학사졸업논문

Reconstruction of Coronal Magnetic Fields Using Physics-Informed Neural Networks

지도교수 문용재

경희대학교 응용과학대학
우주과학과

전민규

2023년 6월

Reconstruction of Coronal Magnetic Fields Using Physics-Informed Neural Networks

전민규
mgjeon@khu.ac.kr

Abstract: This thesis presents a novel approach to calculate nonlinear force-free fields using physics-informed neural networks. The coronal magnetic fields, which are related to solar activities like coronal mass ejections and solar flares, can be approximated as nonlinear force-free fields. In order to calculate nonlinear force-free fields, the partial differential equation called the force-free equation has to be solved. Recently, physics-informed neural networks have emerged as a new method to solve partial differential equations using a computer. This study demonstrates that physics-informed neural networks can be used for calculating nonlinear force-free fields given by the Low-Lou model which is one of the semi-analytical solutions.

1 Introduction

The solar activities, such as flares and coronal mass ejections, can influence the Earth. Understanding these solar phenomena requires accurate knowledge of the coronal magnetic fields, as they play a crucial role in governing such activities. Stokes polarimetry, based on the Zeeman effect, has proven to be an effective method for observing solar magnetic fields (Stenflo 2013). The Helioseismic and Magnetic Imager (HMI) aboard the Solar Dynamics Observatory (SDO) routinely provides high-resolution photospheric vector magnetograms using polarimetry techniques (Schou et al. 2012). However, the measurement of coronal magnetic fields remains challenging due to the difficulty in detecting the Zeeman effect at the extreme temperatures of the solar corona, approximately 10^6 K (Cargill 2009). Consequently, in most cases, only photospheric vector magnetic fields are available, while the estimation of coronal magnetic fields requires the solution of partial differential equations (PDEs) that describe the physical conditions in the solar corona.

One widely adopted assumption for describing the coronal magnetic field is the force-free assumption, which assumes that the Lorentz force dominates over other forces in the solar corona. This assumption is based on the low plasma beta (β) in the coronal plasma, which compares the plasma pressure gradient force to the magnetic pressure gradient force (Gary 2001). The force-free assumption leads to a fundamental equation that the static coronal magnetic field must satisfy, known as the force-free equation: $(\nabla \times \mathbf{B}) \times \mathbf{B} = 0$. Furthermore, since it is a magnetic field, an additional equation, known as Gauss's law for the magnetic field or the solenoidal condition ($\nabla \cdot \mathbf{B} = 0$), must also be satisfied. Magnetic fields that fulfill the force-free equation are referred to as force-free fields. The force-free equation can also be expressed as $\nabla \times \mathbf{B} = \alpha \mathbf{B}$, where α is called the force-free parameter or force-free function. A constant α throughout the computational domain corresponds to linear force-free fields (LFFFs), while a varying α gives rise to nonlinear force-free fields (NLFFFs). Linear force-free fields with $\alpha = 0$ correspond to potential fields. Due to the spatial variation of the force-free parameter in the solar corona, linear force-free fields are considered incomplete representations of the coronal magnetic field.

The optimization method, one of the traditional numerical methods, has been widely used to reconstruct coronal magnetic fields by solving the force-free equation. This method, originally introduced by (Wheatland et al. 2000), involves minimizing a functional L in eq. (2.3). This functional combines the force-free equation and the solenoidal condition. Recently, there has been a growing interest in an emerging technique called physics-informed neural networks (PINNs) for solving partial differential equations (Karniadakis et al. 2021). Physics-informed neural networks use the loss function derived from the partial differential equations and aim to minimize it to obtain the numerical solution. Physics-informed neural networks offer several advantages, such as their capability to handle complex functions, flexibility, and the ability to compute derivatives using automatic differentiation.

The objective of this study is to investigate the performance and capabilities of the physics-informed neural network method in calculating

coronal magnetic fields. I utilize the same loss function L employed in the traditional optimization method, with a specific focus on its application to the Low-Lou model. Through this investigation, I seek to evaluate the potential of the physics-informed neural network method as a novel computational tool for studying solar magnetic fields.

2 Theory

2.1 Nonlinear force-free fields

This section explains the force-free assumption, a popular model for the coronal magnetic field. The most materials in the solar corona is in a fully ionized plasma state due to the high temperature of the corona. The plasma beta β of the coronal plasma is quite low, as discussed in (Gary 2001). This means that the plasma pressure gradient force is negligible compared to the magnetic pressure gradient force, which is a part of the Lorentz force in magnetohydrodynamics (MHD). Additionally, the gravitational force exerted on the coronal plasma is small compared to the Lorentz force due to the low mass density of the plasma. As a result, the Lorentz force dominates other forces in the solar corona. Therefore, the momentum equation in the MHD equations for the solar coronal plasma in the Gaussian unit system can be written as

$$\rho \frac{d\boldsymbol{v}}{dt} = \frac{1}{c}\mathbf{J} \times \mathbf{B},$$

where ρ is the mass density of the plasma element, $\boldsymbol{v}$ the center of mass velocity, $\mathbf{J}$ the current density, and $\mathbf{B}$ the magnetic field.

In most regions of the solar corona, the ideal MHD condition is valid because of the high electrical conductivity of the coronal plasma. This means that the coronal plasma elements are *frozen-in* to the coronal magnetic fields. For the static coronal magnetic fields, the plasma elements must also be static, i.e., have a velocity of zero, meaning that the net force exerted on the plasma elements is zero, which can be expressed as

$$\mathbf{J} \times \mathbf{B} = \mathbf{0}.$$

Since the displacement current term in Ampère-Maxwell equation is ignored in MHD, the current density in MHD is given by $\mathbf{J} = \frac{c}{4\pi}\nabla \times \mathbf{B}$. The static coronal magnetic fields in this force-free assumption should satisfy the following partial differential equations:

$$(\nabla \times \mathbf{B}) \times \mathbf{B} = 0, \tag{2.1}$$

$$\nabla \cdot \mathbf{B} = 0. \tag{2.2}$$

Equation (2.1) is referred to as the force-free equation and eq. (2.2) is known as the Gauss's law for the magnetic field in Maxwell's equations which is often called the solenoidal condition. The magnetic fields that satisfy the force-free equation are called force-free fields. Equation (2.1) can be rewritten as

$$\nabla \times \mathbf{B} = \alpha \mathbf{B},$$

where α is called the force-free parameter or force-free function. If α is constant everywhere in the computational domain, the magnetic field is called a linear force-free field, otherwise it is called a nonlinear force-free field. A linear force-free field with $\alpha = 0$ is called a potential field. The linear force-free field and its special case potential field are considered incomplete models for the coronal magnetic field because the force-free parameter α is not constant everywhere in the solar corona, even in the small region above a single active region. Therefore, it is crucial to calculate nonlinear force-free fields by solving the force-free equation to accurately describe the coronal magnetic field.

