（千葉大学審査学位論文）

Velocity field evaluation in the solar convection layer by machine learning and numerical simulation

February 2025

Hiroyuki Masaki

Graduate School of Science and Engineering
CHIBA UNIVERSITY

In this study, a neural network was developed to evaluate the horizontal velocity field at the solar surface from snapshots of solar intensity, line-of-sight velocity fields, and line-of-sight magnetic fields by combining machine learning and numerical simulations. Various phenomena related to turbulence and magnetic fields occur at the solar surface, and it is known that they also affect the Earth's environment. Therefore, observing solar surface phenomena and evaluating physical quantities have attracted much attention, leading to many high-precision observations. However, some physical quantities are difficult to observe, hindering our understanding of the solar surface. On the other hand, numerical simulations have become more accurate each year due to advances in computer performance and the development of new computational methods. Many simulations have been conducted to replicate solar conditions, successfully reproducing phenomena occurring in the Sun. In this study, we performed a numerical simulation replicating the solar convection zone and used the data obtained as training data for the neural network, developing an algorithm to acquire horizontal velocity fields at the surface and rising velocity fields inside the Sun, which are difficult to observe. The correlation coefficient between the evaluated values using the developed network and the simulation data was approximately 0.9 for the surface velocity field, indicating high predictive performance. For the internal velocity field, even at depths of about 10 Mm, large-scale structures were evaluated with a correlation coefficient of over 0.5.

The performance of these networks was validated by comparing them with Local Correlation Tracking (LCT) existing surface horizontal velocity evaluation methods and helioseismology, which is the method to evaluate internal structures using observational data.

Contents

Chapter 1 Introduction 1

1.1 Solar Thermal Convection 1

1.2 Formation of Emerging Magnetic Fields 5

1.3 Observational Techniques for the Sun 7

1.3.1 Optical Observations of the Sun 7

1.3.2 Local Correlation Tracking (LCT) 7

1.3.3 Helioseismology 9

1.4 Neural Networks 12

1.5 Purposes of this study 13

1.6 Previous Research 14

Chapter 2 Method 17

2.1 Research Outline 17

2.2 Numerical Simulation 18

2.3 Neural Network Structure 20

2.3.1 Convolutional Neural Networks 20

2.3.2 U-net 22

Chapter 3 Evaluation of Surface Velocity Field 23

3.1 Simulation Setting 23

3.2 Training Setup 24

3.3 Evaluation Results 26

3.4 Dependence of Network Performance on Data Amount 31

3.5 Dependence of Network Performance on Input Quantities 33

3.6 Application to Observations 35

3.7 Comparison with LCT 42

3.8 Comparison with Previous Studies 44

Chapter 4 Evaluation of Internal Velocity Fields 47
4.1 Simulation Setting 47
4.2 Training Setup 50
4.3 Adjustment of Simulation Results to Match Observation 51
4.4 Training Results 51
4.5 Comparison with Helioseismology 54
4.5.1 Comparison with Simulation Data 55
4.5.2 Comparison with Observational Data 55
4.6 Dependence of Network Performance on Input Quantities 59

Chapter 5 Summary and Discussion 69
5.1 Surface Velocity Field 69
5.2 Internal Velocity Field 71
5.3 Future Prospects 72

AppendixA Fourth-Order Accurate Differentiation 77
A.1 Spatial Differentiation 77
A.2 Temporal Differentiation 77

AppendixB Network structure and Update 79
B.1 Gradient Descent Method 79
B.2 Neural Network structure and Gradient Calculation 79
B.2.1 Skip Connection 80
B.2.2 Error Function 80
B.2.3 Activation Function 81
B.2.4 Fully Connected Network 81
B.2.5 Convolutional Networks 82
B.2.6 Max Pooling 82
B.2.7 Upsampling 83
B.2.8 Batch Normalization 83
B.3 Adam 85

AppendixC Observation method 89
C.1 Local correlation tracking(LCT) 89
C.1.1 Fourier Local correlation tracking(FLCT) 89
C.1.2 LCT Parameter Survey 90

C.2 Helioseismology 94

Acknowledgments 97

References 103

Chapter 1

Introduction

1.1 Solar Thermal Convection

The Sun shines due to nuclear fusion in the core. The energy produced by nuclear fusion is transported outward by radiation from the center of the Sun up to 70% of the solar radius, covering a region of 500 Mm (the solar radius is about 700 Mm), which is called the radiation zone (Stix, 2002). Beyond the radiation zone, from 70% of the solar radius up to the surface (approximately 200 Mm), energy is transported outward by thermal convection of the plasma that constitutes the Sun. This region is referred to as the convective zone. Figure 1.1 shows a schematic diagram of the sun structure layer. In the radiation zone, the temperature gradient is determined by the energy transport by radiation. However, in the convective zone, due to increased opacity, the temperature gradient determined by radiation energy transport becomes steeper than the value known as the adiabatic temperature gradient. This corresponds to a decrease in entropy in the radial direction. In such conditions, when fluid parcels, which change adiabatically, rise (or sink), they become less dense (or denser), leading to increased instability, resulting in the phenomenon of thermal convection. Near the solar surface, the opacity decreases again, making the region thermally stable against convection. The granulated structures that appear on the solar surface due to this thermal convection are called granulation. Figure 1.2 shows an image of the quiet Sun, a region without sunspots, observed by the DKIST (The Daniel K. Inouye Solar Telescope). The granulations resulting from thermal convection can be seen. The typical lifetime of this granulation is a few minutes, with a size of approximately 1 Mm and a flow speed of 3-4 $\mathrm{km\ s^{-1}}$ (Spruit et al., 1990).

The solar surface exhibits a structure larger than granulation, which is known as supergranulation. This structure cannot be observed directly from simple radiation intensity snapshots but can be detected by measuring the Doppler velocity at the solar surface. Figure 1.3 shows

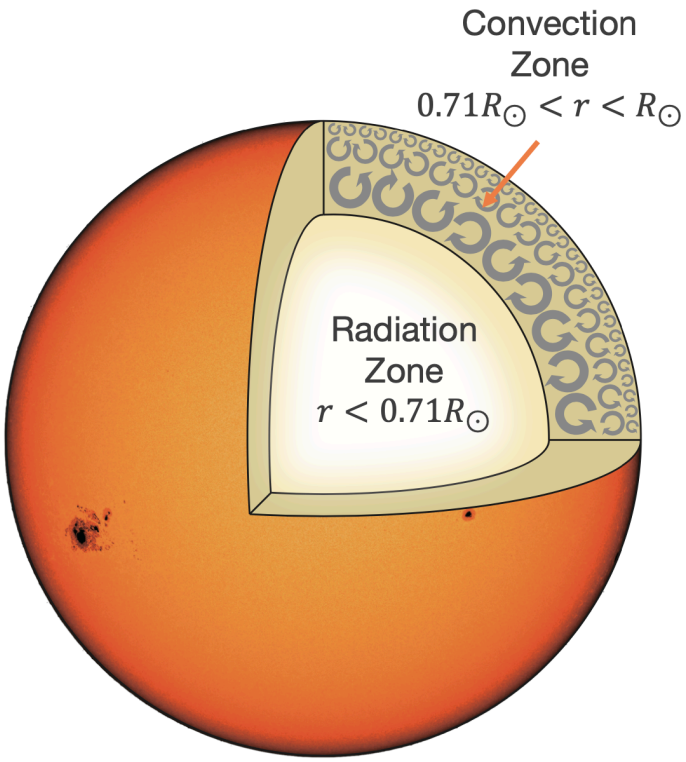


Figure1.1 A schematic diagram of thermal convection in the Sun. $R_\odot$ represents the solar radius, which is approximately 700 Mm. Created by Hideyuki Hotta.

the Doppler velocity of the surface averaged over an hour observed by the SDO/HMI (Solar Dynamics Observatory, Helioseismic and Magnetic Imager, Pesnell et al. (2012)), with the rotation velocity removed. We can see large-scale horizontal flows. The scale of supergranulation is approximately 30 Mm horizontally and has a lifetime of about a day. It is thought to represent convective structures deeper within the solar interior, i.e., the supergranulation roots in a deeper layer than the granulation(Rincon and Rieutord, 2018).

