RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

KASDI MERBAH UNIVERSITY OUARGLA

Faculty of Mathematics and Sciences of the Matter

Department of Mathematics

MASTER

Path: Mathematics

Option:modeling and numerical analysis.

Present by: Okba Kounta Hamadi

Theme

Deep Galerkin Method

Represented on: 07/07/2021

Limb from jury:

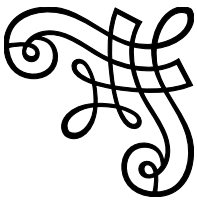
Chacha Djamal Ahmed	Prof	Université KASDI Merbah - Ouargla	Chairman
Ghezal Abderrazek	M.C.A.	Université KASDI Merbah - Ouargla	Examiner
Merabet Ismail	M.C.A.	Université KASDI Merbah - Ouargla	Examiner
Bensayah Abdallah	M.C.A.	Université KASDI Merbah - Ouargla	Supervisor

year: 2020-2021

Dedications

I give this work to the person who taught me, kept up my upbringing, and gave me everything he had from an early age until God passed to my father And to the good school that I am delighted and saddened by my grief that raised me and endured my mother's thumbs preserved by God.My brothers and sisters went to the lalla,sayat, nafissa, Amhamad, Bai, Fatima and Precious, to my grandmother, and to my teachers, who taught me from the primary level to the university,To my classmate Saidou Shibani to all my colleagues, who taught me throughout my course, from Ain Guazzam to Tamnrasset to OUARGLA, and to all my co-workers who helped me and guided me.Since joining all math students, I mention my colleagues in the 2021 class.

Okba Kounta Hamadi .

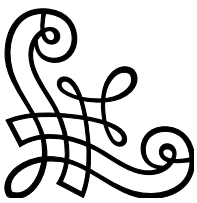
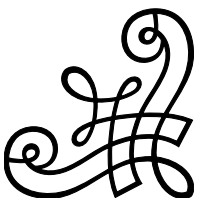



thanks

I thank God very much for
doing this job and for giving me the health, wellness
and will to do it And true to the words of R. Solna, who didn't thank
people, I did not thank God, I thank very much my professor, my role models and my supervisor in this work Who, despite the distance between us,
has kept in touch with me and coordinated and guided in every small and
large part of this work by Professor Dr. Ben Sayeh Abd allah Jazullah for me All the professors and students who didn't bless me,
and every time I asked them, they answered me with a chest,
God bless me I also very much thank the members of
the jury for discussing and evaluating this
work and for wearing their verdict on this work.

♡

Merci à tous

Abstract

In In this work, we analyzed the Deep Galerkin method, starting with deep learning,beginning with the emergence of artificial neural networks, using Universal Approximation Theorem of Neural Networks,where we applied this method to an parabolic problem to consolidate the study and demonstrated the convergence of the neural network to solve the PDE through an example of Trifinyl with tight conditions in which the error function is zero.

Résumé

Dans ce travail, nous avons analysé la méthode Deep Galerkin, en commençant par l'apprentissage profond, en commençant par l'émergence de réseaux neuronaux artificiels,en utilisant Universal Approximation Theorem of Neural Networks, où nous avons appliqué cette méthode à un problème parabolic pour consolider l'étude et démontré la convergence du réseau neuronal pour résoudre le PDE à travers un exemple de Trifinyl avec des conditions serrées dans lesquelles la fonction d'erreur est zéro.

ملخص

في هذا العمل قمنا بدراسة تحليلية لطريقة قالركين العميقة بداية من التعلم العميق وبداية ظهوره وظهور الشبكات العصبية الاصطناعية بالاستعانة بنظرية التقريب العالمية للشبكات العصبية حيث قمنا بتطبيق هذه الطريقة على مشكل مكافئ لتوطيد الدراسة وتبيين تقارب الشبكة لحل المعادلات التفاضلية الجزئية من خلال مثال ثلاثي الفنيل مع شروط محكمة بحيث تكون دالة الخطا تساوي الصفر

Contents

List of Figures

LIST OF SYMBOLS

- ▷ $L(\theta)$ the loss function.
- ▷ $f(t, x, \theta)$ approximation function.
- ▷ y_i real results.
- ▷ $\hat{y_i}$ Results obtained after prediction.
- ▷ $\mathbb{R}$ set of real numbers.
- ▷ $\mathbb{N}$ the set of natural integers.
- ▷ $\mathbb{C}$ set of complex numbers.
- ▷ $C(I)$ The continuous functions space of I in $\mathbb{R}$.
- ▷ $|.|$ the Euclidean norm of $\mathbb{R}^n$.
- ▷ Δ Laplatient.
- ▷ $\frac{\partial u}{\partial n}$ normale derivative of u.
- ▷ $\partial\Omega$ the boundary of Ω .
- ▷ ∇u the gradient of u.

- ▷ $\frac{\partial u}{\partial x}$ the partial derivative of u compared x.
- ▷ $\mathrm{V}^{'}$ $the dual space of \mathrm{V}$.

▷ $\|X\|$ L^2 norm of X.

▷ $f : \mathbb{R} \to \mathbb{R}$ The function with domain $\mathbb{R}$ and range $\mathbb{R}$.

▷ $\mathcal{D}^n$ The whole-order derivation.

▷ $I_n = [0, 1]^n$ is the n-dimensional unit cube.

▷ $M(I_n)$ space of finite, signed regular Borel measures on I_n.

▷ $C(I_n)$ space of continuous functions on I_n.

▷ $D(\Omega)$ the space of infinitely smooth functions with a compact support in Ω.

▷ $\to$ strong convergence.

▷ $\hookrightarrow$ Continous embedding.

INTRODUCTION GENERAL

numerical method that are based on grids can fail when the dimensionality of the problem becomes too large. Furthermore, even if we were to assume that the computational cost was manageable, ensuring that the grid is set up in a way to ensure stability of the nite dierence approach can be cumbersome. With this motivation, Sirignano and Spiliopoulos propose a mesh-free method for solving PDEs using neural networks. The Deep Galerkin Method (DGM) This circumvents the curse of dimensionality which is encountered with the latter approach PDEs using neural networks. With this parameterization, a loss function is set up to penalize the tted function's deviations from the desired dierential operator and boundary conditions.
In this thesis, we study the Deep Galerkin Method This thesis is split into fort main chapters as follows:

- The first chapter contains some basic concepts The most important functions, spaces and theories, with brief definitions, facilitate the study of this method.
- The second chapter is devoted to the study of deep learning, what it is, what its purpose is, why we are interested in studying it and giving it this attention from the study and what it has to do with the deep Galerkin Method and the neural network We looked at artificial neural networks in an integrated manner with their algorithms and their emergence up to

the latest developments.

- In the final chapter, we began to study the deep Qalerkin method that we had prepared to study in the rest of the previous chapters, where we introduced the method and its algorithm with the auxiliary theories to study with the thoughtful examples to which to apply this method. We chose the problem of the equivalent pieces to apply this method.