Both the domain V and boundary condition (and/or initial condition) at the boundary ∂V must be specified to solve differential equations. In the case of the force-free field, there is no initial condition to consider because it is a static magnetic field. In order to reconstruct coronal magnetic fields, the computational domain V is taken to be the region above the photosphere. Although the force-free assumption is not valid in the photosphere, the photospheric magnetic field vector is often used as the one of the boundary conditions because it is the only routinely available high-resolution magnetic field vector. In this thesis, the coronal magnetic fields above a single active region are considered. To accomplish this, an artificial photospheric magnetic field vector is generated using the Low-Lou model. Since the typical area of a solar active region is very small compared to the whole solar surface area, the active region can be assumed to be flat. Thus, the computational domain V for the calculation of nonlinear force-free fields above a single active region becomes a rectangular domain whose bottom boundary coincides with the active region.

One popular traditional numerical method for calculating nonlinear force-free fields is the optimization method, which was first proposed by (Wheatland et al. 2000). The main strategy of the optimization method is to minimize the functional L defined as

$$L = \int_V \left[\frac{|(\nabla \times \mathbf{B}) \times \mathbf{B}|^2}{B^2} + |\nabla \cdot \mathbf{B}|^2 \right] \mathrm{d}V. \tag{2.3}$$

Since each term of the integrand is always positive, if L is zero for a magnetic field $\mathbf{B}$ in V, the magnetic field $\mathbf{B}$ must satisfy both the force-free equation and solenoidal condition everywhere in V, i.e., the field is a nonlinear force-free field. Differentiating eq. (2.3) with respect to t which is a

time-like parameter increasing with the interation steps leads to

$$\frac{1}{2}\frac{dL}{dt} = -\int_V \frac{\partial \mathbf{B}}{\partial t}\cdot\tilde{\mathbf{F}}\mathrm{d}V - \oint_{S=\partial V}\frac{\partial \mathbf{B}}{\partial t}\cdot\tilde{\mathbf{G}}\mathrm{d}S,$$

where $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{G}}$ are certian vector fields given by the magnetic field $\mathbf{B}$. If $\mathbf{B}$ is updated accroding to

$$\frac{\partial \mathbf{B}}{\partial t} = \mu\tilde{\mathbf{F}} \quad \text{in } V,$$

where $\mu > 0$ and the boundary condition is fixed at the boundary $S = \partial V$, i.e.,

$$\frac{\partial \mathbf{B}}{\partial t} = 0 \quad \text{at } S = \partial V,$$

then the functional L monotonically decreases because

$$\frac{dL}{dt} = -2\int_V \mu|\tilde{\mathbf{F}}|^2\mathrm{d}V < 0.$$

The optimiztion method requires the initial magnetic field $\mathbf{B}$ in the computational domain V and at its boundary ∂V. Since the magnetic field observation data is only avaiable at the bottom boundary (photosphere), the boundary condition at top and lateral boundaries is given by the potential magnetic field which can be reconstructed only using normal components at the bottom boundary condition. For convenience, the initial magnetic field for the optimization method is the potential field reconstructed using magnetic field data at the bottom boundary. In contrast, the physics-informed neural network method does not use the potential magnetic field as the initial magnetic field in the domain although the top and lateral boundary conditions are also given by the potential field. Instead, a randomly generated vector field serves as the initial magnetic field in the domain.

2.2 Physics-informed neural networks

Solving partial differential equations requires the calculation of derivatives using computer. There are four techniques to calculate derivatives in the computer program: manual differentiation, numerical differentiation, symbolic differentiation, and automatic differentiation (Baydin et al. 2018; Margossian 2019). The manual differentiation just refers to hard-coded the analytical expression of derviates. It is exact but there are many functions of which derivatives can not be calculated in the analytical approach. The numerical differentiation also known as finite difference method uses the definition of derivatives. The numerical differentiation is widely used to calculate derivatives because it is easy to code. However, it fundamentally has truncation error and it always needs the grid points in the domain, so the mesh must be created to use numerical differentiation. The symbolic differentiation providing the expressions of derivatives is used in computer algebra systems such as SymPy and Mathematica. Although it provides exact expression of derivatives, it can not handle complex functions due to similar problems as the manual differentiation has. The automatic differentiation basically uses the chain rule and the fact that every computation in the computer can be regarded as a calculation of a composite function of basic operations whose derivatives are known. The automatic differentiation is exact and fast compared to other techniques but its computational implementation is not easy although there are many libraries supporting to calculate the automatic differentiation[1].

Deep learning refers to a class of algorithms which are based on artificial neural networks, simply called neural networks, to solve certain problems. The best output of a neural network $\hat{\boldsymbol{u}} = \mathcal{N}(\boldsymbol{x};\boldsymbol{\theta})$ is obtained by the process of finding the an appropriate set of parameters $\boldsymbol{\theta}$ to minimize the loss function $\mathcal{L} = \mathcal{L}(\hat{\boldsymbol{u}};\boldsymbol{\xi})$. This process is mathematically a kind of solving an optimization problem. In the deep learning field, it is said that the neural networks *learn* something from provided data while people *train* the neural networks to update their parameters. There are many architectures of neural networks to solve diverse problems. The feedfoward neural network (FNN) also called multilayer perceptron (MLP) is the simplest neural network. Let $\mathcal{N}^L\colon \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$ be an L-layer neural network with N_ℓ neurons in the ℓth layer ($N_0 = d_{\text{in}},\ N_L = d_{\text{out}}$). The parameter $\boldsymbol{\theta}$ of the neural network $\mathcal{N}^L$ consists of weight matrix $\boldsymbol{W}^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and the bias vector $\mathbf{b}^\ell \in \mathbb{R}^{N_\ell}$, i.e., $\boldsymbol{\theta} = \{\boldsymbol{W}^\ell, \mathbf{b}^\ell\}_{1\le\ell\le L}$. For an elementwise nonlinear activation function σ, the output $\hat{\boldsymbol{u}} = \mathcal{N}^L(\boldsymbol{x})$ of the FNN is recursively calculated from the input $\boldsymbol{x}$ as follows (Lu et al. 2021):

$$\begin{aligned}
&\text{input layer}: \mathcal{N}^0(\boldsymbol{x}) = \boldsymbol{x} \in \mathbb{R}^{d_{\text{in}}},\\
&\text{hidden layer}: \mathcal{N}^\ell(\boldsymbol{x}) = \sigma(\boldsymbol{W}^\ell\mathcal{N}^{\ell-1}(\boldsymbol{x}) + \mathbf{b}^\ell) \in \mathbb{R}^{N_\ell}\\
&\qquad\qquad \text{for} \quad 1 \le \ell \le L-1,\\
&\text{output layer}: \mathcal{N}^L(\boldsymbol{x}) = \boldsymbol{W}^L\mathcal{N}^{L-1}(\boldsymbol{x}) + \mathbf{b}^L \in \mathbb{R}^{d_{\text{out}}}.
\end{aligned}$$

The training in the deep learning requires to minimize the loss function $\mathcal{L}$ which changes if the paramters $\boldsymbol{\theta}$ of neural networks changes. Since this is an optimization problem, many optimization algorithms are used to conduct this process. The gradient descent method, being the simplest, serves as the foundation for many optimization techniques. The main strategy of the gradient descent method is to determine the gradient of the loss function with respect to the parameters and update the

[1] `https://www.autodiff.org/`

parameters in the opposite direction of the gradient. Mathematically, it can be expressed as follows:

$$\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i - \eta \nabla_{\boldsymbol{\theta}_i} \mathcal{L}$$

where η is called a learning rate (commonly represented by α, but η is used here to avoid confusion with the force-free parameter) and i refers to an index of each element of the set of parameters $\boldsymbol{\theta}$. Due to the vast number of parameters in neural networks, calculating derivatives becomes computationally intensive. Manual, numerical, and symbolic differentiation methods are inadequate for accurately and efficiently calculating a large number of derivatives. Only automatic differentiation can fix this problem.