The magnetic field also exists in the quiet Sun. While the average magnetic field strength in the quiet region is around several tens of Gauss (Bellot Rubio and Orozco Suárez, 2019), there are localized areas where the magnetic field exceeds 1000 G. These regions with strong magnetic fields appear as bright points. In regions with strong magnetic fields, the plasma is pushed aside due to magnetic pressure, revealing deeper layers where the temperature is higher, which makes bright points(Stein, 2012). In Figure 1.2, bright regions can be seen in the center, suggesting the presence of strong magnetic fields. As the magnetic fields concentrate, thermal convection is suppressed, causing temperature and pressure to decrease. Due to the reduced pressure, the region becomes further compressed. This process is repeated, leading to the generation of strong magnetic fields. While this mechanism is a major theory, its

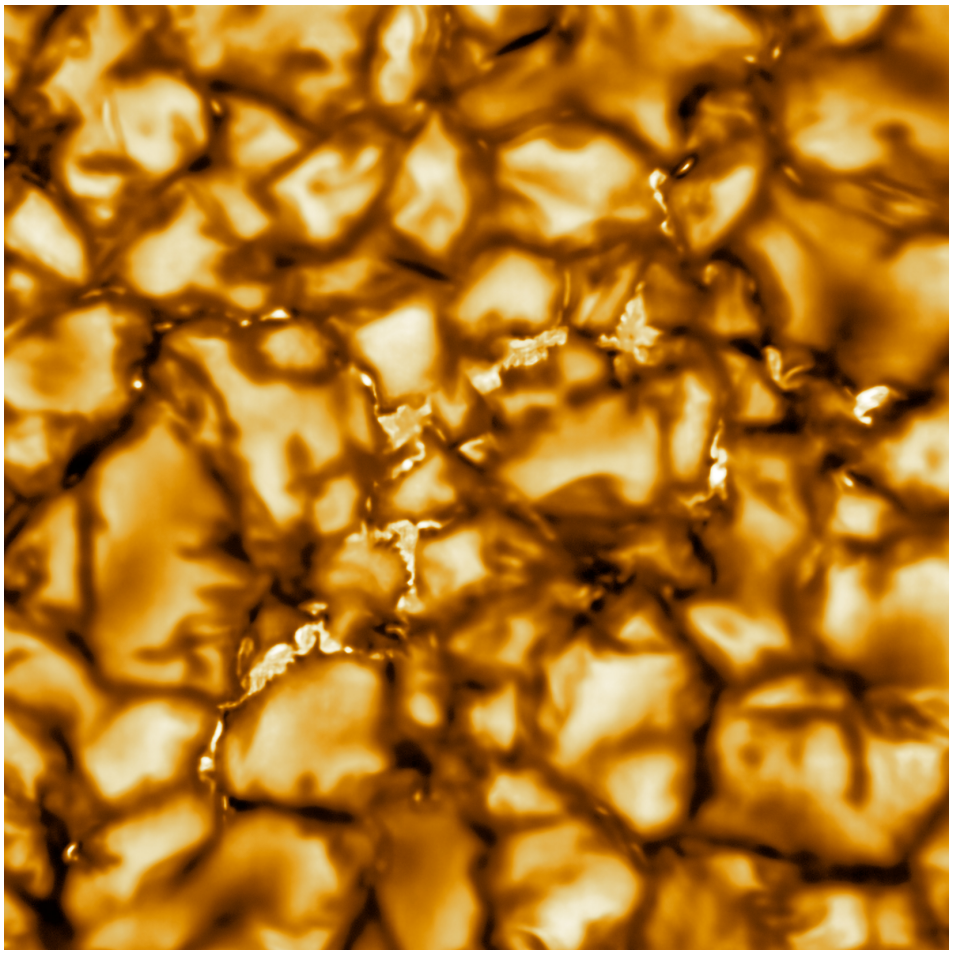


Figure1.2 Observations of the quiet Sun surface by DKIST. This is an 8,200 × 8,200 km region of the solar surface.
Source: https://nso.edu/inouye-solar-telescope-first-light.

details are still being studied (Sakurai et al., 2018). Such strong magnetic field dynamics are influenced by internal flows (Takahata et al., 2021). The interaction between the flow and the magnetic field in the deep layer is an important topic.

The temperature of the solar corona, which appears in the outermost part of the Sun, is about one million K, significantly higher than the surface temperature of 6000 K. The energy source for heating the corona is roughly thought to be the thermal convection on the solar surface and the oscillations of the magnetic field excited by this convection. However, the detailed mechanisms remain unclear. Two mechanisms have been proposed: :1. Convective oscillations transport energy upward through magnetoacoustic waves and dissipate in the corona, and 2. Magnetic fields twisted by convection trigger magnetic reconnection, and magnetic energy is converted to internal energy. A consensus has not yet been reached (Arregui and Van Doorsselaere, 2024).

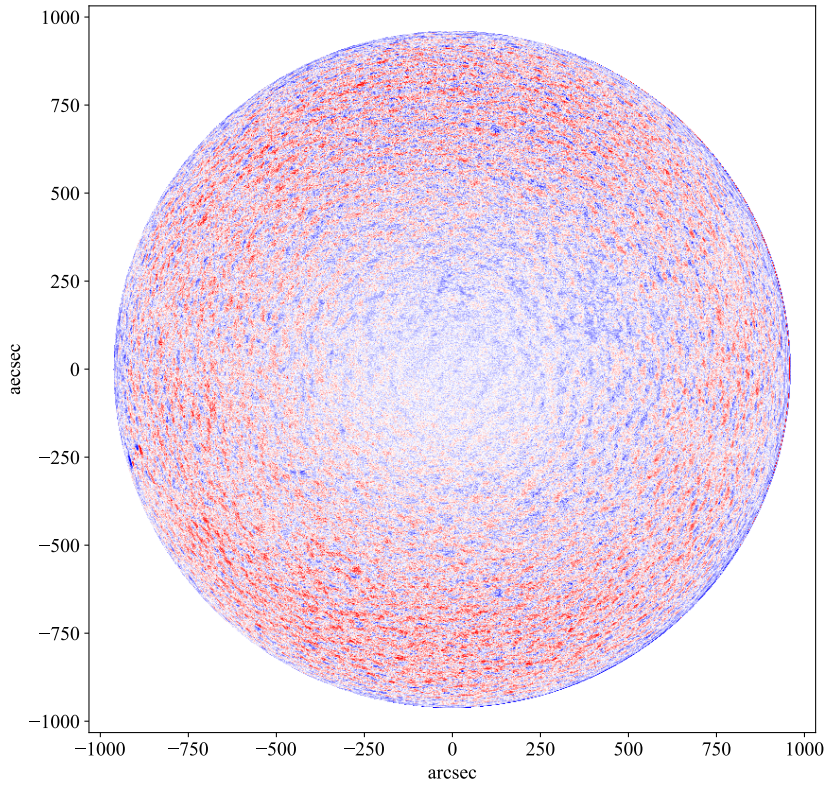


Figure1.3 Line-of-sight velocity field observed by the HMI on board the SDO. Red indicates velocity toward the back, and blue towards the front. The HMI uses the Doppler shift of the Fe i absorption line to measure the velocity. Using data taken from http://jsoc.stanford.edu/

As such, convective motion and the associated magnetic fields are said to be related to unresolved issues in solar physics, such as coronal heating and magnetic field generation, making them critically important structures.

Phenomena occurring on the Sun also affect the Earth through space weather. Therefore, observing the Sun and understanding its conditions and phenomena are of significant importance. Additionally, since the Sun is the closest star to Earth, studying it helps in understanding the properties of other stars, which are farther away and difficult to observe with high precision.

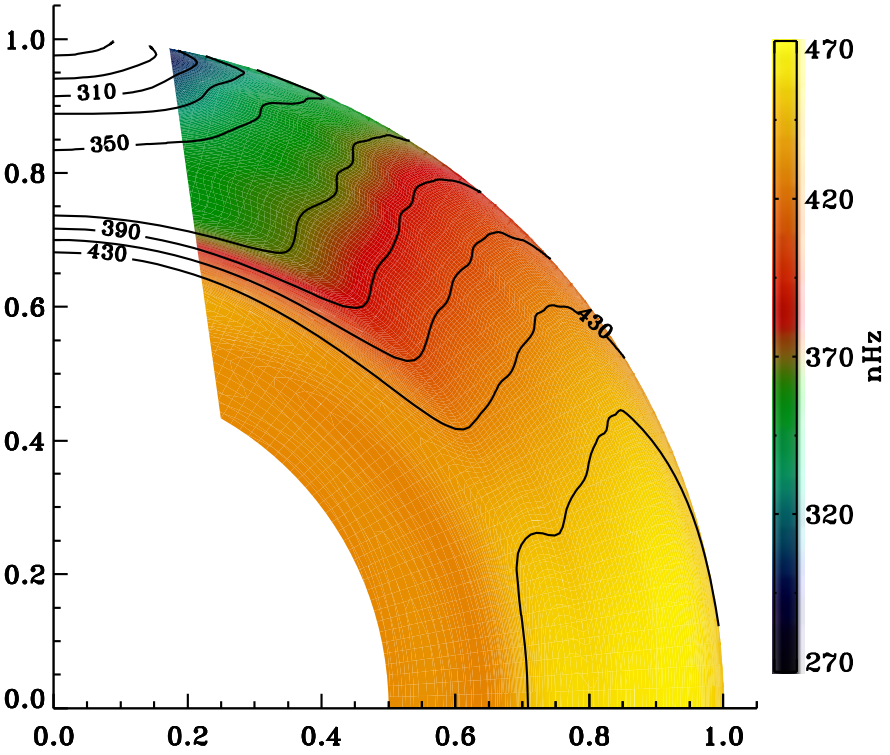


Figure1.4 Cited from Larson and Schou (2018) (Figure 30, left panel). The rotation rate in the radiation zone is thought to be uniform ($< 0.7R_{\odot}$).