CHAPTER I

SOME MATHEMATICAL PRELIMNARIES

I.1 Some important theorems

I.1.1 Sobolev spaces:

Let p be a real number with $0 \leq p \leq \infty$, ω is an open subset of $\mathbb{R}^n$,The Sobolev space $W^{m,p}(\Omega)$ is defined to be:

$$W^{m,p}(\Omega) = \{u \in L^p(\Omega) | D^\alpha u \in L^p(\Omega), |\alpha| \leq m\}.$$

Where $D^\alpha v$ is the derivative in the sense of the distributions for all $v \in L^p(\Omega)$,The space $W^{m,p}(\Omega)$ equipped with the norm

$$\|u\|_{W^{m,p}} = \sum_{0 \leq |u| \leq m} \|D^\alpha u\|_p$$

Definition I.1.1. *In the special case where $p = 2$, we define the Hilbert-Sobolev space $H^m(\Omega) = W^{m,2}(\Omega)$*

$$\textit{for}, m \in \mathbb{N}, H^k(\Omega) = \{u \in L^2(\Omega) | D^\alpha u \in L^2(\Omega), |\alpha| \leq k\}$$

The space $H^m(\Omega)$ is equipped with the inner product

$$\langle u, v\rangle_{H^m} = \sum_{|\alpha|\leq k} \int_\Omega D^\alpha u D^\alpha v \mathrm{d}x,$$

and the norm

$$\|u\|_{H^m} = \sum_{0\leq|\alpha|\leq m} \|D^\alpha u\|_2$$

Theorem I.1. ***(Rellich-Kondrachov)*** *$\Omega \subset \mathbb{R}^n$ be a Lipschitz domain,$m \in \mathbb{N}$ and $1 \leq p \leq \infty$. Then, the following mappings are compact embeddings:*

1.$W^{m,p}(\Omega) \hookrightarrow L^q(\Omega), \quad 1 \leq q \leq p^*, \frac{1}{p^*} = \frac{1}{p} - \frac{m}{d}, if m < \frac{d}{p},$

2.$W^{m,p}(\Omega) \hookrightarrow L^q(\Omega), \quad q \in [1, \infty), if m = \frac{d}{p},$

3.$W^{m,p}(\Omega) \hookrightarrow C^0(\bar{\Omega}) \quad , if m > \frac{d}{p}.$

Proof. *see [3]*

Proposition I.1.1 (Poincaré's inequality)**.** *[2] Let Ω a bounded open then there is a constant$C > 0$ which depends only on, such as*

$$\|u\|_{L^p(\Omega)} \leq C\|\nabla u\|_{L^p(\Omega)} \quad \forall u \in W_0^{1,p}(\Omega) \quad (1 \leq p < \infty)$$

In particular, the expression $\|u\|_{L^p(\Omega)}$ is a standard on $u \in W_0^{1,p}(\Omega)$ which is equivalent to the. standard $\|u\|_W^{1,p}(\Omega)$

Theorem I.2 (Hölder's inequality)**.** *[4] Let $f \in L_p(\Omega)$and $g \in L_q(\Omega)$ with $1 \leq p < \infty$, Then $f, g \in L^1(\Omega)$ and*

$$\int_\Omega |(f(x)g(x))| \mathrm{d}x \leq \|f\|_p . \|g\|_q$$

Definition I.1.2. *The dual $V^{'}$ of a normed vector space V is the normed vector space of continuous linear forms on$V$. The dual space is equipped with the (operator) norm*

$$\|f\|_{V'} = \sup_{0\neq v\in V} \frac{|f(v)|}{\|v\|_V}$$

Theorem I.3. *[5] Let H be a Hilbert space and let $l \in H^{'}$ (dual of H). Then there is a unique $u \in H$such that*

$$l(v) = (u, v) \quad \forall v \in H$$

Moreover

$$\|l\|_{H'} = \|u\|_H$$

I.2 Auxiliary Functions and Theories in Universal Approximation Theorem

I.2.1 σ-algebra

Definition I.2.1. *An algebra of sets on X is a nonempty collection $\mathcal{A}$ of subsets of X that is closed under finite unions and complements.If $E_1, ..., E_n \in \mathcal{A}$, then $\cup_1^n E_j \in \mathcal{A}$; and if $E \in \mathcal{A}$, then $E^c \in \mathcal{A}$.*

Definition I.2.2. *[6] σ-algebra is an algebra that is closed under countable unions.*
Since $\cap_j E_j = (\cup_j E_j^c)^c$, algebras are also closed under finite intersections. Also, if $\mathcal{A}$ is an algebra,
then $\emptyset \in \mathcal{A}$ and$X \in \mathcal{A}$ since $\emptyset = E \cap E^c$ and $X = E \cup E^c$ for$E \in \mathcal{A}$.

Definition I.2.3. *[6] If X has a topology, then we define a Borel σ-algebra on X , as the σ-algebra generated by the family of open sets in X, which is denoted by $\mathcal{B}_X$.*

I.2.2 Measure

We follow Folland [7] that we want the range of our measure to be $[0, 1]$, and we just defined a family of sets algebra for the domain of the measure. Let's define measure.

Definition I.2.4. *Let X be a set that is able to generate a σ-algebra on$\mathcal{M}$ from it A measure on $\mathcal{M}(oron(X,\mathcal{M}))$ is a function $\mu : \mathcal{M} \to [0,1]$ such that*

1.$\mu(\emptyset) = 0$,

2.If $\{E_j\}_1^\infty$ is a sequence of disjoint sets in $\mathcal{M}$, then $\mu(\cup_1^\infty E_j) = \sum_1^\infty \mu(E_j)$.

Property (2) is called countable additivity and implies finite additivity: If $E_j\}_1^n$disjoint sets in $\mathcal{M}$, then $\mu(\cup_1^\infty E_j) = \sum_1^\infty \mu(E_j)$.

If X is a set and $\mathcal{M} \subseteq \mathcal{P}(X)$ is a σ-algebra, $(X,\mathcal{M})$ is called a measurable space and the sets in $\mathcal{M}$ are called measurable sets. If μ is a measure on $(X,\mathcal{M})$, then $(X,\mathcal{M},\mu)$ is called a measure space.

If $\mu(X) < \infty$, then μ is called finite.

If $X = \cup_1^\infty E_j$ where $E_j \in \mathcal{M}$ and $\mu(E_j) < \infty$ for all j, μ is called σ- finite.

Theorem I.4. *[6] Let $(X,\mathcal{M},\mu)$ be a measure space.*

1. (Monotonicity) If $E, F \in \mathcal{M}$, and $E \subseteq F$,then$\mu(E) \leq \mu(F)$

2. (Subadditivity) If $\{E_j\}_1^\infty \subseteq \mathcal{M}$, then$\mu(\cup_1^\infty E_j) \leq \sum_1^\infty \mu(E_j)$

3. (Continuity from below) If $\{E_j\}_1^\infty \subseteq \mathcal{M}$ and$E_1 \subseteq E_2 \subseteq ...$ then $\mu(\cup_1^\infty) = \lim_{j\to\infty} \mu(E_j)$.

4.(Continuity from above) If $\{E_j\}_1^\infty \supseteq \mathcal{M}$, $E_1 \subseteq E_2 \supseteq ...$ and$\mu(E_1) < \infty$ then $\mu(\cap_1^\infty E_j) = \lim_{j\to\infty} \mu(E_j)$

Definition I.2.5. *[6] If $(X,\mathcal{M},\mu)$ is a measure space, a set $E \in \mathcal{M}$ such that $\mu(E) = 0$ is called a null set. A measure whose domain includes all subsets of null sets is called complete.*

Outer Measure

Definition I.2.6. *An outer measure on a nonempty set X is a function$\mu^* : \mathcal{P}(X) \to [0;1]$ that satisfies:*

i $\mu^*(\emptyset) = 0$,

ii $\mu^*(A) \leq \mu^*(B)$ if$A \subseteq B$,

iii $\mu^*(\cup_1^\infty A_j) \leq \sum_1^\infty \mu^*(A_j)$.

