Specifically starting from an initial condition $u(0, x) = -\sin(\pi x/8)$, $x \in [-8, 8]$ and assuming the periodic boundary condition we integrate equation (4) up to a final time $t = 10$. We use the Chebfun package with a spectral Fourier discretization with 256 modes and a fourth-order explicit Runge-Kutta temporal integrator with time-step size $10^{-4}$ . Then the data set is sampled. Given the training data, we are interested in learning N as a function of the solution u and its derivatives up to the 2nd order ; i.e.,

$$u_t = N(u, u_x, u_{xx}). \tag{5}$$

We represent the solution u by a 5-layer deep neural network with 50 neurons per hidden layer. Furthermore, we let N to be a neural network with 2 hidden layers and 100 neurons per hidden layer. These two networks are trained by minimizing the sum of squared errors and loss of equation (3). Then the learned partial differential equation(5) is solved using PINNs algorithm to check the effectiveness.

The original solution (left) and the solution obtained by the learned partial differential equation (right) are as follows.



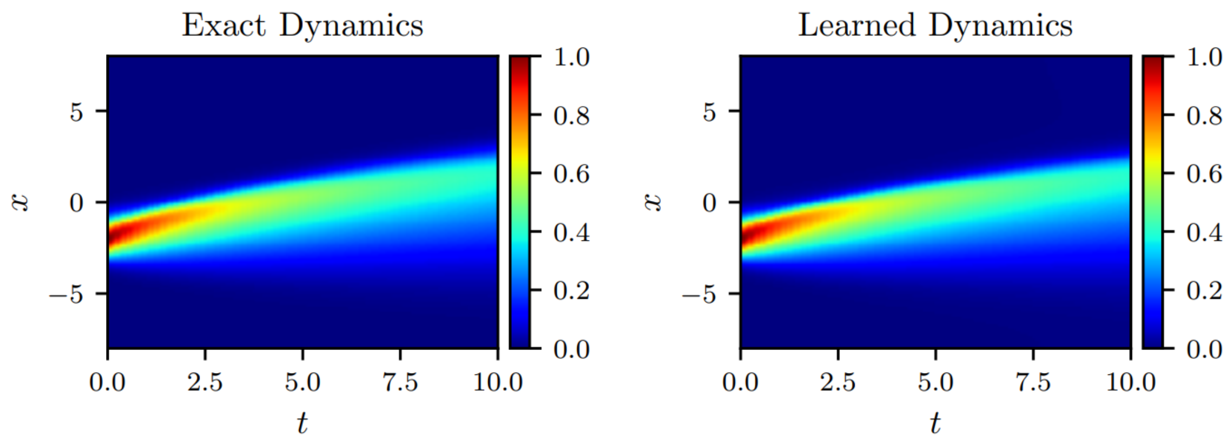

The identified system correctly captures the form of the dynamics and accurately reproduces the solution with a relative L 2 -error of 4.78e-03. It should be emphasized that the training data are collected in roughly two-thirds of the domain between times t = 0 and t = 6.7 represented by the white vertical lines. The algorithm is thus extrapolating from time t = 6.7 onwards. The relative $L^2$ -error on the training portion of the domain is 3.89e-03.

Later the dynamics was further checked by varying the noise in the data used and the relation between the accuracy and the noise at fixed no of training observations were examined. Here the neural network architectures were fixed. The result of the study is as follows

| | Clean data | 1% noise | 2% noise | 5% noise |
|---|---|---|---|---|
| Relative $L^2$ error | 4.78e-03 | 2.64e-02 | 1.09e-01 | 4.56e-01 |

This indicates that the negative consequences of more noise in the data can be remedied to some extent by obtaining more data.

Later to further scrutinize the efficiency of the algorithm the initial condition was changed to $-\exp(-(x+2)^2)$ and the algorithm was applied. The result obtained is depicted in the figure. A solution to the Burger's equation (left panel) is compared to the corresponding solution of the learned partial differential equation (right panel).




The $L^2$ -error obtained here was of 7.33e-02.