1.2 Formation of Emerging Magnetic Fields

The solar interior convection transports angular momentum, resulting in differences in angular velocity depending on location. This is called differential rotation (Figure 1.4). The differential rotation is pronounced around the boundary between the radiative and convective zones, known as the tachocline, where the magnetic field is believed to be stretched and strengthened. The magnetic flux tubes inside the Sun push aside the surrounding plasma due to magnetic pressure, reducing their density and allowing them to rise due to buoyancy. This buoyancy is called magnetic buoyancy. When these magnetic fields appear on the surface, they suppress convective heat transport, leading to a temperature drop, and are observed as sunspots.

As the magnetic flux tubes rise due to buoyancy, they are also influenced by convective flows. Figure 1.5 shows simulation results of emerging magnetic fields in a convective environment. A magnetic flux tube is placed in an environment where the center upflows,

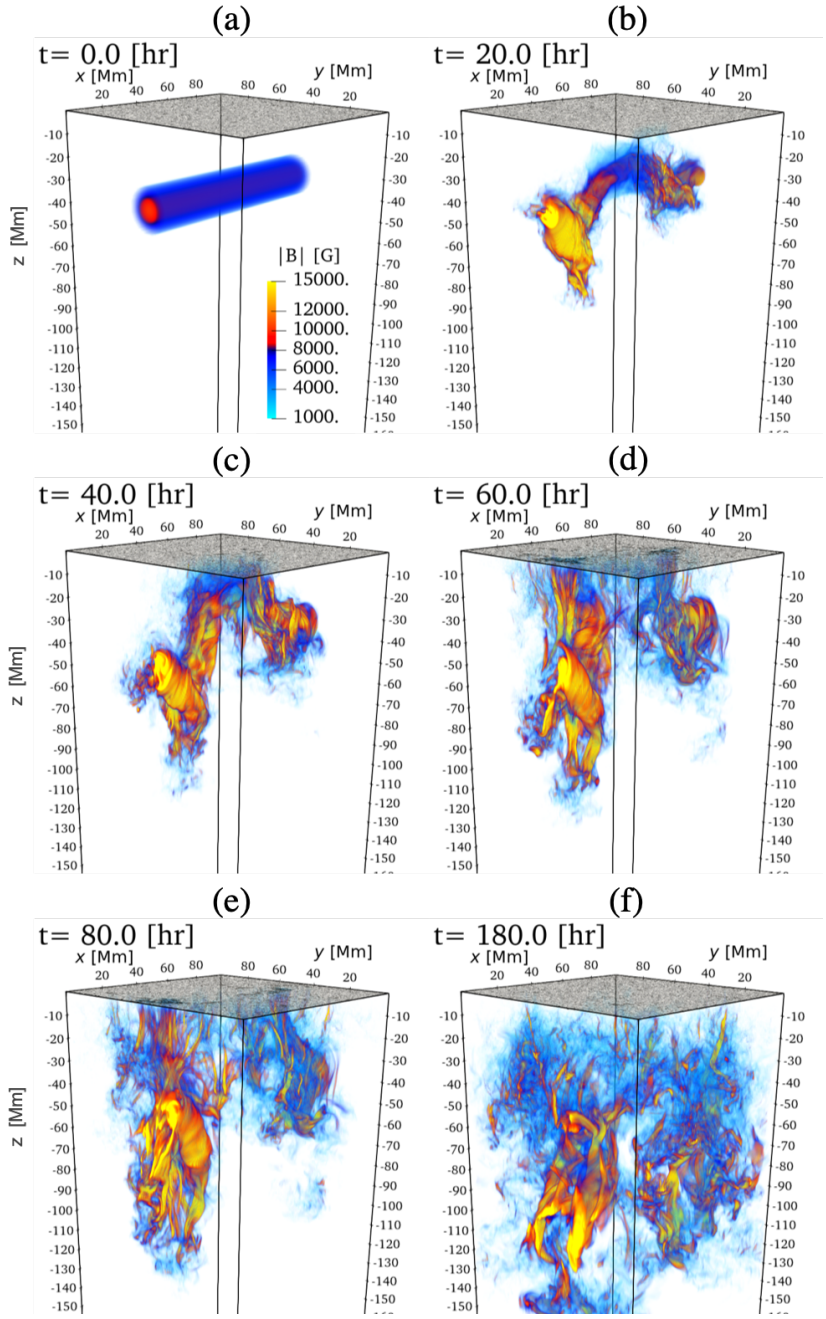


Figure1.5 Time evolution of internal magnetic field strength in a flux emergence simulation. Hotta and Iijima (2020)

and the surroundings downflows rise at the center and descend at the periphery, respectively, similar to convective patterns. It can be concluded that sunspots are more likely to appear in regions with upward flows initially.

1.3 Observational Techniques for the Sun

Solar observations have been conducted for a long time, and today, high-precision observations are possible.

1.3.1 Optical Observations of the Sun

At present, it is extremely difficult to visit the Sun directly to obtain information(in-situ observation). Therefore, observations of the Sun have primarily been conducted by capturing light(remote-sensing observation). The simplest quantity that can be observed is radiation intensity.

Furthermore, by conducting spectroscopic observations of the Sun's light, a great deal of information can be obtained. Electrons in atoms, for instance, have discrete energy levels, and in order to transition between these levels, specific energy of light is absorbed. Since the wavelength of this absorbed light is unique to each material, by conducting spectroscopic observations of the light from the Sun and examining the wavelengths where absorption occurs, we can learn about the substances that make up solar plasma. In addition, the Doppler effect causes shifts in the absorption lines as the solar plasma moves. By observing this shift, we can determine the velocity of the plasma in the line-of-sight direction. However, the Doppler effect only allows us to obtain the velocity parallel to the line-of-sight, and it is not possible to measure the velocity in the direction perpendicular to the line of sight with this method.

Moreover, when a magnetic field exists in the plasma, the Zeeman effect causes a shift in the polarization. By observing this polarization, we can obtain more information about the magnetic field.

Since this information is obtained by observing the light from the surface, it is only possible to know the physical quantities at the surface. Due to the opacity of the solar plasma, light from the interior is absorbed before reaching the surface. Therefore, the observed light does not directly carry information about the internal structure.

1.3.2 Local Correlation Tracking (LCT)

It is difficult to obtain information about the horizontal velocity field on the solar surface directly through observations. However, using a method called Local Correlation Tracking (LCT) (November and Simon, 1988), enables us to evaluate the flows. The LCT was originally