The domain of the outer measure is easier to find since is just the power set of X. We now show that why it's also easy to fulfill the properties of a outer measure.

Proposition I.2.1. *Let $\varepsilon \subseteq \mathcal{P}(X)$ be arbitrary and $\rho : \varepsilon \to [0, \infty]$ be such that ;$\emptyset \in \varepsilon, X \in \varepsilon$, and $\rho(\emptyset) = 0$.For any $A \subseteq X$, define*

$\mu^*(A) = \inf\{\sum_1^\infty \rho(E_j) : E_j \in \varepsilon and A \subseteq \cup_1^\infty E_j\}$

Thenμ^* is an outer measure.

Proof. *see[6]*

Definition I.2.7. *If μ^* is an outer measure on X,a set $A \subseteq X$ is called μ^*- measurable if*

$$\mu^*(E) = \mu^*(E \cap A) + \mu^*(E \cap A^c), \forall E \subseteq X$$

Since $E \subseteq (E \cap A) \cup (E \cap A^c)$, by the definition of outer measure, $\forall E \subseteq X$,

$$\mu^*(E) \leq \mu^*((E \cap A) \cup (E \cap A^c)) \leq \mu^*(E \cap A) + \mu^*(E \cap A^c)$$

Thus, to show that A is μ^*-measurable, it suffices to show that the reverse inequality and if $\mu^*(E) = \infty$,then the reverse inequality works, so A is μ^*-measurable iff

$$\mu^*(E) \geq \mu^*(E \cap A) + \mu^*(E \cap A^c), \forall E \subseteq$$

$Xs.t\mu^*(E) \leq \infty$

Theorem I.5. *(Caratheordory's Theorem)If $\mu^*$is an outer measure on X, the collection$\mathcal{M}$ of μ^*-measurable sets is a σ-algebra, and the restriction of μ^* to $\mathcal{M}$ is a complete measure.*

Proof. *see[6]*

I.2.3 Measurable Functions

Definition I.2.8. *If $(X, \mathcal{M})$ and $(Y, \mathcal{N})$ are measurable spaces, a mapping $f : X \to Y$ is called $(\mathcal{M}, \mathcal{N})$-measurable, or just measurable, if $f^{-1}(E) \in \mathcal{M}, \forall E \in \mathcal{N}$*

Proposition I.2.2.

- *A function $f : X \to \mathbb{C}$ is $\mathcal{M}$-measurable iff Ref and Imf are $\mathcal{M}$-measurable.*
- *If $f, g : X \to \mathbb{C}$ are $\mathcal{M}$-measurable, then so are $f + g$ and fg.*
- *If $f, g : X \to \bar{\mathbb{R}}$ are measurable, then so are $\max(f, g)$ and $\min(f, g)$.*

Proof. *see[6]*

Definition I.2.9. *If $f, g : X \to \bar{\mathbb{R}}$ where $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$, we define the positive and negative parts of $f$to be*

$$f^+(x) = \max(f(x), 0),$$
$$f^-(x) = \max(-f(x), 0).$$

Then by this definition $f = f^+ + f^-$. If f is measurable, so are f^+ and f^-, by Proposition 2.3. We just defined the measurable function and now we try to discuss functions that are building blocksfor the theory of integration.

Definition I.2.10. *(Folland Section 2.1) Suppose that$(X, \mathcal{M})$ is a measurable space. If $E \subseteq X$, the charac- teristic function $\mathcal{X}_E$ of E (sometimes called indicator function of E denoted by 1_E) is defined by*

$$\mathcal{X}_E(x) \begin{cases} 1 & \text{if} x \in E \\ 0 & \text{if} x \notin E \end{cases}$$

Since the image of the characteristic function is $\{0, 1\}$, then $\mathcal{X}_E$ is measurable iff $E \in \mathcal{M}$.

Definition I.2.11. *(Folland Section 2.1) A simple function on X is a finite linear combination, with complex coeffcients, of characteristic functions of sets in $\mathcal{M}$. Equivalently, $f : X \to \mathbb{C}$ is simple iff f is measurableand the range of f is a finite subset of $\mathbb{C}$. The standard representation of f is:*

$$f = \sum_1^n z_j \mathcal{X}_{E_j}, \text{where} E_j = f^{-1}(\{z_j\}) \text{and range(f)} = \{z_1, ..., z_n\}.$$

This definition show that $f$is a linear combination of characteristic functions and the union of thesecharacteristic functions is X.

Theorem I.6. *Let $(X, \mathcal{M})$ be a measurable space.*

i If $f : X \to [0, \infty]$ is measurable, there is a sequence $\{\phi_n\}$ of simple functions such that $0 \leq |\phi_1| \leq |\phi_2| \leq ... \leq |f|, \phi_n \to f$ point-wise, and $\phi_n \to f$ uniformly on any set on which f is bounded.

ii If $f : X \to \mathbb{C}$ is measurable, there is a sequence $\{\phi_n\}$ of simple functions such that $0 \leq \phi_1 \leq \phi_2 \leq ... \leq f, \phi_n \to f$ point-wise, and $\phi_n \to f$ uniformly on any set on which f is bounded.

Proof. *see[6]*

I.2.4 Integration of nonnegative functions

In this section we fix a measure space $(X, \mathcal{M}, \mu)$, and we define

$$L^+ = \{f | f : X \to [0, \infty], fmeasurable.g\}$$

Definition I.2.12. *If ϕ is a simple function in L^+ with standard representation $\phi = \sum_1^n a_j \mathcal{X}_{E_j}$, we define the integral of ϕ with respect toμ by*

$$\int \phi \mathrm{d}\mu = \sum_1^n a_j \mu(E_j)$$

We note that $\int \phi \mathrm{d}\mu$ may equal to ∞ since $\mu(E_j)$ may be infinite. If $A \in \mathcal{M}$, then $\phi_{\mathcal{X}_A}$ is also simple by definition($\phi_{\mathcal{X}_A} = \sum a_j \mathcal{X}_{A \cap E_j}$), and we define $\int_A \phi \mathrm{d}\mu = \int_A \phi = \int_A \phi(x) \mathrm{d}\mu(x)$ and $\int = \int_X$.

Proposition I.2.3. *Let ϕ and ψ be simple functions in L^+*

1.if $c \geq 0, \int c\phi = c \int \phi$

2.$\int (\phi + \psi) = \int \phi + \int \psi$

3.if $\phi \leq \psi$, then $\int \phi \leq \int \psi$

4.The map $A \to \int_A \phi \mathrm{d}\mu$is measure on $\mathcal{M}$.