Consequently, in cases of practical interest, the available information on the boundary of the domain could help us determine the order of the partial differential equation we are trying to identify. Using this idea the robustness of the algorithm was studied and it was found out that the method seems to be generally robust with respect to the number and order of derivatives included in the nonlinear function N . Therefore, in addition to any information residing on the domain boundary, studies conducted like this, albeit for training or validation datasets, could help us choose the best order for the underlying partial differential equation. It was also found out that including higher order derivatives reduces the speed of the algorithm because it makes the resulting computational graph for the corresponding deep hidden physics model more complex. It also reduces the accuracy.

In the same way Korteweg-de Vries (KdV), Kuramoto-Sivashinsky, nonlinear Schr¨odinger, and Navier-Stokes equations were used and the efficiency of the proposed algorithm was validated.

## 5.6. Inference from the paper

In the research paper, a deep learning approach for extracting nonlinear partial differential equations from spatio - temporal dataset were discussed in detail. The algorithm proposed in the paper leverages the recent developments in automatic differentiation to construct efficient algorithms for learning infinite dimensional dynamic systems using deep neural networks. But, these types of methods and solvers are still in their infancy. There are a lot more questions existing in this field on which further research is needed. Some of them are:-

1. Many real-world partial differential equations depend on parameters and, when the parameters are varied, they may go through bifurcations.

2. Application of convolutional architectures for mitigating the complexity associated with partial differential equations with very high-dimensional inputs.

After that we evaluate the research paper written by Hayden Schaeffer on Partial Differential equations via Data Discovery & Sparse Optimization. Our findings are as follows:-

# 6. PARTIAL DIFFERENTIAL EQUATION VIA DATA DISCOVERY & SPARSE OPTIMIZATION

This research paper investigates the problem of learning an evolution equation directly from some given data. This work develops a learning algorithm to identify the terms in the underlying PDE and to approximate the coefficients of terms only using data. The algorithm uses sparse optimization. In recent years, sparsity and sparse optimization have been applied to problems in differential equations and scientific computing. In this work, the linear system is typically solved via an L1 regularized least-squares minimization. The L1 norm is used to penalize the number of non-zero coefficients in order to promote coefficient sparsity.

## 6.1. Motivation and model:

We considered the case of viscous Burgers, equation and created a general approach to solve PDEs.

With the viscous Burger equation given by:

$$u_t = (u^2/2)_x + v u_{xx}$$

Here we considered an unknown form of equation as the solution and approached it with a general form of $2^{nd}$ order PDE with quadratic nonlinearity. The r.h.s. was obtained by approximating a general second-order evolution equation $u_t=F(u,u_x,u_{xx})$ via a second-order Taylor expansion.

$$u_t = \alpha_1 + \alpha_2 u + \alpha_3 u^2 + \alpha_4 u_x + \alpha_5 u_x^2 + \alpha_6 uu_x + \alpha_7 u_{xx} + \alpha_8 u_{xx}^2 + \alpha_9 uu_{xx} + \alpha_{10} u_x u_{xx}.$$

$$u_t = [1\ u\ u^2\ u_x\ u_x^2\ uu_x\ u_{xx}\ u_{xx}^2\ uu_{xx}\ u_x u_{xx}] \cdot \alpha$$

Considering $f_l(t)$ as the feature vector, the collection of feature vectors defines the feature matrix, i.e. $F_u(t) = [f_l(t)]$

V(t) is the velocity vector. Here we get this problem liner in α: $V(t) = F_u(t)\,\alpha$

This equation is solved using $L^1$ regularized least square minimisation given by:

$$min_{\alpha} \frac{1}{2} \sum_{k=1}^{M} \left|\left| V(t_k) - F_u(t_k) \right|\right|_2^2 + \lambda||\alpha||_1$$

Here λ > 0 is a balancing parameter. The L1 regularization was chosen to promote sparsity in the vector α, with the modelling assumption that the underlying dynamics were governed by a few terms.