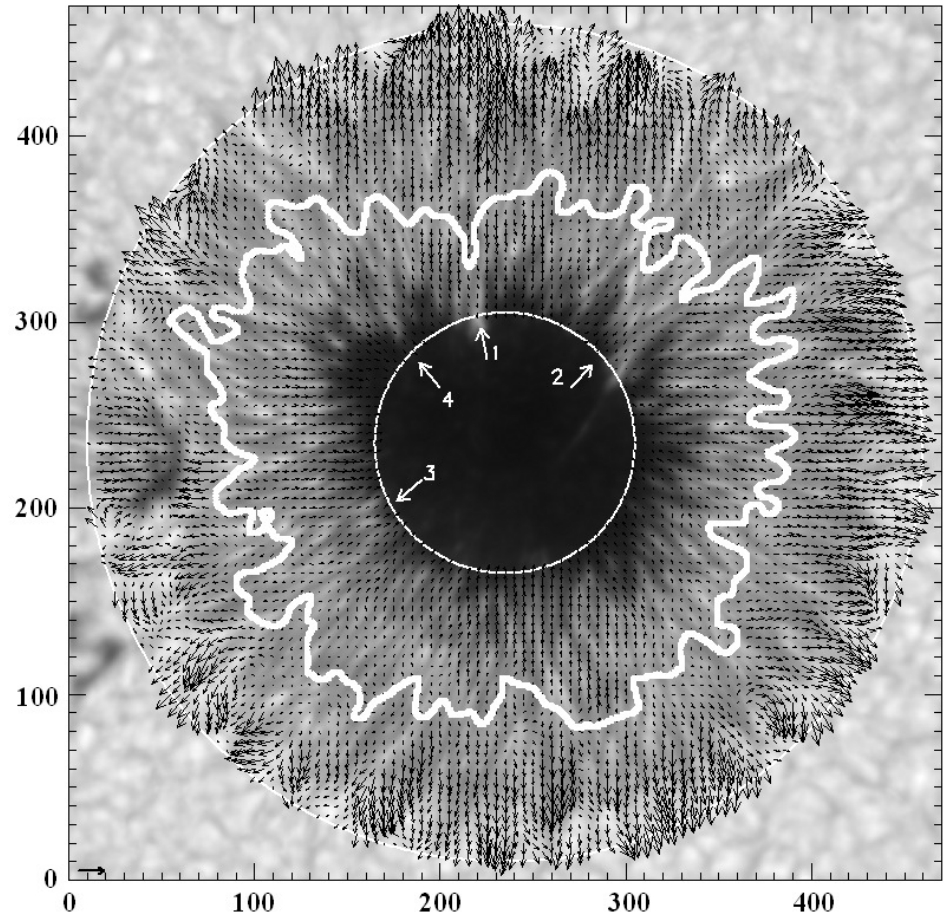


Figure1.6 The velocity field around a sunspot evaluated by The LCT. The black arrows represent the velocity field, with the sample in the lower left indicating one $\mathrm{km\,s^{-1}}$. Adapted from Figure 1 in Hamedivafa (2015).

a method developed to evaluate wind speed by tracking the movement of clouds in images taken by the telescope. By extracting characteristic structures such as clouds from multiple observation images taken at different times and calculating their displacement, the velocity field is computed by dividing the displacement by the time interval between the images. This LCT method has also been applied to evaluate the horizontal velocity field of the Sun.

Specifically, the area around the pixel of interest, where the velocity is to be determined, is extracted, and the same-sized region is extracted from temporally different images. The degree of matching between the two regions is calculated. In many cases, the cross-correlation coefficient

$$C\left(d^{(1)}(x_1,y_1),d^{(2)}(x_2,y_2)\right)=\sum_{ij}d_{ij}^{(1)}(x_1,y_1)d_{ij}^{(2)}(x_2,y_2) \tag{1.1}$$

is used. Here, $d_{ij}^{(1)}(x_1, y_1)$ and $d_{ij}^{(2)}(x_2, y_2)$ represent the ijth component of matrices extracted from regions centered on (x_1, y_1) and (x_2, y_2) in the first and second images, respectively. In addition, to avoid accidentally detecting similar structures in regions far apart, a Gaussian filter is applied to both images before extracting regions centered on the target pixel (x_1, y_1) where the velocity is to be evaluated. This calculation is repeated for various pixels (x_2, y_2), and the point with the highest cross-correlation is assumed to be the displacement from (x_1, y_1). The specific calculations performed by the code used in this study are presented in Appendix C. However, the LCT has the disadvantage of high computational cost due to image extraction, filtering, and exhaustive search.

The LCT on the solar surface tracks structures such as granulation and bright points to evaluate the velocity field. However, the LCT has certain limitations. The apparent movement does not always correspond to the actual flow velocity. For example, at the boundaries of the granulation, downflow into the solar interior converges, resulting in a dark mesh-like structure due to lower temperatures. The LCT only captures movement in the areas where this flow converges; it does not observe the convergence of the flow itself. Additionally, solar photospheric bright points are formed where strong magnetic fields converge on the surface, displacing surrounding gas due to increased magnetic pressure and exposing deeper, hotter layers. Due to the magnetic field being frozen into the plasma, these bright points move in close conjunction with the fluid flow, making them good targets for the LCT tracking. However, they only provide information about the flow at a single point, and the overall flow field cannot be determined. In Verma et al. (2013), The LCT is applied to simulation data and compared with the actual velocity. It is shown that the results of the LCT display a smoothed velocity, and the LCT's evaluation underestimates the actual velocity by approximately a factor of three.

1.3.3 Helioseismology

It is difficult to obtain direct observational information about the interior of the Sun. Therefore, a method called helioseismology exists to investigate the solar interior. Helioseismology determines the internal structure by analyzing waves that propagate inside the Sun. This concept is similar to how the internal plate structure of the Earth is investigated by analyzing the propagation of seismic waves during an earthquake. By creating a $k - \omega$ diagram (Fig. 1.7) based on observation data from the solar surface, specific oscillation modes can be identified. By solving the inverse problem of finding the velocity distribution that reproduces such a $k - \omega$ diagram, we can derive information about the velocity distribution inside the Sun. This

method, which treats the entire Sun, is called global helioseismology.

On the other hand, a method that analyzes oscillations between two points in a limited region to obtain finer details of the solar structure is called local helioseismology. In local helioseismology, the travel time of waves passing between two points is calculated by focusing on the oscillations at those points and computing their correlation. One of the local helioseismologies evaluates the internal state of the Sun by calculating the travel-time shift, which is the deviation in travel time from the model. By using local helioseismology, internal flow structures have been revealed (Gizon et al., 2020).

The basic method for calculating the travel-time shift is as follows. Denote the intensity of oscillations at time t at a location $\boldsymbol{x_i}$ in the observed image as $\phi(\boldsymbol{x_i}, t)$. The cross-correlation between two points is calculated as

$$C(\boldsymbol{x_1}, \boldsymbol{x_2}, \Delta t) = \sum_i \phi(\boldsymbol{x_1}, t_i)\phi(\boldsymbol{x_2}, t_i + \Delta t), \tag{1.2}$$

where the Δt that maximizes this value is the travel time between the two points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. If there is a deviation from the typical travel time determined by the model or the average value across the region, it indicates that there is a characteristic structure between the two points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$.

Helioseismology achieves the results, such as the detection of meridional flow (Haber et al., 2002) and the understanding of the three-dimensional structure of sunspots (Zhao and Kosovichev, 2003). Ilonidis et al. (2011) points out the possibility of detecting sunspots by examining internal structures. In DeGrave et al. (2018), a comparison is made between internal velocities derived from helioseismology and time-averaged internal velocities from simulations, showing good agreement up to a depth of 3 Mm, but noting that the evaluation deteriorates at 5 Mm due to the influence of noise.

Data spanning several hours, with measurements taken at approximately 1-minute intervals, are used to analyze the internal structure to calculate the travel-time shift. As a result, the data volume is large, which limits the immediacy of the results. Furthermore, since the correlation is analyzed using several hours of data, the obtained information represents an average over that period. If there are changes in other physical quantities, further calculations based on different theories are required.