Proof. *see[6]*

Definition I.2.13.

$$\int f \mathrm{d}\mu = \sup \left\{ \int \phi \mathrm{d}\mu : 0 \leq \phi \leq f, \phi \textit{simple} \right\}$$

The above definition makes sense because the family of simple functions over which the supremum istaken includes f *itself. By the definition of* $\int f$ *and Proposition* $(1, 2, 3)$*, we have that*

$$\int f \leq \int g . \textit{whenever} f \leq g . \textit{and} \int cf = c \int f \textit{for all} c \in [0, 1)$$

Theorem I.7. *The Monotone Convergence Theorem. If* $\{f_n\}$ *is a sequence in* L^+ *such that* $f_j \leq f_{j+1}$ *for all* j *, and* $f = \lim_{n\to\infty} f_n$*, then* $\int f = \lim_{n\to\infty} \int f_n$

Proof. *see[6]*

Theorem I.8. *If* $\{f_n\}$ *is a finite or infinite sequence in* L^+ *and* $f = \sum_n f_n$ *, then* $\int f = \sum_n \int f_n$*.*

Proof. *see[6]*

CHAPTER II

AN INTRODUCTION TO DEEP LEARNING

II.1 Introduction

Deep Learning it's a branch of machine learning that relies on artificial neural networks inspired by the structure and functions of the human brain, and it's made a big fuss in the field of artificial intelligence since its emergence, especially in 2006It has many characteristics, one of the most important of which is

1.Deep learning has its roots in neural networks.

2.Deep learning is way of classifying, clustering, and predicting things by using a neural network that has been trained on vast amounts of data.

3.Deep learning creates many layers of neurons, attempting to learn structured representation of big data, layer by layer.

And what's given him all this attention is that it's a very important branch of machine learning, which in turn is a branch of artificial intelligence So what's machine learning, what's artificial

intelligence, what's their relationship to the artificial neural network?

Machine learning:It's a form of artificial intelligence, giving computers the ability to learn and improve through experiments The machine learning algorithm can learn how to provide predictions or solve problems when provided with sufficient data, as in Object recognition in pictures or winning certain games[1] The algorithm contains three basic concepts: Mission,. performance, experience.

Artificial intelligence:Techniques that have the ability to perform tasks that require human intelligence, such as speech recognition and visual perception,machine translation Current AI systems have the potential to adapt to new experiences..[1]

II.1.1 Historical Trends in Deep Learning

Deep learning has had a long and rich history, but has gone by many names reflecting different philosophical and Mathematical viewpoints, and has waxed and waned in popularity it's a brief reminder of the names of the creators and a statement of what you're talking about it was the beginning of human thinking that we could make a smart machine.

- In 1943, the first artificial neuron was invented by scientists and researchers "warren Mc cullochet and water pitts ". This was through a scientific article in 1943 [8]
- (the perceptron).And then in 1957, the world did Frank Resemblatt developpem the first dhistor daprisentage algorith
- 1974-1986:invention of multi-layered perception (antelegence artificiel)
- 1990.network ae neurones convelutututifs-lenet yann le cun
- 1997.neur-recurrent networks-LSTMs
- 2012.(Geeky Hinton)concours image net Envoldu deep learning

II.2 Neural Network

Neural networks are a model of machine learning inspired by the structure and activity of the human brain to create a computer program that learns from data that first emerged in the 1890s by psychology"W.James"It started in the 1940s by the worlds' W. Pitts ' and ' J. Mc Culloch' Neural networks are divided into two sections.

1.Biological Neural Networks

2.Artificial Neural Networks:It's divided into three sections.

- nput layer(It's the primary data of the neural network that transmits the information of the hidden layers machine.)
- Hidden layers(is the intermediate layer between the input layer And the output layer is where all the calculations are done.)
- Output layer(It's the class that gives results to certain inputs.)

II.2.1 perceptron Network

It's a set of experiments by the world(Frank Resemblatt) that he started by mimicking the human mind that led to his creation of the first neural network in history called Perceptron, which would later become the cornerstone of artificial intelligence, as shown in figure (2,1).

Remark II.2.1 (**He described the Perseptron network**). *This network consists of two basic layers The first layer is the input*$(X_1, X_2, ..., X_K)$ *And the second layer has neurons, or it's called the assembly function* $\sum$ *Links the two layers to weights* $(W_1, W_2, ..., W_K)$*With an estimated one-on-one bias over Percptron Which also links the two layers apply the activation function to the output end of the process So that in Perceptron this relationship applies to get the result*

$$y_j = \sum_{i=1}^{n} W_i.X_i \qquad (*)$$

Output results are as follows

$$\begin{cases} 0 & \textit{ify} \leq \textit{threshold} \\ 1 & \textit{ify} \leq \textit{threshold} \end{cases} \tag{II.1}$$

Set weights until we reach the lowest possible error rate

Corollary II.1. *All output from the network is either 0 or 1*

II.2.2 The functioning of the artificial neural network with different activation functions

An artificial neuron is a function f_j of the input [9]$x = x_1, x_2 \ldots, x_d$ weighted by a vector of connection weights $w_j = (w_{j,1} \ldots w_{j,d})$ completed by a bias b_j , and associated to an activation functionφ , namely

$$y_j = f_j(x) = \phi(\langle w_j, x \rangle + b_j) \tag{II.2}$$

Definition II.2.1 (**Activation functions**). *Activation functions determine whether the information (weight and bias) received by the nerve from the previous layer is sufficient to activate neurons or not and then send the output to the next layer of neurons as input.*

The activation function is chosen by shape, and there are several activation functions, of which we note the following.

- The identity function

$$\phi(x) = x \tag{II.3}$$

- The sigmoid function (or logistic)

$$\phi(x) = \frac{1}{1 + exp(-x)} \tag{II.4}$$

- The hyperbolic tangent function ("tanh")

$$\phi(x) = \frac{exp(x) - exp(-x)}{exp(x) + exp(-x)} = \frac{exp(2x) - 1}{exp(2x) + 1} \tag{II.5}$$

- The hard threshold function

$$\phi_\beta(x) = 1_{x \geq \beta} \tag{II.6}$$

- The Rectified Linear Unit (ReLU) activation function

$$\phi(x) = \max(o, x) \tag{II.7}$$

Here is a schematic representation of an artificial neuron where $\sum = \langle w_j, x \rangle + b_j$

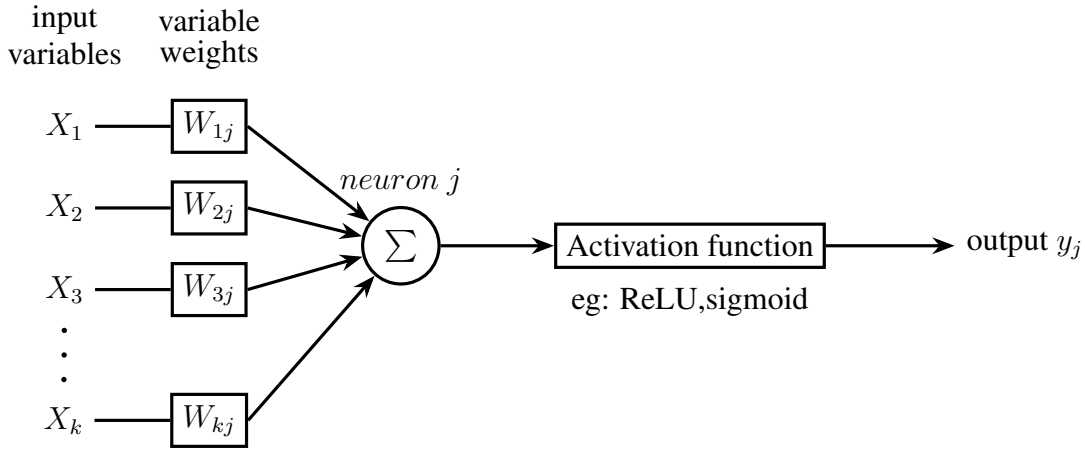


Figure II.1 – source: andrewjames turner.com.uk[9]