There are two modes in the automatic differentiation: forward-mode and reverse-mode. When calculating derivatives of a function $f\colon \mathbb{R}^n \to \mathbb{R}^m$, the foward-mode automatic differentiation is more efficient if $n < m$ and the reverse-mode automatic differentiation is more efficient if $n > m$ (Baydin et al. 2018; Margossian 2019). Since the result of the loss function is a scalar, the reverse-mode automatic differentiation is able to calulate fast many derivatives in the deep learning, which is often called the backpropagation algorithm. Therefore, all modern deep learning libraries such as TensorFlow, PyTorch, and JAX support at least reverse-mode automatic differentiation.

Physics-informed neural networks are neural networks proposed by (Raissi et al. 2019) to solve partial differential equations. The main idea of PINNs is to train feedforward neural network to make the output of the FNN to be the solution of the given PDEs. Let us consider a PDE $f(\boldsymbol{x}; \mathsf{D}(\boldsymbol{u}); \boldsymbol{\lambda}) = 0$ in the domain V with the boundary condition (BC) $\mathcal{B}(\boldsymbol{x}; \boldsymbol{u}) = 0$ at the boundary ∂V. The parameters of the PDE are denoted as $\boldsymbol{\lambda}$ and its operator is denoted as D which includes differential operators with other operations and can be linear or nonlinear. The first step is to construct an FNN $\mathcal{N}$ whose output is $\hat{\boldsymbol{u}} = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta})$ with parameter $\boldsymbol{\theta}$. The second step is to create a set of points randomly distributed in the computational domain V. Let $\mathcal{T}_f \subset V$ be the set of points inside the domain V, called collocation points and let $\mathcal{T}_b \subset \partial V$ be the set of points at the boundary. Since the parameter $\boldsymbol{\theta}$ of the FNN is initialized randomly, $\hat{\boldsymbol{u}}$ is of course different from the true solution $\boldsymbol{u}$ which satisfy the given partial differential equation and boundary condition. The third step is to define the loss function $\mathcal{L} = \mathcal{L}(\boldsymbol{\theta}; \mathcal{T})$ as a weighted sum of the mean squared errors. This loss function quantifies the discrepancy between $\hat{\boldsymbol{u}}$ and $\boldsymbol{u}$ for the partial differential equation and boundary condition:

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{T}) = w_f \mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) + w_b \mathcal{L}_b(\boldsymbol{\theta}; \mathcal{T}_b),$$

where

$$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{\boldsymbol{x} \in \mathcal{T}_f} |f(\boldsymbol{x}; \mathsf{D}(\hat{\boldsymbol{u}}); \boldsymbol{\lambda})|^2,$$

$$\mathcal{L}_b(\boldsymbol{\theta}; \mathcal{T}_b) = \frac{1}{|\mathcal{T}_b|} \sum_{\boldsymbol{x} \in \mathcal{T}_b} |\mathcal{B}(\boldsymbol{x}; \hat{\boldsymbol{u}})|^2.$$

and w_f and w_b are the weights. The operations $\mathsf{D}(\hat{\boldsymbol{u}})$ includes calculating derivatives of $\hat{\boldsymbol{u}}$ with respect to the input $\boldsymbol{x}$ which can be easily calculated via the automatic differentiation implemented in the chosen deep learning library. The last step is to find best paramter $\boldsymbol{\theta}^*$ by minimizing the loss function $\mathcal{L}$ using optimizers such as Adam (Kingma and Ba 2014) and L-BFGS (Zhu et al. 1995).

3 Methods

The neural network architecture utilized in this study is a 9-layer feedforward neural network with 8 hidden layers, each containing 256 neurons. For simplicity, let us consider the case where the batch size is one. In this case, the input $\boldsymbol{x} \in \mathbb{R}^3$ represents the position vector within the computational domain V. The components of $\boldsymbol{x}$ correspond to the Cartesian coordinates of the position. The output $\hat{\mathbf{B}} = \mathcal{N}^9(\boldsymbol{x}; \boldsymbol{\theta}) \in \mathbb{R}^3$ represents the mangetic field vector at the position $\boldsymbol{x}$. The activation function σ employed in this neural network is the sine function:

$$\begin{aligned}
&\text{input layer} : \mathcal{N}^0(\boldsymbol{x}) = \boldsymbol{x} \in \mathbb{R}^3, \\
&\text{hidden layer} : \mathcal{N}^\ell(\boldsymbol{x}) = \sin(\boldsymbol{W}^\ell \mathcal{N}^{\ell-1}(\boldsymbol{x}) + \mathbf{b}^\ell) \in \mathbb{R}^{256} \\
&\qquad\qquad \text{for} \quad 1 \le \ell \le 8, \\
&\text{output layer} : \mathcal{N}^9(\boldsymbol{x}) = \boldsymbol{W}^9 \mathcal{N}^8(\boldsymbol{x}) + \mathbf{b}^9 \in \mathbb{R}^3
\end{aligned}$$

The total number of parameters, denoted as $\{\boldsymbol{W}^\ell, \mathbf{b}^\ell\}_{1 \le \ell \le 9}$, in this neural network architecture is $528,131 = (3 \times 256 + 256) + 8 \times (256 \times 256 + 256) + (256 \times 3 + 3)$. Evidently, the parameter space where the optmization will be conducted is very high dimensional space, i.e., 528131-dimensional space. Therefore, the loss of the physics-informed neural network method does not monotonically decrease, which is a fundamental drawback of the neural network. It is worth noting that the loss of traditional optimization methods for calculating nonlinear force-free fields, as explained in section 2.1, monotonically decreases when specific conditions are satisfied.

There is already a *loss function* for the nonlinear force-free calculation problem. That is, the functional L in the optimization method (Equation (2.3)) can be used as the loss function. The computational domain V is the rectangular domain whose bottom boundaryy ($z = 0$) is the photosphere and $\hat{\mathbf{z}}$ is directed to the outside the

photosphere. The loss function $\mathcal{L}_f$ for the partial differential equations is

$$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{\boldsymbol{x} \in \mathcal{T}_f} \left[\frac{|(\nabla \times \hat{\mathbf{B}}) \times \hat{\mathbf{B}}|^2}{|\hat{\mathbf{B}}|^2} + |\nabla \cdot \hat{\mathbf{B}}|^2 \right],$$

where $\hat{\mathbf{B}} = \mathcal{N}^9(\boldsymbol{x}; \boldsymbol{\theta})$ is the output of the neural network, or the mangetic field calculated at the point $\boldsymbol{x}$ by the neural network $\mathcal{N}^9$. The loss function $\mathcal{L}_b$ for the boundary condition can be written as

$$\mathcal{L}_b(\boldsymbol{\theta}; \mathcal{T}_b) = \frac{1}{|\mathcal{T}_b|} \sum_{\boldsymbol{x} \in \mathcal{T}_b} |\hat{\mathbf{B}} - \mathbf{B}|^2,$$

where $\mathcal{T}_b$ is a subset of the boundary of the computational domain V, i.e., $\mathcal{T}_b \in \partial V$ and $\mathbf{B}$ is the given magnetic field at the boundary ∂V.