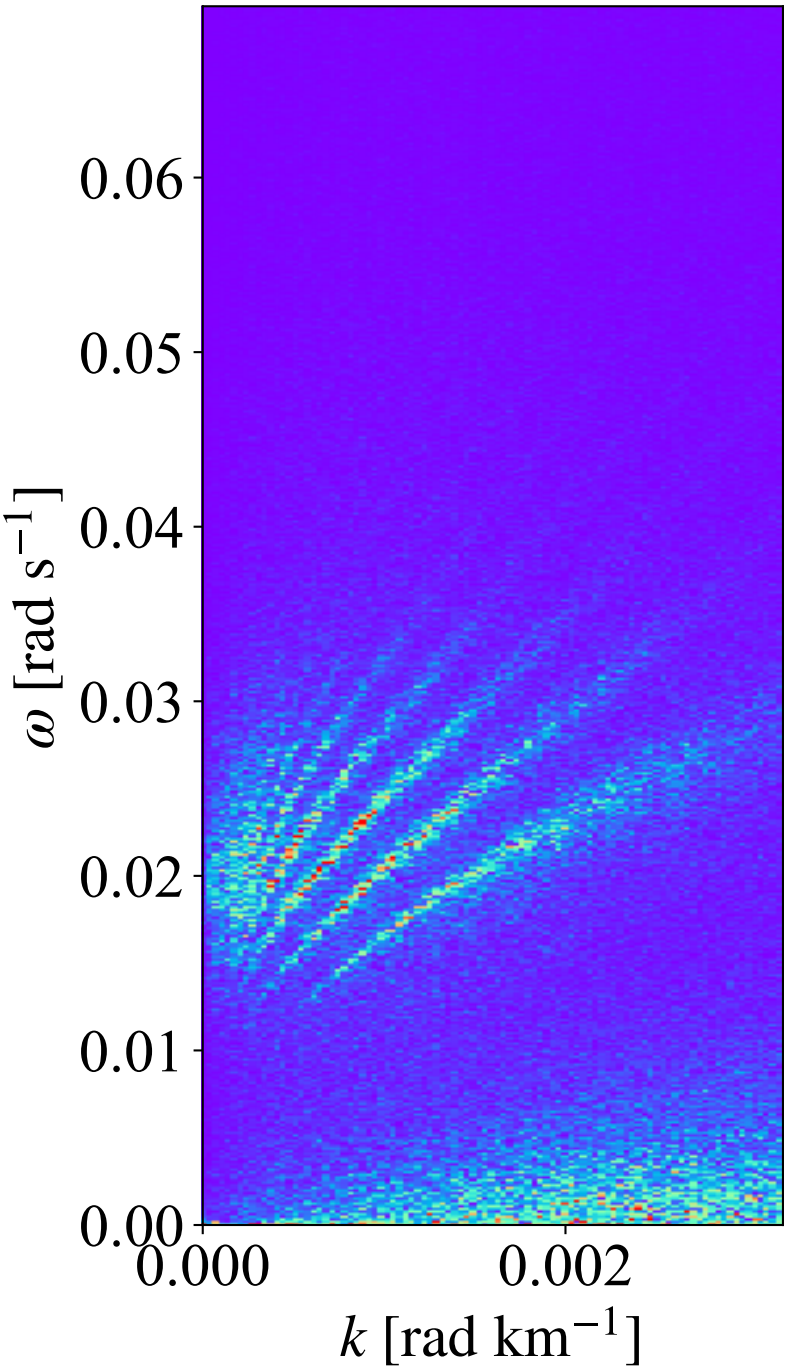


Figure1.7 The distribution of wavenumber k and frequency ω for one day of velocity field observations by SDO/HMI. The intrinsic oscillations of the solar surface are visible.

1.4 Neural Networks

Neural networks are mathematical models that simulate the process of information processing in the human brain and can be applied to a wide range of problems, not limited to the field of physics. The human brain is composed of cells called neurons. Neurons are connected to adjacent neurons and exchange signals, forming a network structure. When a neuron receives a signal from neighboring neurons that exceeds a certain threshold, it also sends out a signal, causing a chain reaction of signal transmission. This structure is mathematically modeled as what is called a neural network.

A schematic diagram of a neural network is shown in Figure 1.8.

The units that simulate the neurons in a neural network are called nodes, and the signal y received by a node can be written as

$$y = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b \tag{1.3}$$

where x_i is the output from the ith adjacent node, w_i is the weight that determines the influence of the output from the adjacent nodes, and b is the bias that determines the final magnitude of the input, representing the node's influence on the network. Based on the value of y, the signal to the next node is determined. In the past, step functions were used to determine the final output x, such that if y is zero or less, the output is 0, and if y is greater than 0, the output

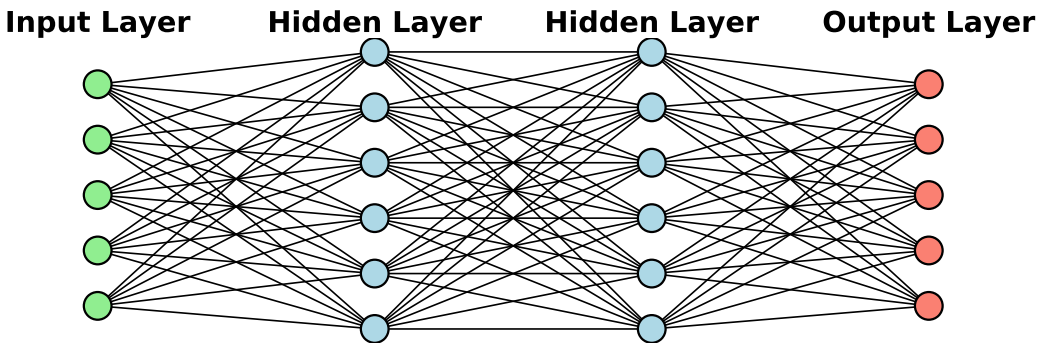


Figure1.8 Schematic diagram of a network. The circles represent nodes, and they are connected layer by layer. By changing the strength of these connections, various functions can be reproduced.

is 1:

$$x = \begin{cases} 1 & (y \geq 0) \\ 0 & (y < 0) \end{cases} \tag{1.4}$$

This process is used to model neuron behavior in early works (Rosenblatt, 1958; Rumelhart, 1986). Applying this process to multiple nodes can be written as a recurrence relation using matrix and vector notation:

$$\boldsymbol{y}^{(n)} = \boldsymbol{w}x^{(n-1)} + \boldsymbol{b} \tag{1.5}$$

$$\boldsymbol{x}^{(n)} = f(\boldsymbol{y}^{(n)}) \tag{1.6}$$

where $\boldsymbol{x}$ is the vector of outputs from all adjacent nodes, $\boldsymbol{w}$ is the weight matrix corresponding to each element of $\boldsymbol{x}$ and $\boldsymbol{y}$, and $\boldsymbol{b}$ is the bias vector. The function f is called an activation function, which makes the network nonlinear. Although step functions were used in the past as activation functions, they made it difficult to adjust the network parameters because the output would not change despite parameter adjustments.

For this reason, the ReLU (Rectified Linear Unit) function, given by

$$x = \begin{cases} y & (y \geq 0) \\ 0 & (y < 0) \end{cases} \tag{1.7}$$

is now often used (Lecun et al., 2015), although the appropriate activation function should be chosen depending on the problem. The network has a large number of parameters, including the weight matrices $\boldsymbol{w}$ and the bias vectors $\boldsymbol{b}$ for each layer n. This large number of parameters allows the network to have expressive capability power. Deep learning is a method for automatically determining these parameters. Since neural networks are mostly composed of simple additions and multiplications, their computations and parameter optimization can be efficiently performed in parallel.

1.5 Purposes of this study

As explained so far, observations of the solar surface are crucial for solving various problems in solar physics. However, there are many physical quantities that are difficult to obtain through pure observation alone. On the other hand, magnetohydrodynamic (MHD) simulations (See Section 2.2.) have significantly advanced in recent years (Stein and Nordlund, 1998; Vögler et al., 2005), and with improvements in computer performance and new algorithms, it is now possible to perform highly accurate numerical simulations of the Sun.

However, due to the turbulent nature of the solar interior, the state of the Sun at a given moment does not perfectly match simulation data or observational data taken at different positions or times. Therefore, it is not possible to determine unobservable physical quantities or predict future phenomena purely by comparing observational data with simulations.

This study aims to use the high evaluation capabilities of neural networks to associate the observed solar conditions with those simulated by MHD models, thereby evaluating the current solar state that cannot be strictly reproduced by simulations. We will use neural networks to evaluate the instantaneous horizontal velocity and internal upward velocity fields, which are difficult to observe, by correlating them with easily observable physical quantities such as radiation intensity, line-of-sight velocity field, and line-of-sight magnetic field.

If the approximate locations of internal upflows can be identified, it may be possible to narrow down the candidate regions for sunspot emergence. As a first step toward the future application of space weather forecasting, the evaluation of the solar convective structures is conducted.

1.6 Previous Research

There are already methods for evaluating horizontal velocity fields using neural networks and numerical simulations (Asensio Ramos et al., 2017; Tremblay and Attie, 2020; Ishikawa et al., 2021). DeepVel, proposed by Asensio Ramos et al. (2017), was developed based on the idea that scanning the entire image with the LCT-like filter is similar to the convolutional neural network approach. Therefore, similar to the LCT, DeepVel evaluates the horizontal velocity field using two intensity images taken at different times. In addition to the velocity field on the optical depth $\tau = 1$, which is close to the surface actually observed, DeepVel also evaluates the velocity fields on the layers above this, at $\tau = 0.1$ and $\tau = 0.01$. Figure 1.9 shows the evaluation results from DeepVel. The top row shows the horizontal velocity field obtained from numerical simulations, and the bottom row shows the horizontal velocity field evaluated by DeepVel. The two appear to be in reasonable agreement. The correlation coefficients between the two fields are 0.83 at $\tau = 1$, 0.86 at $\tau = 0.1$, and 0.78 at $\tau = 0.01$.

DeepVelU, an improved version of DeepVel (Tremblay and Attie, 2020), can evaluate the horizontal velocity field using not only intensity but also line-of-sight velocity and line-of-sight magnetic field. The correlation coefficient between the simulation data and the evaluated horizontal velocity field at $\tau = 1$ reaches as high as 0.947.