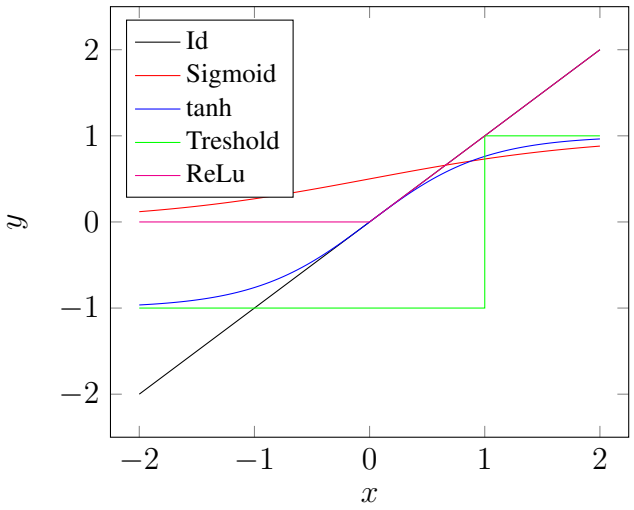


Figure II.2 – represents the activation function described above[9] .

II.2.3 Neural network learning algorithms

In supervised learning, network training is a repeat process of going back and forth in the network We compare the outcome and the real result, and this comparison is using the error function to measure the quality of the prediction in the network.Expresses the end process of b Forward propagation And about the process of returning b Back propagation.

- **Forward propagation**: The spread of initial input data In hidden web layers, after application Neurons calculate on these data we reach the final layer as a result of predicting .

- **Back propagation** It's a method of training artificial neural networks based on weights control based on error ratio.

- **Loss function** It's an important factor in determining how close the neural network is to the ideal weight. It helps us calculate the difference between the real results and the results obtained, which we express in loss, and which we aim to reduce. There are several ways to measure the Loss function, which is

$$MSE = \frac{1}{n}\sum_{j=1}^{n}(y_i - \hat{y_i})^2 \qquad \text{(Mean Square Error)} \tag{II.8}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_i - \hat{y_i})^2} \qquad \text{(Root Mean Square Error)} \tag{II.9}$$

$$BCE = -\left(y * \log(\hat{y}) + (1-y) * \log(1-\hat{y})\right) \text{(Binary cross entropy)} \tag{II.10}$$

n:The number of input data

y_i:real results,i:number of categories

$\hat{y_i}$:Results obtained after prediction

Corollary II.2. *Machine learning requires human intervention to complete tasks, while deep learning requires no human intervention to complete tasks.*

II.3 Universal Approximation Theorem of Neural Networks

-Discriminatory Functions

Definition II.3.1. *[10] We say that a function σ is discriminatory if given a measure $\mu \in M(I_n)$ such that*

$$\int_{I_n} \sigma(y^T X + b)\mu(x) = 0, y \in \mathbb{R}^n, b \in \mathbb{R}$$

implies that$\mu = 0$

Remark II.3.1.

- ***functions are NOT discriminatory*** *Let $\sigma = \frac{1}{2}$, and μ be a measure with density$g = -\mathcal{X}_{(0,\frac{1}{2}]} + \mathcal{X}_{[\frac{1}{2},1)}$*
  Clearly$\mu \neq 0$,but$\int_{[0,1]} \sigma(y^T X + b)\mu(x) = 0, \forall b, y$

- ***Functions are Discriminatory*** *Any bounded, measurable, sigmoidal function is discriminatory. In particular, any continuous sigmoidal function is discriminatory.*

Theorem II.1. *[10] Let σbe any continuous discriminatory function Then finite sums of the form*

$$G(x) = \sum_{j=1}^{N} \alpha_j \sigma(y_j^T x + b_j), where y_j \in \mathbb{R}^n, b, \alpha_j \in \mathbb{R}$$

are dense in $C(I_n)$.

In other words,given any $\varepsilon > 0$ and $f \in C(I_n)$,there is a sum $G(x)$ of the above form such that

$$|G(x) - f(x)| < \varepsilon \quad , x \in I_n$$

-Proof of Universal Approximation Theorem[10]

Most Important Tools for Proof

Theorem II.2 (Hahn-Banach)**.** *[10]*

Let V be a normed vector space,$R \subset V$ a subspace of VLet $L \in R^*$,Then there exists$\hat{L} \in V_*$ that extends L to V and satisfies$\|\hat{L}\|_{V_*} = \|L\|_{R^*}$

Corollary II.3. *Let V be a normed vector space, $R \subset V$ a subspace of V*

Let $x_0 \in V$ such that $d(x_0, R) = \gamma > 0$.

Then there exists$L \in V_*$ such that

- $\|L\|_{V_*} = 1$
- $L(x_0) = \gamma$
- $L(R) = 0$

Theorem II.3 (Riesz Representation Theorem)**.** *[10]*

Let L be a bounded linear functional on $C(I_n)$.

Then there exists a unique $\mu \in M(I_n)$ such that

$$L(h) = \int_{I_n} h(x) \mathrm{d}\mu(x)$$

Proof. *Goal:Let $S \subset C(I_n)$ be the set of functions of the form*

$$G(x) = \sum_{j=1}^{N} \alpha_j \sigma(y_j^T x + b_j)$$

We want to to prove that $R := \bar{S} = C(I_n)$.

-S is a linear subspace of $C(I_n)$.

-By contradiction suppose suppose $R \subsetneq C(I_n$, that is $\exists f \in C(I_n)$such that $d(f, R) > 0$.

-By the Corollary to H-B $\exists L$ bounded linear functional on$C(I_n)$ such that $L \neq 0$, but $L(S) = L(R) = 0$

-By RRT $\exists! \mu \in M(I_n)$ such that

$$L(h) = \int_{I_n} h(x) \mathrm{d}\mu(x), \forall h \in C(I_n)$$

-Since$L(R) = 0$and since $\sigma(y^T x + b) \in R, \quad \forall y, b$ then

$$0 = L(\sigma(y^T x + b)) = \int_{I_n} \sigma(y^T x + b) \mathrm{d}\mu(x), \quad \forall y, b \tag{II.11}$$

Since σ is discriminatory, $(1, 1)$ implies$\mu = 0$, which in turn implies $L = 0$, and this is a contradiction.

II.4 Approximation of Classification Functions

Theorem II.4. *[10] Let σ be a continuous sigmoidal function. Let f be a classification function for any finite measurable partition of I_n.*

Then for any $\varepsilon > 0$ there exists a finite sum of the form

$$G(x) = \sum_{j=1}^{N} \alpha_j \sigma(y_j^T x + b_j) \tag{II.12}$$

and a set $D \subset I_n$with $m(D) \geq 1\varepsilon$ such that

$$|G(x) - f(x)| < \varepsilon \quad , \forall x \in D$$

Most Important Tools for Proof

Theorem II.5 (Lusin)**.** *Let $f : I_n \to \mathbb{R}$ be measurable.*

Then for any$\varepsilon > 0$there exists a set$D \subset I_n$ with $m(D) \geq 1 - \varepsilon$ and $h \in C(I_n)$ such that

$$h(x) = f(x), \quad , \forall x \in D$$

Proof. *-By Lusin's theorem$\exists D \subset I_n$ with $m(D) \geq 1 - \varepsilon$ and $h \in C(I_n)$such that$h(x) = f(x), \forall x \in D$.*