The PDE loss function $\mathcal{L}_f$ is divided into two terms as follows:

$$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) = w_{\mathrm{ff}} \mathcal{L}_{\mathrm{ff}}(\boldsymbol{\theta}; \mathcal{T}_f) + w_{\mathrm{div}} \mathcal{L}_{\mathrm{div}}(\boldsymbol{\theta}; \mathcal{T}_f)$$

with certain weights w_{ff} and w_{div}. The loss functions $\mathcal{L}_{\mathrm{ff}}$ and $\mathcal{L}_{\mathrm{div}}$ refer to the force-free condition and the solenoidal condition, respectively. The definitions of these loss functions are

$$\mathcal{L}_{\mathrm{ff}}(\boldsymbol{\theta}; \mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{\boldsymbol{x} \in \mathcal{T}_f} \frac{|(\nabla \times \hat{\mathbf{B}}) \times \hat{\mathbf{B}}|^2}{|\hat{\mathbf{B}}|^2},$$

$$\mathcal{L}_{\mathrm{div}}(\boldsymbol{\theta}; \mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{\boldsymbol{x} \in \mathcal{T}_f} |\nabla \cdot \hat{\mathbf{B}}|^2.$$

The total loss function $\mathcal{L}$ is

$$\begin{aligned} \mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f, \mathcal{T}_b) = w_{\mathrm{ff}} \mathcal{L}_{\mathrm{ff}}(\boldsymbol{\theta}; \mathcal{T}_f) + w_{\mathrm{div}} \mathcal{L}_{\mathrm{div}}(\boldsymbol{\theta}; \mathcal{T}_f) \\ + w_b \mathcal{L}_b(\boldsymbol{\theta}; \mathcal{T}_b), \end{aligned}$$

where w_b is a certain weight for the boundary condition loss function $\mathcal{L}_b$.

The performance of a numerical method only can be correctly evaluated if it is applied to the analytical solution. If the numerical method is able to reconstruct the analytical solution with boundary condition provided by that analytical solution, it satisfies the essential minimum requirement. For the nonlinear force-free field problem, there are two popular semi-analytical solutions: Low-Lou (LL) model proposed by (Low and Lou 1990) and Titov-Démoulin (TD) model introduced by (Titov and Démoulin 1999). In this thesis, the LL model is only employed to evaluate the performance of the physics-informed neural network method. However, future research may consider utilizing the TD model as a reference model.

The LL model used in this study is defined by four parameters (n, m, l, Φ). For the reference field, I employed the LL model with $(n = 1, m = 1, l = 0.3, \Phi = \pi/2)$. The computational domain was a $(64 \times 64 \times 64)$ Cartesian box. The bottom boundary condition was given by the LL model, while the top and lateral boundary conditions were determined by the potential fields calculated using the Green's function method (Sakurai 1982), which uses the normal components of magnetic fields at the bottom boundary. The number of total iterations is 50,000 and the model is saved every 100 iterations. The weights assigned to the force-free condition (w_{ff}) and the solenoidal condition (w_{div}) were both 0.1. The weight for the boundary condition (w_{b}), initially set to 1000, exponentially decreased to 1 within the first 25,000 iterations. The optimization process utilized the default Adam optimizer in PyTorch[2], with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and a learning rate that exponentially decayed 5×10^{-4} to 5×10^{-5} over the 50,000 iterations. To calculate the nonlinear force-free fields using the physics-informed neural network, I employed the NF2 code developed by (Jarolim et al. 2022). This code is publicly available under the GPLv3 license[3].

4 Results

There are *figures of merit* in order to evaluate how similar the calculated magnetic fields $\mathbf{B}$ are to the reference field $\mathbf{b}$ (Metcalf et al. 2008; Schrijver et al. 2006; Yi et al. 2022): the vector correlation metric

$$C_{\mathrm{vec}} = \frac{\sum_i \mathbf{B}_i \cdot \mathbf{b}_i}{(\sum_i |\mathbf{B}_i|^2 \sum_i |\mathbf{b}_i|^2)^{1/2}},$$

the Cauchy-Schwartz metric

$$C_{\mathrm{CS}} = \frac{1}{M} \sum_i \frac{\mathbf{B}_i \cdot \mathbf{b}_i}{|\mathbf{B}_i||\mathbf{b}_i|},$$

the normalized vector error metric

$$E_n = \frac{\sum_i |\mathbf{B}_i - \mathbf{b}_i|}{\sum_i |\mathbf{b}_i|},$$

the mean vector error metric

$$E_m = \frac{1}{M} \frac{\sum_i |\mathbf{B}_i - \mathbf{b}_i|}{\sum_i |\mathbf{b}_i|},$$

the total magnetic energy ratio

$$\epsilon = \frac{\sum_i |\mathbf{B}_i|^2}{\sum_i |\mathbf{b}_i|^2}.$$

In addition to the above metrics, the following two metrics where the reference field $\mathbf{b}$ is not used are also often employed: the total magnetic energy compared to the potential field energy

$$\epsilon_p = \frac{\sum_i |\mathbf{B}_i|^2}{\sum_i |\mathbf{B}_{\mathrm{p},i}|^2},$$

[2] `https://pytorch.org/docs/stable/generated/torch.optim.Adam.html`

[3] `https://github.com/RobertJaro/NF2`

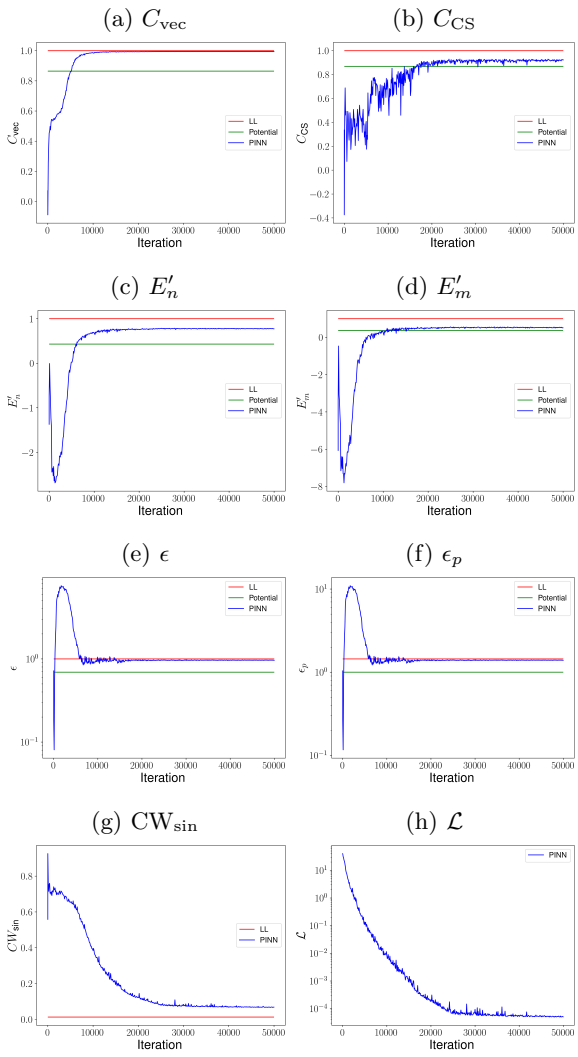

Figure 4.1: The figures of metrit and total loss of PINN with respect to the iteration steps. The red denotes Low-Lou (LL) model, the green denotes the potential field calulated using that LL model, and the blue denotes the magnetic field reconstructed by PINN. The y-axes of the graphs for ϵ, ϵ_p, and $\mathcal{L}$ are in a base-10 logarithmic scale.