## 6.2. Numerical method and Algorithm

From the inference obtained by solving viscous Burgers' equation we move on to a generalized method. If we have the data w(x,t) at (possibly non-uniform) grid points $x_j$ for $1 \le j \le N$ and time stamps $t_k$ for $1 \le k \le M$, we approximate the underlying evolution equation $w_t = G(w)$ by first computing numerical approximations to the spatial derivatives of w(x,t) using spectral method and then by constructing a feature matrix by collection of feature vectors.

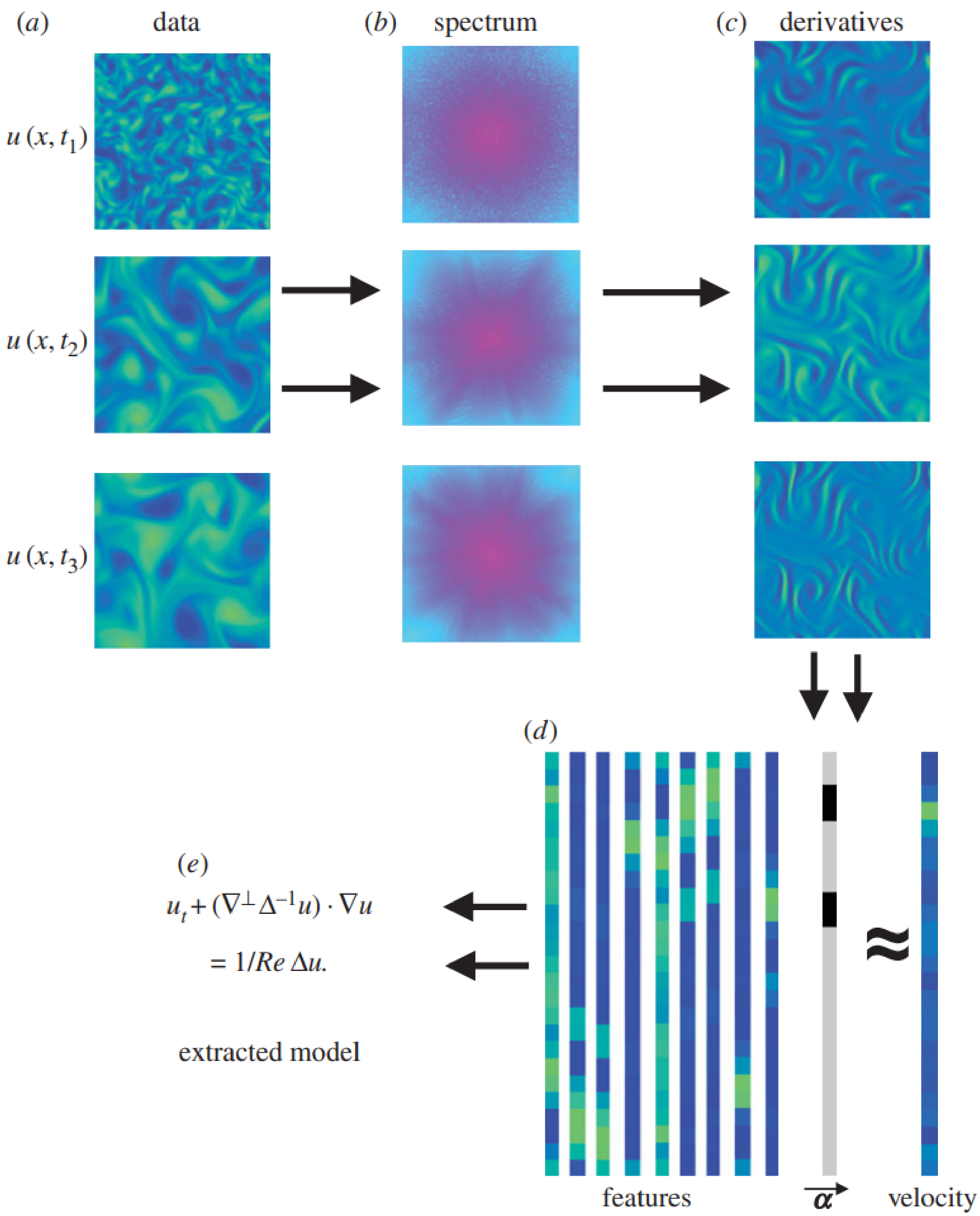
Finally the coefficients of the terms are computed using the least squares *method*.