-Since $h \in C(I_n)$ by the Universal Approximation Theorem,there exists a network G of the form $(1, 2)$ such that

$$|G(x) - h(x)| \leq \varepsilon \quad x \in I_n$$

-Then for all $x \in D$ we have

$$|G(x) - f(x)| = |G(x) - h(x)| < \varepsilon$$

II.5 Conclusion

The intersection between deep learning, machine learning and artificial intelligence is a techno-wealth that has given way to technological development. Although deep learning is

a branch of machine learning, a cycle can go beyond that, and this is evident through the inventions of both

CHAPTER III

THE DEEP GALERKIN METHOD

III.1 Offering of the Deep Galerkin Method

III.1.1 Mathematical Details

The form of the PDEs of interest are generally described as follows:

-an unknown function of time and space defined on the region $[0;T] \times \Omega$ where $\Omega \subset \mathbb{R}^d$

-assume that u satisfies a parabolic PDE of the following form:[11]

$$\begin{cases} (\partial_t + \ell)u(t,x) & = 0, \quad (t,x) \in [0,T] \times \Omega \\ u(0,x) & = u_0(x) \quad , x \in \Omega \quad \text{(initial condition)} \\ u(t,x) & = g(t,x) \quad (t,x) \in [0,T] \times \partial\Omega, \quad \text{(boundary condition)} \end{cases}$$

The goal is to approximate u with an approximating function $f(t;x;\theta)$ given by a deep neural network with parameter set θ.

Remark III.1.1. *The loss functional for the associated training problem consists of three parts:*

*1) A measure of how satisfies the **differential operator**:*

$$\|(\partial_t + \ell)f(t;x;\theta)\|^2_{[0,T]\times\Omega,\nu_1}$$

*2) A measure of how satisfies the **boundary condition**:*

$$\|f(t;x;\theta) - g(t,x)\|^2_{[0,T]\times\partial\Omega,\nu_2}$$

*3) A measure of how satisfies the **initial condition**:*

$$\|f(0;x;\theta) - u_0(x)\|^2_{\Omega,\nu_3}$$

Remark III.1.2. *:*

parameterizing f as a neural network means that the differential operator can be conputed easily using backpropagation.

Remark III.1.3. *:[11]*

this measure can be modified for problems with a terminal condition or extended for problems with both initial and terminal conditions.

In all three terms above the error is measured in terms of L^2-norm, i.e. using $\|h(y)\|_{\mathcal{Y},\nu} = \int_{\mathcal{Y}} |h(y)|^2\nu(Y)\mathrm{d}y$ with $\upsilon(y)$ being a density defined on the region $\mathcal{Y}$. Combining the three terms above gives us the cost functional associated with training the neural network:

$$L(\theta) = \|(\partial_t + \ell)f(t;x;\theta)\|^2_{[0,T]\times\Omega,\nu_1} + \|f(t;x;\theta) - g(t,x)\|^2_{[0,T]\times\partial\Omega,\nu_2} + \|f(0;x;\theta) - u_0(x)\|^2_{\Omega\nu_3}$$

so that

$\|(\partial_t + \ell)f(t;x;\theta)\|^2_{[0,T]\times\Omega,\nu_1}$ is a (differential operator)

$\|f(t;x;\theta) - g(t,x)\|^2_{[0,T]\times\partial\Omega,\nu_2}$ is a (boundary condition)

$\|f(0;x;\theta) - u_0(x)\|^2_{\Omega,\nu_3}$ is a (initial condition)

The next step is to minimize the loss functional using stochastic gradient descent. More specifically, we apply the algorithm defined in Algorithm of the Deep Galerkin Method The description given in Algorithm of the Deep Galerkin Method should be thought of as a general outline

and the algorithm should be modified according to the particular nature of the PDE being considered.

III.2 Deep Galerkin Method (DGM) algorithm.

[12] 1. Initialize the parameter set θ_0 and the learning rate α_n.

2. Generate random samples from the domain's interior and time/spatial boundaries, i.e.

- Generate $(t_n; x_n)$ from $[0; T] \times \Omega$ according to ν_1

- Generate $(\mathcal{T}_n; z_n)$ from $[0; T] \times \partial\Omega$according to ν_2

- Generate w_n from Ω , according to ν_3

3. Calculate the loss functional for the current mini-batch, i.e. the randomly sampled points$s_n = \{(t_n; x_n), \mathcal{T}_n; z_n), w_n\}$:

- Compute $L_1(\theta_n; t_n; x_n) = ((\partial_t + \ell)f(t_n, x_n, \theta_n))^2$

- Compute $L_2(\theta_n; \mathcal{T}_n; z_n) = (f(\mathcal{T}_n, z_n, \theta_n) - g(\mathcal{T}_n, z_n))^2$

- Compute $L_3(\theta_n; w_n) = (f(0, w_n, \theta_n) - u_0(w_n))^2$

-Compute $L(\theta_n; s_n) = L_1(\theta_n; t_n; x_n) + L_2(\theta_n; \mathcal{T}_n; z_n) + L_3(\theta_n; w_n)$

4. Take a descent step at the random point s_n with Adam-based learning rates:

$$\theta_{n+1} = \theta_n - \alpha_n \nabla_\theta L(\theta_n; s_n)$$

5. Repeat steps $(2) - (4)$ until $\|\theta_{n+1} - \theta_n\|$ is small.

Corollary III.1. *It is important to notice that the problem described here is strictly an optimisation problem This is unlike typical machine learning applications where we are concerned with issues of underfitting, overfitting and generalization. The only case where generalization becomes relevant is when we are unable to sample points everywhere within the region where the function is defined, e.*

III.3 Implementation Details for DGM algorithm

The architecture adopted by Sirignano and Spiliopoulos is similar to that of LSTMs and

Highway Networks described in the previous chapter. It consists of three layers, which we refer to as DGM layers: an input layer, a hidden layer and an output layer, though this can be easily extended to allow for additional hidden layers .Figure $(3, 1)$ for a visualization of the overall architecture

Corollary III.2. *The DGM consists of three layers, one input layer,one exit layer and one hidden layer.*

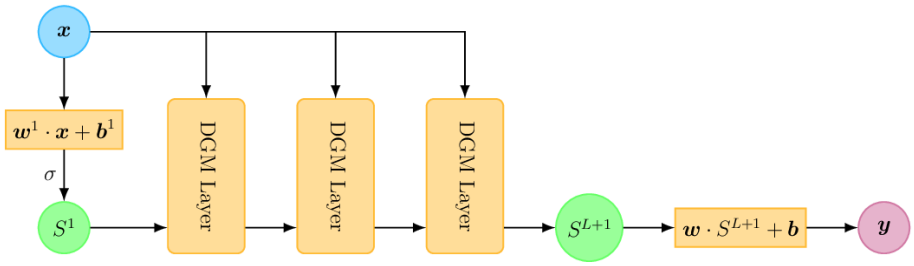


Figure III.1 – Bird's-eye perspective of overall DGM architecture.[12]