the current-weighted sine metric (it is originally denoted by σ_J in Wheatland et al. 2000)

$$\mathrm{CW}_{\mathrm{sin}} = \frac{\sum_i \frac{|\mathbf{J}_i \times \mathbf{B}_i|}{|\mathbf{B}_i|}}{\sum_i |\mathbf{J}_i|}.$$

In the expressions of the figures of merit, the variable i represents each individual point within the computational domain, M corresponds to the total number of points in the computational domain, $\mathbf{B}_{\mathrm{p}}$ is the potential magnetic field whose bottom boundary condition is the same as that of $\mathbf{B}$, and $\mathbf{J} = \nabla \times \mathbf{B}$ is the current density in an appropriate unit system. The vector correlation metric C_{vec} serves as a correlation coefficient specifically designed for vector fields. It has a value of 1 when the vectors $\mathbf{B}$ and $\mathbf{b}$ are identical, and 0 when $\mathbf{B}_i$ is perpendicular to $\mathbf{b}_i$ at each point. The Cauchy-Schwartz metric C_{CS} quantifies the angles between vector fields. For $\mathbf{B}$ and $\mathbf{b}$ that are parallel, C_{CS} equals 1, while it becomes -1 for an anti-parallel case. When $\mathbf{B}_i$ is perpendicular to $\mathbf{b}_i$ at each point, C_{CS} has a value of 0. To measure the disparities between vector fields, the normalized and mean vector error metrics, E_n and E_m respectively, are employed. Both E_n and E_m are set to 0 when $\mathbf{B}$ and $\mathbf{b}$ are identical. Since only these two metrics yield zero for a perfect match between $\mathbf{B}$ and $\mathbf{b}$, it is common to use $E'_n = 1 - E_n$ and $E'_m = 1 - E_m$ instead. Throughout this thesis, I have utilized E'_n and E'_m. The total magnetic energy ratio, denoted as ϵ, represents the ratio between the total magnetic energy of the two vector fields, $\mathbf{B}$ and $\mathbf{b}$. When $\mathbf{B}$ and $\mathbf{b}$ are identical, ϵ equals 1. Hence, if the physics-informed neural network method accurately reconstructs the Low-Lou analytical solution, all five metrics—C_{vec}, C_{CS}, E'_n, E'_m, and ϵ—will be equal to 1.

Considering that the potential field represents the state of lowest possible energy of any force-free fields with the same boundary condition, the magnetic energy of a nonlinear force-free field always surpasses that of the potential field. The surplus energy is known as free magnetic energy, which characterizes the energy associated with solar activities. The ratio ϵ_p between the total magnetic energy and the potential field energy indicates the amount of free magnetic energy stored in the nonlinear force-free field. For any field other than the potential field, ϵ_p exceeds 1. The current-weighted sine metric, denoted as $\mathrm{CW}_{\mathrm{sin}}$, serves as an indicator of the force-freeness of the calculated magnetic field $\mathbf{B}$. When $\mathbf{B}$ is an exact force-free field, $\mathrm{CW}_{\mathrm{sin}}$ equals 0. However, this metric, $\mathrm{CW}_{\mathrm{sin}}$, cannot be defined for the potential field because it has zero current density, $\mathbf{J} = \mathbf{0}$.

The optimizer of PINN aims to minimize the loss function $\mathcal{L}$ in a 528,131-dimensional parameter space. As a result, the loss function of PINN does not typically exhibit a monotonic decrease with respect to the iteration steps, as shown in fig. 4.2. Although all the losses generally exhibit a monotonically decreasing trend after reaching certain critical points, the presence of numerous spikes in the graph indicates the challenges posed by the high-dimensionality of the parameter space. Initially, due to the high value of the boundary condition weight w_b (starting from 1000), the weighted boundary condition loss $w_b\mathcal{L}_b$ dominates the total loss. Subsequently, as the boundary condition weight w_b exponentially decreases, the weighted PDE losses $w_{\mathrm{ff}}\mathcal{L}_{\mathrm{ff}}$ and $w_{\mathrm{div}}\mathcal{L}_{\mathrm{div}}$ become increasingly influential in the total loss. In that time, they already has the decreasing pattern like the BC loss. Therefore, the

overall pattern of the total loss closely resembles that of the boundary condition loss.

The force-free condition loss $\mathcal{L}_{\rm ff}$ and the solenoidal condition loss $\mathcal{L}_{\rm div}$ display patterns of both increase and decrease. This behavior can be attributed to the fact that the initial field generated randomly for PINN, in this case, is a nearly constant vector field, as depicted in (a) of figs. 4.3 to 4.6. Consequently, the derivatives of the initial magnetic field are close to zero, resulting in nearly zero PDE losses $\mathcal{L}_{\rm ff}$ and $\mathcal{L}_{\rm div}$.

As the training progresses, the field reconstructed by PINN satisfies the boundary conditions at ∂V, but it may not adhere to the force-free condition and solenoidal condition due to the initially high boundary condition weight w_b. Once the weighted boundary condition loss decreases sufficiently, the optimizer focuses on minimizing the PDE losses within the volume V, which now constitute the majority of the total loss. Consequently, the field gradually evolves into a nonlinear force-free field that satisfies the prescribed boundary conditions while simultaneously reducing the PDE losses.

Although the primary objective of the optimizer is to minimize the total loss $\mathcal{L}$, the seven metrics ($C_{\rm vec}$, $C_{\rm CS}$, E'_n, E'_m, ϵ, ϵ_p, and $\mathrm{CW}_{\rm sin}$) of the PINN-reconstructed magnetic field closely resemble those of the Low-Lou model, as depicted in fig. 4.1. This observation confirms that the loss function employed in traditional optimization methods can also be utilized in the PINN approach, demonstrating the ability of the physics-informed neural network method to reconstruct nonlinear force-free fields. It is worth noting that the PINN-reconstructed solution surpasses the potential field as it exhibits closer proximity to the Low-Lou analytical solution.