These methods are considered superior to other methods for evaluating horizontal flows, such as LCT (Tremblay et al., 2018). In Ishikawa et al. (2022), a deep learning method

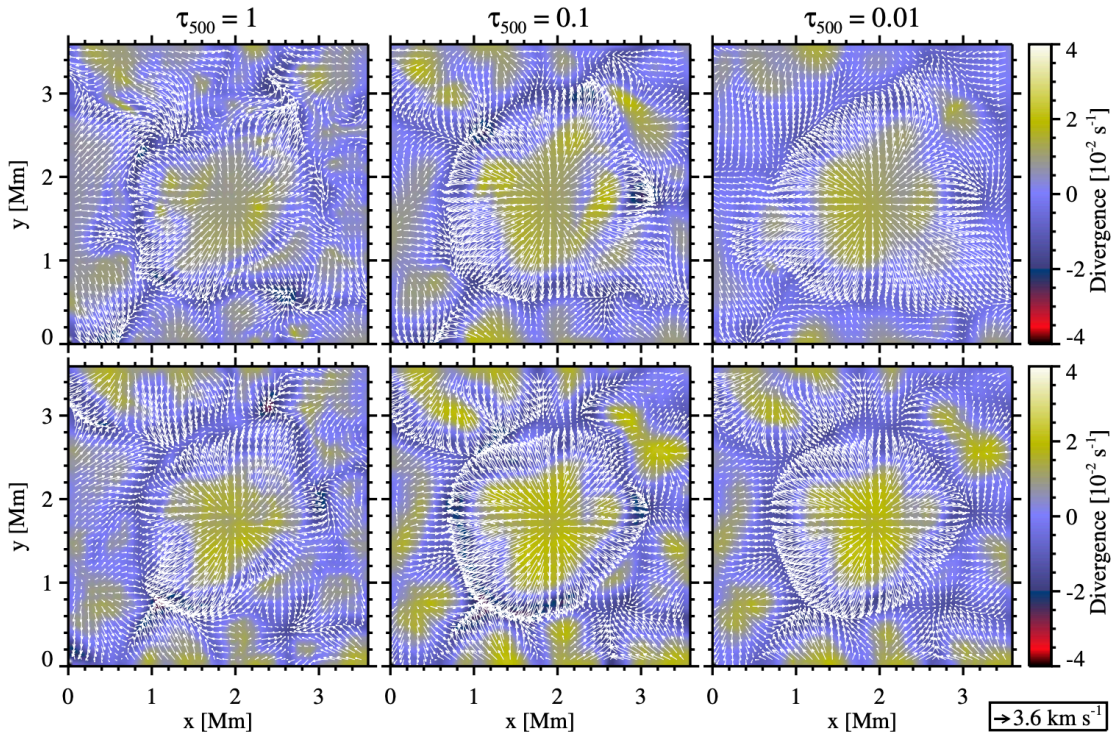


Figure1.9 Evaluation results of the horizontal velocity field at each optical thickness using DeepVel. The top row shows the horizontal velocity field prepared using numerical simulations, and the bottom row shows the horizontal velocity field evaluated by DeepVel. The white arrows represent the horizontal velocity field, and the background colormap indicates the horizontal divergence of the velocity field. Quoted from Fig. 2. of (Asensio Ramos et al., 2017).

is proposed to improve the evaluation performance for smaller spatial scales, which are challenging for machine learning methods, by applying multiple convolutional structures in parallel.

However, these methods require two image data points taken 30 seconds apart, making them unsuitable when the time interval of the observational data is not available. Furthermore, they cannot handle cases where there are gaps in the observational data. This is particularly problematic for ground-based observations, where changing atmospheric conditions can often prevent continuous observation. In addition, researchers may also increase the time interval depending on the project objectives due to the amount of communication data. Therefore, it is not always possible to apply networks that require two images for every observation. If horizontal velocity could be evaluated from a single snapshot, its applicability would be

broader.

This study aims to develop a method for evaluating the horizontal velocity field of the solar surface from a single snapshot. This content is described in Chapter 3 and is based on Masaki et al. (2023)

Moreover, while these studies focus on evaluating the flow field on the solar surface, there is no research utilizing machine learning to investigate the internal structure of the Sun. This content is described in part of Chapter 4 and is based on Masaki and Hotta (2024)

Chapter 2

Method

2.1 Research Outline

This research proceeds as follows. Figure 2.1 illustrates the flow of this study.

(1) Perform numerical simulations that reproduce the solar surface to prepare data for training the network.

(2) Construct a neural network using the data prepared in (1), with inputs such as intensity and vertical magnetic field and outputs such as the velocity field. The network is optimized using extensive numerical computations.

(3) Validate and apply the network trained in (2) by using different simulation or observation data from the training data. (Chapters from 3.3 onward)

Although the process generally follows this order, it may revert to previous steps depending on the results of network training.

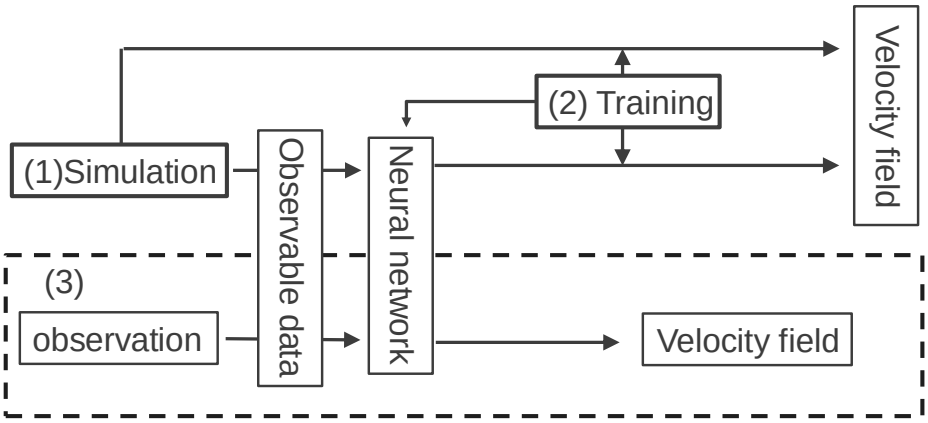


Figure2.1 Flow of this study. It proceeds in the order of (1), (2), and (3).

2.2 Numerical Simulation

The simulations in this research are conducted using the radiation MHD simulation code R2D2 (Radiation and RSST for Deep Dynamics: Hotta et al., 2019; Hotta and Iijima, 2020; Hotta and Toriumi, 2020). The following magnetohydrodynamic (MHD) equations are solved in the simulation: Continuity equation

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \boldsymbol{v}) \tag{2.1}$$

Equation of motion

$$\frac{\partial}{\partial t}(\rho \boldsymbol{v}) = -\nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v}) - \nabla p + \rho \boldsymbol{g} + \frac{1}{4\pi}(\nabla \times \boldsymbol{B}) \times \boldsymbol{B} \tag{2.2}$$

Induction equation

$$\frac{\partial \boldsymbol{B}}{\partial t} = \nabla \times (\boldsymbol{v} \times \boldsymbol{B}) \tag{2.3}$$

Energy equation

$$\rho T \frac{\partial s}{\partial t} = \rho T \left(\boldsymbol{v} \cdot \nabla\right) s + Q \tag{2.4}$$

Equation of state

$$p = p\left(\rho,\ s\right) \tag{2.5}$$

Here, ρ, $\boldsymbol{v}$, p, T, $\boldsymbol{g}$, $\boldsymbol{B}$, s, and Q represent density, velocity, pressure, temperature, gravitational acceleration, magnetic field, entropy, and radiative heating, respectively. R2D2 solves these equations using fourth-order spatial derivatives and fourth-order Runge-Kutta time integration (see Appendix A). Pressure is obtained from the entropy and density using a table prepared from the OPAL equation of state (Rogers et al., 1996), which considers partial ionization.