$$min_{\alpha} \frac{1}{2} \sum_{k=1}^{M} \left|\left| V(t_k) - F_u(t_k) \right|\right|_2^2 + \lambda||\alpha||_1$$

The minimizer satisfies the following inclusion relation:

$$\sum_{k=1}^{M} (F_w^T(t_k) F_w(t_k) + \sum_{k=1}^{M} F_w^T(t_k) V(t_k) + \lambda\delta||\alpha||_1$$

The above equation is solved using the Douglas-Rachford algorithm.




**Visual algorithm:**The data are given as a sequence at different time stamps (a). Then, the data are converted to the Fourier domain (b) and the spatial derivatives are calculated (c). The derivatives are combined to construct features, which are then vectorized(d). An L 1 optimization algorithm is used to extract the sparsity coefficientsα that fit the features to the velocity.The coefficients α are unravelled to reveal the underlying model (e).

## 6.3. Simulation and numerical experiments

- To test the method,
  - Data are simulated using the Crank–Nicolson method. The simulated data u(x,t) is assigned to the variable $w^0(x,t)$.
  - The noise-free velocity $w^0_t$ is computed by a first-order backward difference: $w_t^0(x_j, x_t) \approx \left[w^0(x_j, t_k) - w^0(x_j, t_{k-1})\right]/dt$, where dt > 0 is the time step.
  - The computed velocity $w^0_t$ is corrupted by additive Gaussian noise. The noisy velocity is assigned to $w_t$.
  - The noise level is displayed for all examples by measuring the ratio between the L2 norm of the noise and the L2 norm of the data, i.e.

    $$\textbf{\textit{noise level}} = \left(\left\|w_t - w_t^0\right\|_2 \times 100\%\right) / \left\|w_t^0\right\|_2$$

- All spatial derivatives are approximated using **the spectral method,** and features are computed via direct products. All features are computed using $w^0_t(x_j,t_k)$.
- In all examples, the Douglas–Rachford algorithm parameters are set to γ = 1/2, μ = 1, and the maximum number of iterations to 5000. Convergence of the iterates is achieved before the maximum iteration bound is reached.

## 6.4. Examples and Applications

### Viscous Burgers' equation:

$$u_t = (u^2/2)_x + \nu u_{xx}$$

where $\nu>0$ is viscosity and $x\in[0,1]$

Data are simulated using the Crank–Nicolson method and balance parameter is fixed to λ = 500. The feature space is made up of the terms from a third-order Taylor expansion using up to second-order derivatives.The proposed algorithm is applied to the data with various levels of noise. For all noise levels, 2.0%, 19.5%, and 94.5%, the coefficients are identified to be within 1% in relative error.

### Cahn–Hilliard equation :

$$u_t = -\gamma\Delta^2 u + \Delta(u^3 - u)$$

In this example, we will assume that the model takes the form: $u_t = \Delta F(u)$. The data are simulated up to T = 2 using the Crank–Nicolson method with γ = 0.5. The learning is stable with respect to redundancies in the feature space. The input data to the learning method use uniform time steps with a total of 385, 85 and 25 time stamps. In all cases,the noise level is kept at 5.0%.The Learned Coefficient shows that the method is robust to the number of time stamps used in the learning process. This suggests that the proposed method is robust to the **size of the data.**

## 6.5. Inference:

This work presented L1 regularized least-squares minimization to select the correct features that learn the governing PDE and the identification of the terms in the PDE (form of model) and the approximation of the coefficients come directly from the data. The numerical experiments show that the proposed method is robust to data noise and size, is able to capture the true features of the data, and is capable of performing additional tasks such as extrapolation, prediction and (simple) homogenization. But, model ambiguity is a potential issue with this approach when a large feature space is used. Also there is a potential issue with this approach when applied directly to noisy data.