However, while $\mathcal{L}$ serves as a valuable loss function, it is not flawless. As shown in table 4.1, the figures of merit ($C_{\rm vec}$, E'_n, E'_m, ϵ, and ϵ_p) for the PINN(50000) model are inferior (deviating from the values of the Low-Lou model) compared to those of the PINN(25000) model, even though $\mathcal{L}$ decreases from 0.00006 to 0.00005. Meanwhile, $C_{\rm CS}$ and $\mathrm{CW}_{\rm sin}$ for the PINN(50000) model outperform those of the PINN(25000) model. This observation suggests the need for the development of more appropriate loss functions in future research.

Furthermore, the energy metrics ϵ and ϵ_p of the PINN(50000) model exhibit lower values compared to the Low-Lou solution. This indicates that the total magnetic energy and free magnetic energy of the PINN(50000) model are smaller than those of the Low-Lou solution. This finding may imply the presence of spectral bias in the PINN method (Rahaman et al. 2019), i.e., the neural network tends to prioritize learning low-frequency features while facing challenges with high-frequency features. Actually, a smooth distribution of magnetic fields,

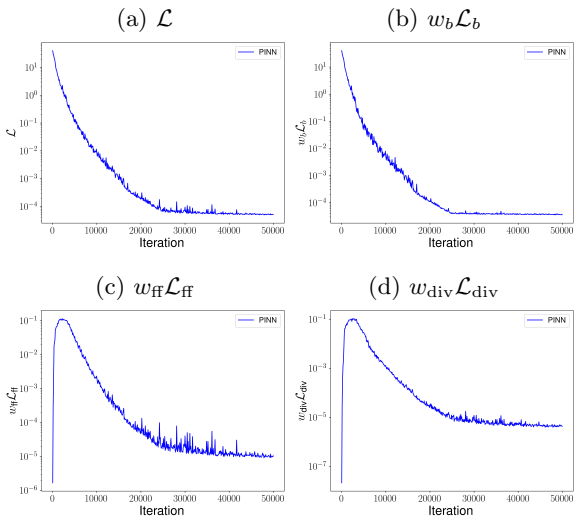

Figure 4.2: The values of loss funtions with respect to the iteration steps. (a) The total loss of PINN. The weighted loss of (b) the boundary condition, (c) the force-free condition, and (d) the solenoidal condition. The y-axes of these graphs are in a base-10 logarithmic scale.

favored by the PINN approach, typically results in lower magnetic energy compared to a more spiky distribution because of the squaring operation in the definition of magnetic energy $\int_V \frac{B^2}{8\pi}\mathrm{d}V$: $8 = 2^2 + 2^2 < 3^2 + 1^2 = 10$.

The colors in the $z = 0$ plane in the figs. 4.3 to 4.6 follows the typical convention for solar magnetograms: white represents positive B_z values, while black represents negative B_z values. All figures share the same scalar bar, which spans a range of (-150, 150). Additionally, 25 field lines have been selected, and each field line is assigned a unique color to distinguish it from the others. The colors are randomly assigned based on the seed value calculated from the coordinates of the field line's starting point. Thus, the field lines originating from the same point have the same color across all figures. These figures demonstrate that the PINN method successfully reconstructs the nonlinear force-free fields described by the Low-Lou model in qualitative terms. Furthermore, PINN(0) in these figures highlight that PINN does not require any pre-calculated initial field, such as a potential field, as is typically needed in traditional optimization methods.

5 Conclusions

The findings of this study yield valuable insights into the performance and capabilities of the physics-informed neural network method in calculating nonlinear force-free fields. The evaluation of various

Table 4.1: Figures of merit for various fields and total loss of PINN. The reference field is the Low-Lou model with $(n = 1, m = 1, l = 0.3, \Phi = \pi/2)$. PINN($i$) refers to the magnetic field calculated using PINN at the iteration step i.

Field	C_{vec}	C_{CS}	E'_n	E'_m	ϵ	ϵ_p	$\mathrm{CW}_{\mathrm{sin}}$	$\mathcal{L}^a$
Low-Lou	1.00000	1.00000	1.00000	1.00000	1.00000	1.44490	0.01308[b]	–
Potential	0.86465	0.86925	0.43003	0.36242	0.69209	1.00000	–[c]	–
PINN(0)	0.07115	0.33540	-1.37088	-6.07067	0.71824	1.03779	0.55813	40.23570
PINN(100)	0.25771	0.46100	-0.00643	-0.46701	0.08056	0.11640	0.76066	31.32017
PINN(1000)	0.54244	0.32404	-2.31914	-6.41640	5.59377	8.08246	0.71701	6.46577
PINN(10000)	0.98630	0.63065	0.68766	0.28275	0.96939	1.40068	0.38403	0.00848
PINN(25000)	0.99402	0.92134	0.78452	0.55897	0.96481	1.39406	0.07853	0.00006
PINN(50000)	0.99359	0.92271	0.77129	0.50928	0.96247	1.39068	0.06848	0.00005

[a] The total loss of PINN.
[b] This is not zero due to numerical grids.
[c] The metric $\mathrm{CW}_{\mathrm{sin}}$ is undefined for the potential field.

metrics demonstrates the PINN approach's ability to accurately reconstruct the Low-Lou model. The majority of these metrics demonstrate values that are quite similar with the analytical solutions, highlighting a substantial level of agreement between the PINN-reconstructed fields and the analytical solutions (Table 4.1 and Figure 4.1). Furthermore, this agreement is qualitatively confirmed by observing the graphical representation of the magnetic field (Figures 4.3 to 4.6). Additionally, the PINN approach successfully surpasses the potential field in terms of proximity to the Low-Lou solution, highlighting its effectiveness in capturing the complex magnetic field structures associated with solar activities.

While the PINN approach proves to be a promising method for calculating nonlinear force-free fields, this study also sheds light on its limitations. The figures of merit for the PINN(50000) model are inferior to those of the PINN(25000) model, despite the former having a lower loss value (Table 4.1). This indicates the need for further exploration and refinement of loss functions. Additionally, the magnetic energy of the PINN(50000) model is lower than that of the analytical solution, suggesting that the PINN exhibits a spectral bias and struggles to learn high-frequency features. Enhancing the loss function and neural network architecture could improve the PINN's ability to capture high-frequency features and achieve even closer agreement with the analytical solutions. This research underscores the importance of ongoing efforts to enhance the performance and accuracy of the PINN approach, allowing for its wider application in solar physics and computational magnetohydrodynamics.

This thesis has presented a novel approach to calculating nonlinear force-free fields by employing physics-informed neural networks. This research specifically focuses on the application of physics-informed neural networks to the Low-Lou model, which provides semi-analytical solutions for nonlinear force-free fields. The results demonstrate the effectiveness of the proposed approach in accurately computing these fields. The successful implementation of physics-informed neural networks in this context highlights their versatility and potential for further advancements in the study of coronal magnetic fields and their relationship to solar phenomena. By recognizing the significance of coronal magnetic fields, this study demonstrates the potential of physics-informed neural networks in solar physics research.

References

Baydin, Atilim Gunes, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind (2018). "Automatic differentiation in machine learning: a survey". In: *Journal of Machine Learning Research* 18, pp. 1–43.

Cargill, Peter J (2009). "Coronal magnetism: difficulties and prospects". In: *The Origin and Dynamics of Solar Magnetism*, pp. 413–421.