The PDE solution requires a model $f(t, x, \theta)$ which can make sharp turns" due to the final condition,which is of the form $u(T, x) = \max(p(x), 0)$ (the first derivative is discontinuous when $p(x) = 0$. The shapeof the solution $u(t, x)$ for $t < T$, although smoothed" by the diffusion term in the PDE, will still have a nonlinear profile which is rapidly changing in certain spatial

regions. In particular, we found the following network architecture to be effective:

$$
\begin{aligned}
S^1 &= \sigma(W^1.\vec{x} + b^1) \\
Z^\ell &= \sigma(u^{z,\ell}.\vec{x} + w^{z,\ell}.S^\ell + b^{z,\ell}) \quad \ell = 1 \dots L \\
G^\ell &= \sigma(u^{g,\ell}.\vec{x} + w^{g,\ell}.S^\ell + b^{g,\ell}) \quad \ell = 1 \dots L \\
R^\ell &= \sigma(u^{r,\ell}.\vec{x} + w^{r,\ell}.S^\ell + b^{r,\ell}) \quad \ell = 1 \dots L \\
H^\ell &= \sigma(u^{h,\ell}.\vec{x} + w^{h,\ell}.(S^\ell \odot R^\ell) + b^{h,\ell}) \quad \ell = 1 \dots L \\
S^{\ell+1} &= (1 - G^\ell) \odot H^\ell + Z^\ell \odot S^\ell \quad \ell = 1 \dots L \\
f(t; x; \theta) &= w.S^{L+1} + b
\end{aligned}
$$

where$\vec{x} = (t, x)$, the number of hidden layers is L+1, and denotes element-wise multiplication$(i.e., z\odot v = z_0 v_0, ..., z_N v_N)$. The parameters are

$$
\theta = \left\{ W^1, b^1, \left(u^{z,\ell}, w^{z,\ell}, b^{z,\ell}\right)_{\ell=1}^{L}, \left(u^{g,\ell}, w^{g,\ell}, b^{g,\ell}\right)_{\ell=1}^{L}, \left(u^{r,\ell}, w^{r,\ell}, b^{r,\ell}\right)_{\ell=1}^{L}, \left(u^{h,\ell}, w^{h,\ell}, b^{h,\ell}\right)_{\ell=1}^{L}, W, b \right\} \tag{III.1}
$$

The number of units in each layer is M and $\sigma : \mathbb{R}^M \to \mathbb{R}^M$is an element-wise nonlinearity:

$$
\sigma(z) = (\phi(z_1), \phi(z_2), ..., \phi(z_M)\} \tag{III.2}
$$

where $\sigma : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function such as the tanh function, sigmoidal function $\dfrac{e^y}{1 + e^y}$, or rectified linear unit $(ReLU)$ $\max(y, 0)$.

The architecture (H) is relatively complicated. Within each layer, there are actually many "-layers" of computations.

The key hyperparameters in the neural network (H) are the number of layers L, the number of units M in each sub-layer, and the choice of activation unit $\phi(y)$.

III.4 A Neural Network Approximation Theorem

Theoretical motivation for using neural networks to approximate solutions to PDEs[11] is given as an elegant result in **Sirignano** and **Spiliopoulos** (2018) which is similar in spirit to the Uni-

versal Approximation Theorem. More specifically, it is shown that deep neural network approximators converge to the solution of a class of quasilinear parabolic PDEs as the number of hidden layers tends to infinity. To state the result in more precise mathematical terms, define the following:

1)$L(\theta)$, the loss functional measuring the neural network's fit to the differential operator and boundary/initial/terminal conditions;

2)c^n, the class of neural networks with n hidden units;

3) $f^n = arg\min_{f \in c^n}$, the best n-layer neural network approximation to the PDE solution.

The main result is the convergence of the neural network approximators to the true PDE solution:

$$f^n \to u \quad \text{as} \quad n \to \infty$$

Remark III.4.1. *Further details, conditions, statement of the theorem and proofs are found in Section 7 of* ***Sirignano and Spiliopoulos (2018)****[15]. It is should be noted that, similar to the Universal Approximation Theorem, this result does not prescribe a way of designing or estimating the neural network successfully.*

III.5 Convergence of the neural network to the PDE solution

We now prove, under stronger conditions, the convergence of the neural networks f^n to the solution u of the PDE[11]

$$\begin{cases} \partial_t u(t,x) - div\left(\alpha(t,x,u(t,x),\nabla u(t,x))\right) + \gamma\left(t,x,u(t,x),\nabla u(t,x)\right) = 0 \text{for} (t,x) \in \Omega_T \\ u(0,x) = u_0(x) \text{for} x \in \Omega \\ u(t,x) = 0 \text{for} (t,x) \in \partial\Omega_T \end{cases} \tag{B}$$

as $n \to \infty$. Notice that we have restricted the discussion to homogeneous boundary data. We do this for both presentation and mathematical reasons

The objective function is

$$J(f) = \|\mathcal{G}[f]\|^2_{2,\Omega_T} + \|f\|^2_{2,\Omega_T} + \|f(0,.) - u_0\|^2_{2,\Omega_T}$$

Recall that the norms above are$L^2(X)$ norms in the respective space $X = \Omega_T, \partial\Omega_T$ and Ω respectively.From Neural Network Approximation Theorem , we have that

$$J(f^n) \to 0\text{as}n \to \infty$$

Each neural network f^n satisfies the PDE

$$\mathcal{G}[f^n](T,X) = h^n(t,x)\text{for}(t,x) \in \Omega_T$$
$$f^n(o,x) = u_0^n(x)\text{for}x \in \Omega$$
$$f^n(t,x) = g^n(t,x)\text{for}(t,x) \in \partial\Omega_T$$

for some f^n, u_0^n , and g^n such that

$$\|f^n\|^2_{2,\Omega_T} + \|g^n\|^2_{2,\Omega_T} + \|u_0^n - u_0\|^2_{2,\Omega_T} \tag{III.3}$$

For the purposes of this section, we make the following set of assumptions.

Condition

1. There is a constant $\mu > 0$ and positive functions $k(t,x); \lambda(t.x)$ such that for all $(t,x) \in \Omega_T$ we have

$$\|\alpha(t,x,u,p)\| \leq \mu(k(t,x) + \|p\|), \text{end}|\gamma(t,x,u,p)| \leq \lambda(t,x)\|p\|)$$

with $k \in L^2(\Omega_T)$,$\lambda \in L^{d+2+\eta}(\Omega_T)$ for some $\eta > 0$

2.$\alpha(t,x,u,p)$ and $\gamma(t,x,u,p)$are Lipschitz continuous in $(t,x,u,p) \in \Omega_T \times \mathbb{R} \times \mathbb{R}^D$ uniformly on compacts of the form $\{(t,x) \in \bar{\Omega}_T|u| \leq C, |p| \leq C\}$

3.defferentiable with respect to (x,u,p) with continuous derivatives.

4.There is a positive constant $\nu > 0$ such that

$$\alpha(t,x,u,p) \geq \nu|p|^2$$

and

$$\langle \alpha(t,x,u,p_1) - \alpha(t,x,u,p_2), p_1 - p_2 \rangle > 0 \text{for every} p_1, p_2 \in \mathbb{R}^d, p_1 \neq p_2$$

5.$u_0(x) \in \mathcal{C}^{0,2,\xi}(\bar{\Omega})$ for some $\xi > 0$ with itself and its first derivative bounded in $\bar{\Omega}$.

6.Ω is a bounded, open subset of $\mathbb{R}^d$ with boundary $\partial\Omega \in \mathcal{C}^2$.

For every $n \in \mathbb{N}$, $f^n \in \mathcal{C}^{1,2}(\bar{\Omega}_T)$. In addition, $(f^n)_{n\in\mathbb{N}} \in L^2(\Omega_T)$.