# 7. COMPARISON OF BOTH THE PAPERS

## Paper By Maziar Raissi

- A deep learning approach using neural networks for extracting nonlinear partial differential equations from spatio-temporal datasets.
- Form of the model is known.
- Derivatives of u with respect to time t and space x is obtained by using **<u>Automatic differentiation.</u>**
- The proposed framework provides a universal treatment of nonlinear partial differential equations of fundamentally different nature. This generality is demonstrated by applying the algorithm to a wide range of problems .

## Paper By Hayden Schaeffer

- Sparse optimization method for identifying partial differential equation to a given data set.
- Form of the model is not known.
- We construct derivative approximations using the **<u>spectral method and sparse optimisation</u>**.
- The proposed method is robust to data noise and size, and can capture the actual features of the data.

# 8. INFERENCE AND RESULT

In this exploratory project we went through two papers: Hidden physics model: Deep Learning of Nonlinear Partial Differential Equations by Maziar Raissi and Partial Differential equations via Data Discovery & Sparse Optimization by Hayden Schaeffer. Both the papers were quite interesting and also the methods and theories mentioned were new to us but were also very exciting to learn. We went through both the papers in depth and they opened the window of the vast field of various machine learning techniques that can be applied in complex mathematical problems. It was fun to explore and use Machine Learning which is born from mathematics to solve mathematical problems itself.

We were exposed to a vast range of new methods, algorithms and scientific terms through the project. For the better understanding of the two research papers we had to explore various resources and utilize them to gain in depth knowledge of the subject.

In the initial phase we deeply went through the various solution methods for differential equations. After that we all did several online courses related to machine learning and made an understanding of the basic framework of machine learning algorithms. Later we started to learn about the possibilities of implementation of ML algorithms to solve differential equations.

Through the project we learned about automatic differentiation, sparse optimisation and spectral methods. We acquired the skill to review and analyze various types of research papers.

Both the papers that we dealt with in the project were related to the topic "Application of machine learning in differential equations". While the paper by Maziar Raissi proposed a method to find the solution of partial differential equation when the form model is known using automatic differentiation the paper by Hayden Schaeffer dealt with the method to find the solution of partial differential equations when the form model is unknown with the help of spectral method and sparse optimisation

Even though the methods proposed in both the papers were accurate and efficient in most of the cases there were situations where the proposed methods were not much efficient to predict the solution. This calls for in depth research in the topic and the need for the introduction of more efficient methods.

# 9. SCOPE

The topic that we went through has a very high scope . By implementing machine learning to solve differential equations we can get the solution of more complex equations with a simple code if we have the sufficient data to process the output.

This is more useful in real life conditions where we have to solve differential equations of higher complexity and with large numbers of terms. Such as in fluid mechanics, stock prediction etc. By applying this technique we can understand the flow mechanism of a fluid in certain situations where it is nearly impossible or very difficult for humans to do the calculations and predict the mechanism.

In the case of financial mathematics these methods to solve differential equations can be utilized to predict the stock curve and this can be implemented in fintech applications.

# 10. FUTURE PLANS

2nd year: Go through the learned topics in depth and write a paper on ML in PDEs also write a review paper on both the research papers.

3rd year: Stream Project starts: Learn the application of these techniques in stock prediction.

4th year: Develop an application for stock prediction based on the algorithms learned and developed through research.

5th year: Improvement of the application. Publish a thesis based on the work done in the previous years.

# 11. REFERENCES


1. Research paper: Hayden Schaeffer rspa.2016.0446.pdf
2. Research paper: Maziar Raissi 1801.06637.pdf
3. Sparsity in Deep Learning 2102.00554.pdf
4. Youtube resources:
   - What is Sparsity?
   - Sparsity and the L1 Norm
   - Regularization Part 2: Lasso (L1) Regression
   - Regularization Part 1: Ridge (L2) Regression