Gary, G Allen (2001). "Plasma beta above a solar active region: rethinking the paradigm". In: *Solar Physics* 203, pp. 71–86.

Jarolim, Robert, Julia Thalmann, Astrid Veronig, and Tatiana Podladchikova (2022). "Probing the solar coronal magnetic field with physics-informed neural networks". In: *Research Square preprint*.

Karniadakis, George Em, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang (2021). "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6, pp. 422–440.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

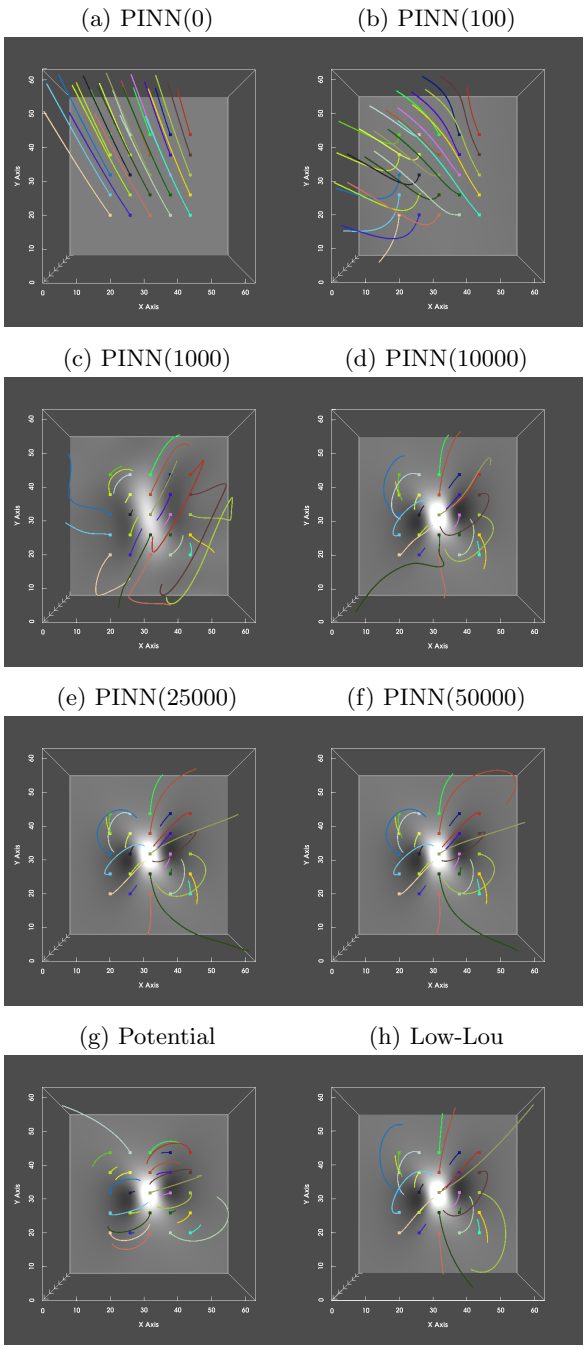

Figure 4.3: The magnetic fields in the xy plane. Each field line is distinguished by its unique color, which is determined by its footpoint at $z = 0$ plane.

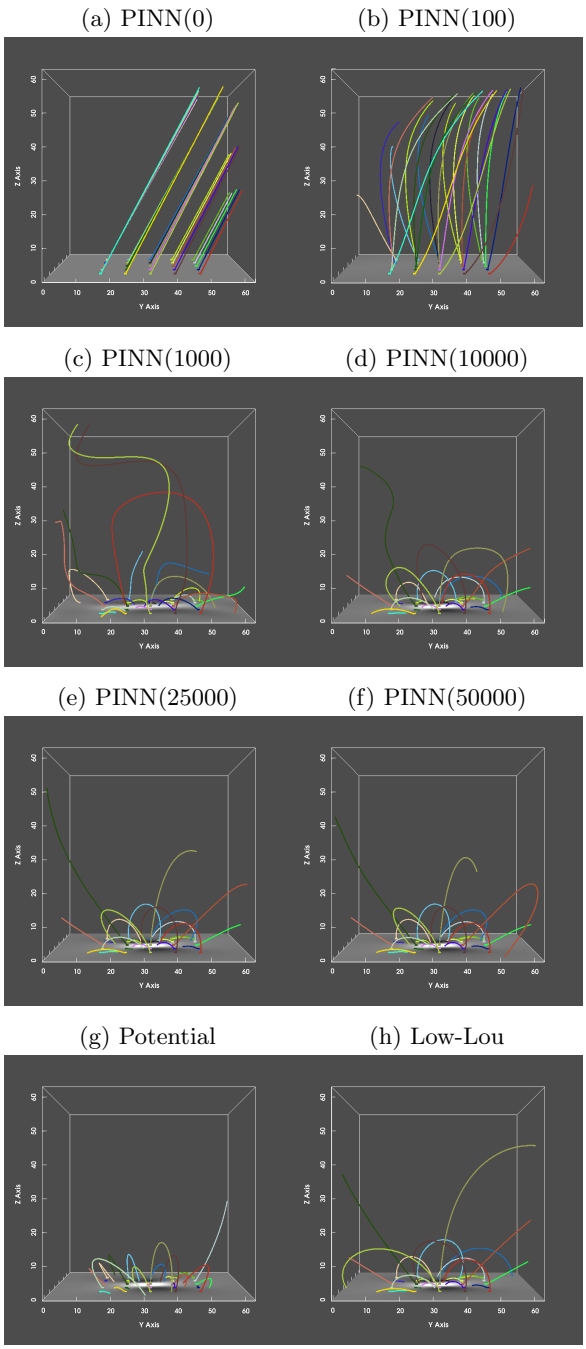

Figure 4.4: The magnetic fields in the yz plane. Each field line is distinguished by its unique color, which is determined by its footpoint at $z = 0$ plane.