Radiative heating Q is calculated by solving the radiative transfer equation in the gray approximation. The radiative transfer equation is expressed as

$$\frac{\partial I}{\partial \tau} = S - I \tag{2.6}$$

where I, τ, and S represent the intensity, optical depth, and source function, respectively. The optical depth τ is defined using the Rosseland mean opacity κ_{R} (Seaton et al., 1994) as follows

$$d\tau = \rho \kappa_{\mathrm{R}} ds \tag{2.7}$$

Radiative heating is calculated in optically thick regions as

$$\boldsymbol{F} = \int I(\boldsymbol{\mu}) \boldsymbol{\mu} d\omega \tag{2.8}$$

$$Q_F = -\nabla \cdot \boldsymbol{F} \tag{2.9}$$

In optically thin regions, it is calculated as

$$J = \frac{1}{4\pi} \int I d\omega \tag{2.10}$$

$$Q_J = 4\pi\rho\kappa \left(J - S\right) \tag{2.11}$$

where $\boldsymbol{\mu}$ is the unit vector in the direction of radiation, κ is opacity, and ω is solid angle. In R2D2, the heating in optically thick and thin regions is switched by calculating

$$Q = \exp\left(-\frac{\tau}{\tau_0}\right) Q_J + \left[1 - \exp\left(-\frac{\tau}{\tau_0}\right)\right] Q_F \tag{2.12}$$

using $\tau_0 = 0.1$ as the threshold.

As an initial condition, the solar standard model (Christensen-Dalsgaard et al., 1996, Model S), modified to include entropy, is used. (See Hotta and Iijima (2020), Appendix A). The upward flow is set according to the entropy values obtained from Model S so that the radiative flux from the solar surface, known from observations as $F_\odot = 6.34 \times 10^{10}\,\mathrm{erg\,cm^{-2}\,s^{-1}}$, is naturally satisfied.

Other individual simulation settings are described in Sections 3.1 and 4.1.

2.3 Neural Network Structure

2.3.1 Convolutional Neural Networks

The network structure can generally be expressed by Equation 1.6. In this structure, the output from a node to the next layer is determined based on signals from all input nodes. However, when dealing with certain types of data, such as images or audio, it is not necessary to determine the output based on all inputs. For example, in the task of extracting text from audio data of human speech, it is not necessary for the network to interpret sounds that are several tens of seconds apart when determining a single character spoken at a specific moment. In such cases, there may be unnecessary parameters in the network. Additionally, in the basic structure, different network parameters are used to determine one character at a certain time and another character at a different time. If the same parameters could be used, unnecessary parameters would be reduced.

To address this, convolutional neural networks (CNNs) were developed, inspired by the mechanism through which humans process sounds or images (Fukushima, 1980). When humans interpret such information, they gather partial information from a point and its surroundings and then obtain the overall information by shifting this process temporally or spatially. Convolutional neural networks simulate this process by creating filters that pass information from a limited area around a point to the next node, shifting these filters by a set number of steps. A schematic diagram of this process is shown in Figure 2.2. The filter is

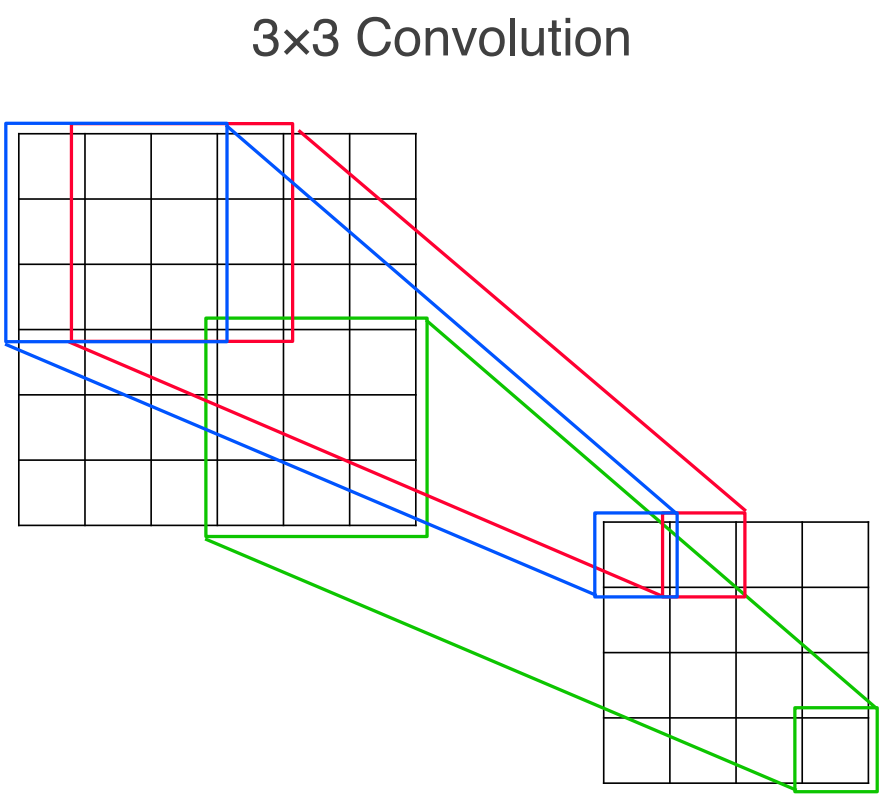


Figure2.2 Schematic of a 3×3 convolution on an image. The value of one pixel in the output image (right) is determined from a 3×3 pixel region in the input image (left). This process is repeated by shifting one pixel at a time to determine the values for the entire output.

adjusted using machine learning to achieve the desired performance.

Let x and y denote the input and output images, respectively, and let x_{kl} and y_{kl} represent the pixel values at position (k, l) in x and y. This operation can be expressed by the equation:

$$y_{kl} = \sum_{i,j} w_{ij} x_{k+i,l+j} \tag{2.13}$$

If a 3×3 filter is used then $i, j = -1, 0, 1$.

Convolutional neural networks significantly reduce the number of network parameters, improving both computation speed and learning speed. Additionally, because CNNs create filters for images, they can be applied regardless of the size of the input data.

2.3.2 U-net

Neural networks are composed of a combination of several operations, including convolution. The network architecture used in this study is based on U-net (Ronneberger et al., 2015). This structure was originally developed for the medical field to detect regions in microscope images that meet specific conditions, such as the presence of certain cells. It has since been applied to a wide range of image-processing problems across various fields. The name "U-net" comes from the U-shape of the network structure. U-net essentially follows an encoder-decoder (compression-decompression) structure. First, the encoder compresses the image to extract the necessary information for evaluation. Then, the decoder restores the compressed information to the original image size. However, one drawback of this encoder-decoder structure is that compressing the image can lead to the loss of positional information in the original image. To solve this, U-net uses skip connections, which combine the original image with the restored image from the decoder before compressing the image.

The specific network structure is different for each chapter and is described in 3.2 and 4.2.

The processing and parameter optimization methods used in this study are described in Appendix B.

Chapter 3

Evaluation of Surface Velocity Field

This chapter explains the neural network designed to evaluate horizontal velocity fields on the solar surface using data such as radiation intensity.

3.1 Simulation Setting

We run numerical simulations to prepare data for machine learning. The computational domain is set to 6.144 Mm $\times$ 6.144 Mm in the horizontal direction to include multiple granulations, which are characteristic structures of thermal convection and have a typical size of approximately 1 Mm. In the vertical direction, the domain size is 700 km above the solar surface, where the optical thickness is $\tau = 1$, to 2.3 Mm below it, to capture the thermal convection near the solar surface. The grid spacing is set to 48 km in the horizontal direction and 24 km in the vertical direction to resolve the granulation to some extent, resulting in a total of 128^3 grid points.

The total magnetic flux in the simulation domain remains constant due to magnetic flux conservation, making the initial magnetic flux a free parameter. To ensure the generality of the training data for the neural network, simulations are conducted with several initial magnetic field strengths (magnetic flux amounts). This allows for adjustment of the number of bright points, which are areas of strong magnetic fields. The initial magnetic field is set to 1 G, 20 G, and 30 G in the upward direction from the bottom of the computational domain. The total simulation time is equivalent to 100 days, which is used as training data. Additionally, three days of simulations with an initial magnetic field of 10 G are used as evaluation data for the network. The radiation intensity used is calculated by solving the radiative transfer equations with 24 rays, and the values at the top of the computational domain are used. Other physical quantities are taken from the $\tau = 1$ surface, which is defined by the Rosseland mean opacity.

Since the sound speed in the solar photosphere is approximately ten $\mathrm{km\,s^{-1}}$ and the maximum speed of thermal convection is approximately 30 $\mathrm{km\,s^{-1}}$, the CFL condition gives a time step Δt of about 1 second, calculated as 48 (km) / 40 ($\mathrm{km\,s^{-1}}$). A total of 9 million time steps is required to compute 100 days of simulation.

The simulation data output is taken at 5-minute intervals. While reducing this interval would allow for more computational data, the typical lifetime of granulation is several minutes. Therefore, if the time interval is shorter than 5 minutes, the changes in the data are minimal, and the learning efficiency does not improve significantly. For this reason, a 5-minute interval is chosen to reduce the time required for network training. This resulted in obtaining a total of 30,000 data sets for training, 1,000 for validation, and 1,000 for evaluation.