Theorem III.1. *[14] Assume that Condition B and (3.3) hold. Then, problem (B) has a unique bounded solution in $\mathcal{C}^{0,\delta,\frac{\delta}{2}}(\bar{\Omega}_T) \cap L^2\left(0,T,W_0^{1,2}(\Omega)\right) \cap W_0^{(1,2),2}(\Omega_T')$ for some $\delta > 0$ and any interior sub domain Ω_T' ofΩ_T. In addition, f^n converges to u, the unique solution to(B), strongly in $[13]L^\rho(\Omega_T)$for every $\rho < 2$. If,in addition, the sequence $\{f^n(t,x)\}_{n\in\mathbb{N}}$ is uniformly bounded in n and equicontinuous then the convergence to u is uniform in Ω_T .*

Proof. *see[14]*

III.6 Conclusion

In short, the different stages of the deep Galerkin method can be summarized according to the following scheme.

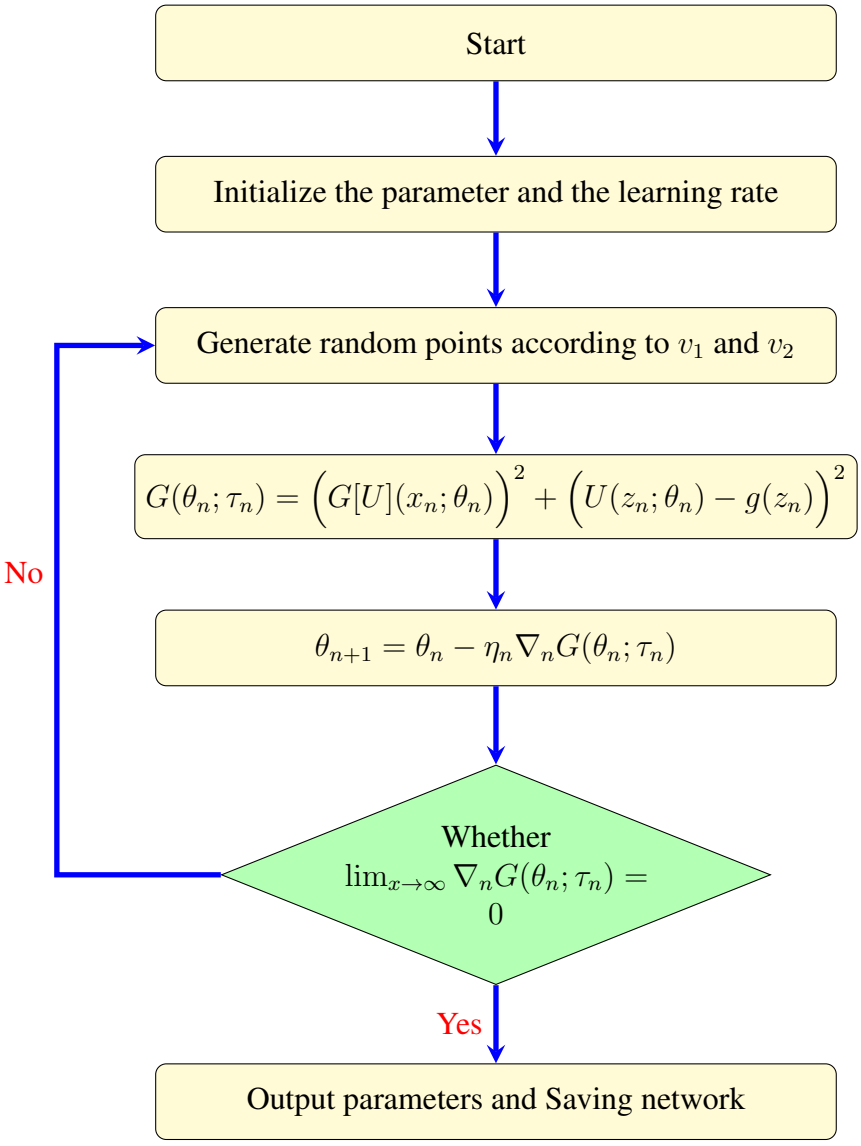


Figure III.2 – Flowchart of the DGM.

CONCLUSION GENERAL

Finally, the various partial diferential equations, PDEs, and their methods of solving and analyzing them, have been an important part of the interest of many mathematical researchers and even physicists until they have given them their right to research. One of the most important methods of Methods for solving partial differential equations that has been very well received by researchers is DGM, around which this study is centered, so that we elaborate on them in chapter III.We extend the Deep Galerkin Method (DGM) introduced in Sirignano and Spiliopoulos (2018) to solve a number of partial diferential equations (PDEs) that arise in the context of optimal stochastic control and mean field games.

In contrast, the main idea behind solving PDEs using the Deep Galerkin Method (DGM) described in the work of Sirignano and Spiliopoulos (2018) is to represent the unknown function of interest using a deep neural network. Noting that the function must satisfy a known PDE, the network Through this research, we have come up with some of the most important findings and points.

- The deep Galerkin method is a way to solve the different partial equations.
- Deep Galerkin Method (DGM) introduced in Sirignano and Spiliopoulos (2018) to solve

a number of partial differential equations (PDEs) that arise in the context of optimal stochastic control and mean field games.

- could Convergence of the neural network to the PDE solution of the Deep Galerkin Method This is to be $L(\theta) = 0$
- extend the DGM algorithm to solve for the value function and the optimal control simultaneously by characterizing both as deep neural networks. Training the
- And we didn't get to Galerkin Methods for Variational Inequalities

BIBLIOGRAPHY

[1] Jitao Gu Zhengdong LU,Hang Li and victor okli incorporating copying mechnism in sequence to sequence learning axive priprint zxive 160306393/2016

[2] LECTURE NOTES: THE GALERKIN METHOD arXiv:1112.1176v1 [math.AP] 6/ Dec /2011

[3] Adams, R.A. Sobolev spaces. Academic Press, New York. 1975

[4] H. Brézis, Analyse Fonctionnelle Et Applications, Dunod, /1999.

[5] M. Sofonea, A. Matei, Variational Inequalities with Applications. Advances in Mechanics and Mathematics Volume 18

[6] Ji, Zongliang, "Approximation of Continuous Functions by Artificial Neural Networks" (2019). Honors Theses. 2306. https://digitalworks.union.edu/theses/2306 June, /2018

[7] Gerald B Folland. Real analysis: modern techniques and their applications. John Wiley Sons, /2013.

[8] Deep Learning with Python FRANÇOIS CHOLLET Printed in the United States of America orders@manning.com

[9] Neural Networks and Introduction to Deep Learning wiki stat page 1

[10] Universal Approximation Theorem, G. Cybenko,Elisa Negrini,Worcester Polytechnic Institute,October 29 /2019

[11] DGM: A deep learning algorithm for solving partial differential equations. arXiv preprint arXiv:1708.07469 Sirignano, J., Spiliopoulos, K., /2018.

[12] Solving Nonlinear and High-Dimensional Partial Differential Equations via Deep Learning,EMAp, Fundac¸ ~ao Getulio Vargas, Rio de Janeiro, Brazil.

[13] Cybenko, G., Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems 2 (4), 303–314./ 1989.

[14] Kingma, D. P., Ba, J., . ADAM: A method for stochastic optimization. arXiv preprint arXiv:1412.6980:/2014

[15] Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y., Deep Learning. Vol. 1. MIT press Cambridge/ 2016.