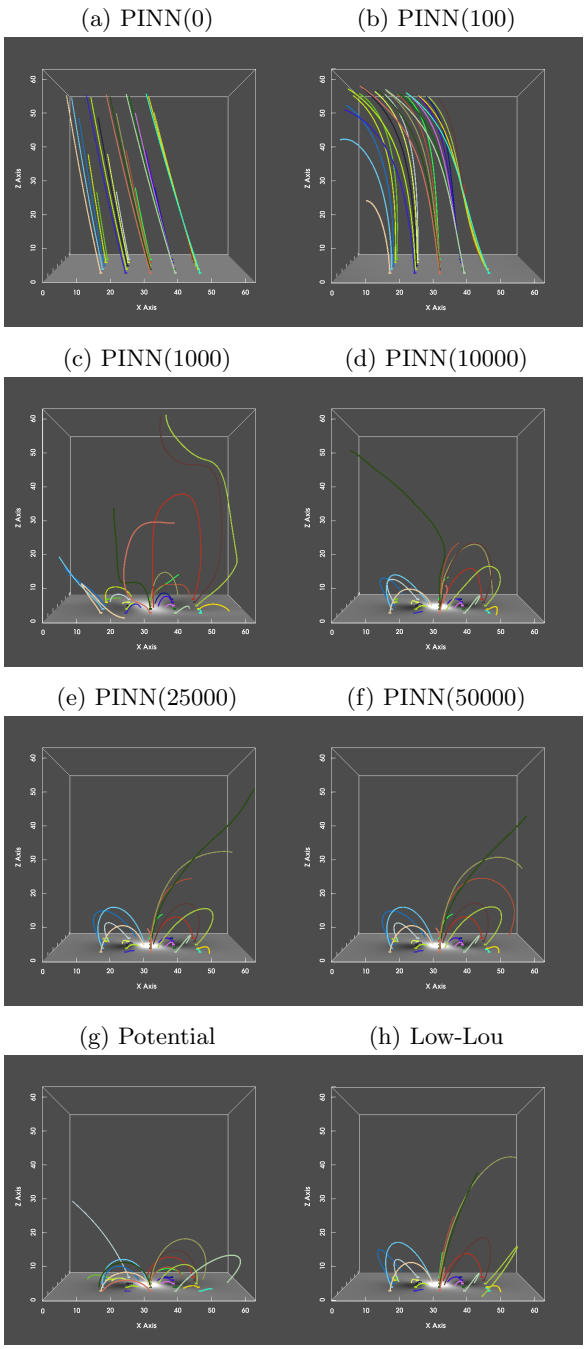

Figure 4.5: The magnetic fields in the xz plane. Each field line is distinguished by its unique color, which is determined by its footpoint at $z = 0$ plane.

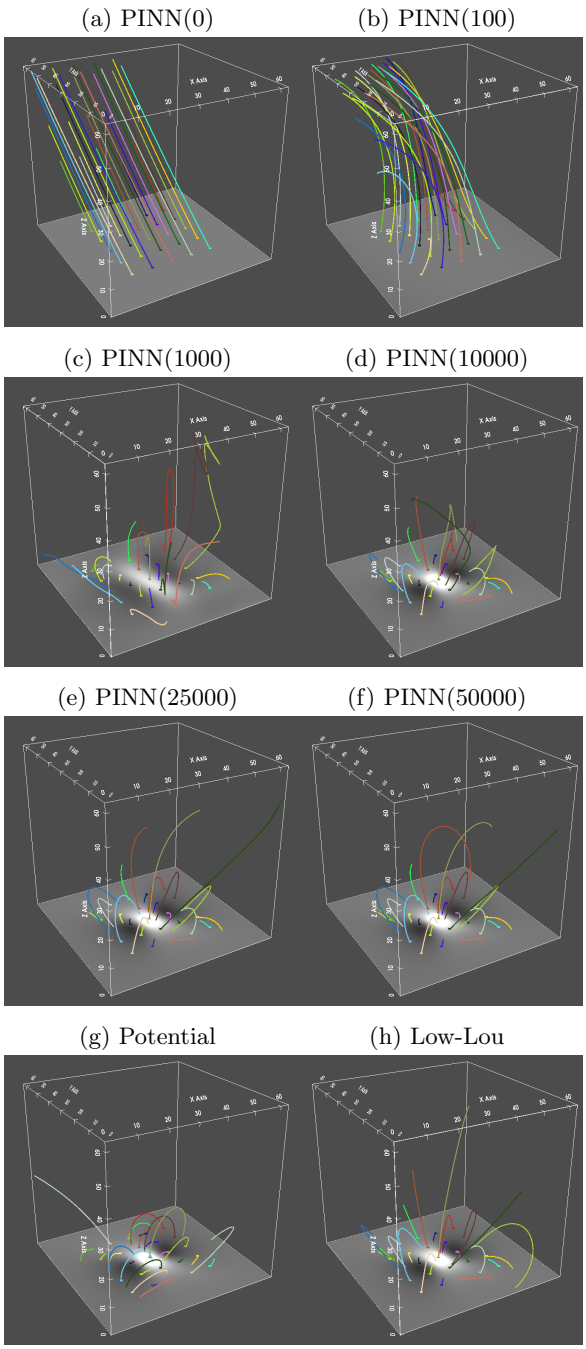

Figure 4.6: The magnetic fields. Each field line is distinguished by its unique color, which is determined by its footpoint at $z = 0$ plane.

Low, BC and YQ Lou (1990). "Modeling solar force-free magnetic fields". In: *The Astrophysical Journal* 352, pp. 343–352.

Lu, Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis (2021). "DeepXDE: A deep learning library for solving differential equations". In: *SIAM review* 63.1, pp. 208–228.

Margossian, Charles C (2019). "A review of automatic differentiation and its efficient implementation". In: *Wiley interdisciplinary reviews: data mining and knowledge discovery* 9.4, e1305.

Metcalf, Thomas R, Marc L DeRosa, Carolus J Schrijver, Graham Barnes, Adriaan A van Ballegooijen, Thomas Wiegelmann, Michael S Wheatland, Gherardo Valori, and James M McTtiernan (2008). "Nonlinear force-free modeling of coronal magnetic fields. II. Modeling a filament arcade and simulated chromospheric and photospheric vector fields". In: *Solar Physics* 247, pp. 269–299.

Rahaman, Nasim, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville (2019). "On the spectral bias of neural networks". In: *International Conference on Machine Learning*. PMLR, pp. 5301–5310.

Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational physics* 378, pp. 686–707.

Sakurai, Takashi (1982). "Green's function methods for potential magnetic fields". In: *Solar Physics* 76, pp. 301–321.

Schou, Jesper, JM Borrero, AA Norton, Steven Tomczyk, D Elmore, and GL Card (2012). "Polarization calibration of the helioseismic and magnetic imager (HMI) onboard the Solar Dynamics Observatory (SDO)". In: *The Solar Dynamics Observatory*, pp. 327–355.

Schrijver, Carolus J, Marc L Derosa, Thomas R Metcalf, Yang Liu, Jim Mctiernan, Stéphane Régnier, Gherardo Valori, Michael S Wheatland, and Thomas Wiegelmann (2006). "Nonlinear force-free modeling of coronal magnetic fields part I: a quantitative comparison of methods". In: *Solar Physics* 235, pp. 161–190.

Stenflo, Jan Olof (2013). "Solar magnetic fields as revealed by Stokes polarimetry". In: *The Astronomy and Astrophysics Review* 21, pp. 1–58.

Titov, VS and Pascal Démoulin (1999). "Basic topology of twisted magnetic configurations in solar flares". In: *Astronomy and Astrophysics* 351, pp. 707–720.

Wheatland, MS, PA Sturrock, and G Roumeliotis (2000). "An optimization approach to reconstructing force-free fields". In: *The Astrophysical Journal* 540.2, p. 1150.

Yi, Sibaek, GS Choe, Kyung-Suk Cho, Sami K Solanki, and Jörg Büchner (2022). "Reconstruction of Coronal Magnetic Fields Using a Poloidal–Toroidal Representation". In: *The Astrophysical Journal* 937.1, p. 11.

Zhu, Ciyou, RH Byrd, P Lu, and J Nocedal (1995). "A limited memory algorithm for bound constrained optimisation". In: *SIAM J. Sci. Stat. Comput* 16.5, pp. 1190–1208.