3.2 Training Setup

The inputs to the network are physical quantities that are relatively easy to observe, such as intensity, vertical magnetic field, and vertical velocity field. To compare the effect of inputs on evaluation accuracy, multiple networks are created by combining these three inputs in several combinations (see Section 3.5). We create the networks that output the two components of the horizontal velocity field corresponding to the input physical quantities. In this study, multiple networks with different inputs are created, but they all share the same architecture. This structure is shown in Figure 3.1.

In Figure 3.1, the left half represents the encoder. For the input image, two convolutions with a kernel size of 3 $\times$ 3 are performed, followed by a 2 $\times$ 2 convolution with a stride of 1 pixel, halving the image size in both directions. This process is repeated twice, reducing the image size to one-eighth in the encoder. In the right half, representing the decoder, the same structure is used but in reverse, where the image is enlarged by converting 1 pixel of information into a 2 $\times$ 2 block. Additionally, a residual block (Figure 3.1 b) is placed between the encoder and decoder to organize the compressed information from the encoder.

The training data used in this study consists of three types of inputs: intensity, vertical magnetic field, and vertical velocity field. The outputs are the horizontal velocity field, and All of these data are represented as 128 $\times$ 128 grid point images. To augment the training data, the 30,000 sets of simulation data are rotated by 90 degrees, resulting in a total of 120,000 data sets. To evaluate the performance of the trained network, an additional 1,000 data sets are used. Each training batch consists of 32 data sets.

For each batch of 32 data sets, the data are normalized by subtracting the mean and dividing by the standard deviation so that the standard deviation is one and the mean is 0.

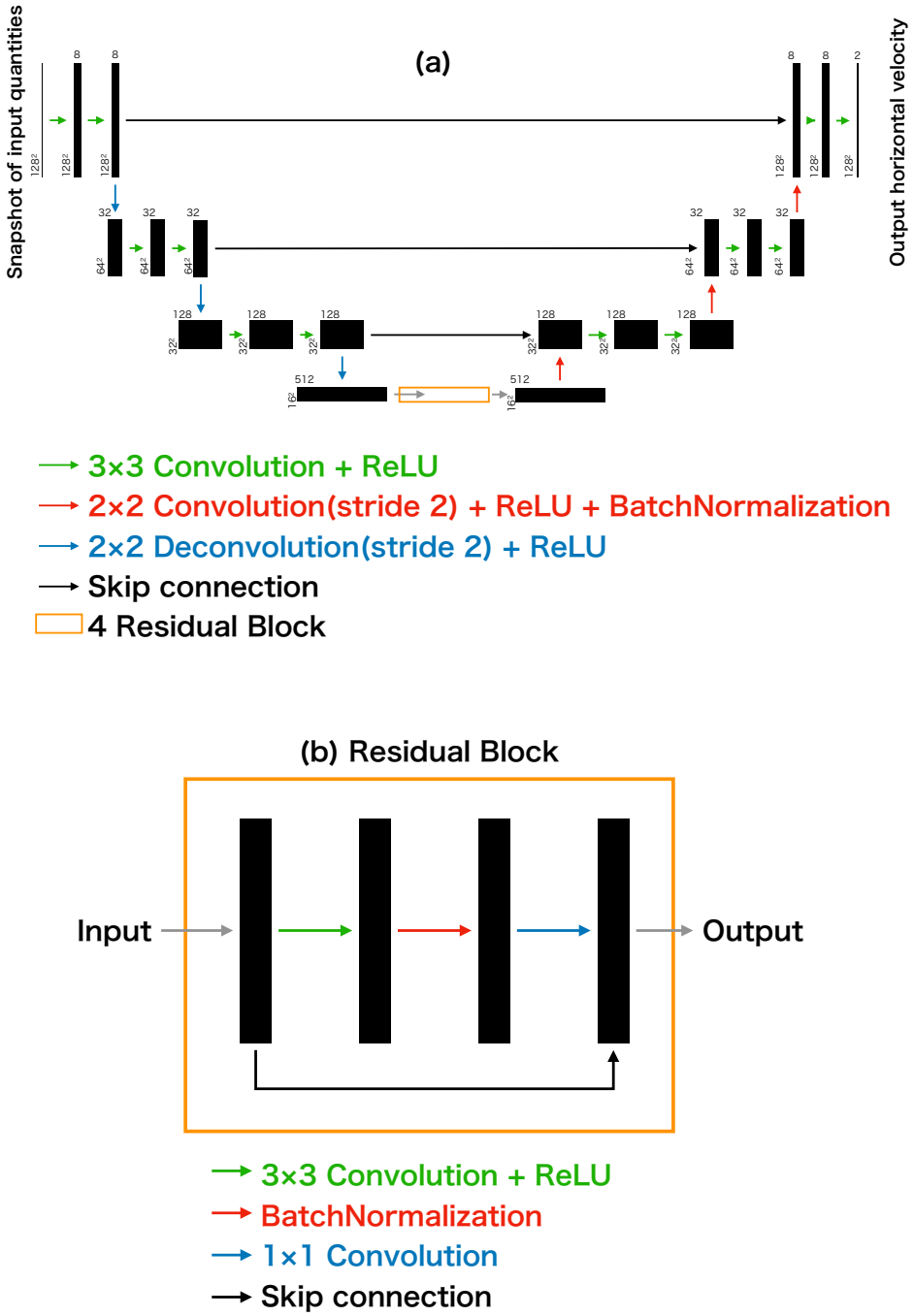


Figure3.1 Diagram of the network structure. Panel (a) shows the overall network architecture. Panel (b) is an enlargement of the residual block, highlighted in orange in the overall Diagram. The vertical numbers in each box represent the image size, while the horizontal numbers represent the number of images. The data size remains unchanged in the residual block across all layers.

Training is conducted by learning all the data for 128 epochs in random order. The network with the smallest error, as evaluated using the validation data, is adopted as the final model. The mean squared error is used as the loss function, and Adam (Kingma and Ba, 2014) is used as the optimizer. Details of Adam are explained in Appendix B.

The implementation is done using Keras and TensorFlow, and training is performed using an Nvidia GeForce RTX 2080 Ti GPU.

3.3 Evaluation Results

The physical quantities input into the network are shown in Figure 3.2. Panels (a), (b), (c), (d), (e), and (f) in the figure correspond to the intensity, vertical velocity field, vertical magnetic field, horizontal velocity field from the simulation, horizontal velocity field evaluated using only intensity, and horizontal velocity field evaluated using intensity and vertical velocity, respectively. The white arrows in the velocity field indicate the velocity, and the background shows the horizontal divergence. From these results, it appears that the network is able to realistically reproduce the simulated velocity field it is trained to evaluate. Additionally, the vortex at $(x, y) = (4\ \mathrm{Mm},\ 4\ \mathrm{Mm})$ is also well reproduced.

Next, the performance of the network is evaluated statistically. The following five metrics are used for evaluation:

- Correlation coefficient (CC)

$$\frac{\sum(v_{\mathrm{sim}} - \bar{v}_{\mathrm{sim}})(v_{\mathrm{eva}} - \bar{v}_{\mathrm{eva}})}{\sqrt{\sum(v_{\mathrm{sim}} - \bar{v}_{\mathrm{sim}})^2}\sqrt{\sum(v_{\mathrm{eva}} - \bar{v}_{\mathrm{eva}})^2}}, \tag{3.1}$$

- R2 score

$$1 - \frac{\sum(v_{\mathrm{sim}} - v_{\mathrm{eva}})^2}{\sum(v_{\mathrm{sim}} - \bar{v}_{\mathrm{sim}})^2} \tag{3.2}$$

- Mean squared error (MSE)

$$\overline{(\vec{v}_{\mathrm{sim}} - \vec{v}_{\mathrm{eva}})^2} \tag{3.3}$$

- Mean absolute error (MAE)

$$\overline{|\vec{v}_{\mathrm{sim}} - \vec{v}_{\mathrm{eva}}|} \tag{3.4}$$

- Mean angular difference

$$\theta = \arccos\left(\frac{\boldsymbol{v}_{\mathrm{eva}} \cdot \boldsymbol{v}_{\mathrm{sim}}}{|\boldsymbol{v}_{\mathrm{eva}}||\boldsymbol{v}_{\mathrm{sim}}|}\right) \tag{3.5}$$

Here, v_{sim} represents the values from the simulation data, and v_{eva} represents the values evaluated by the network. The overbars indicate averages over the entire dataset and pixels.

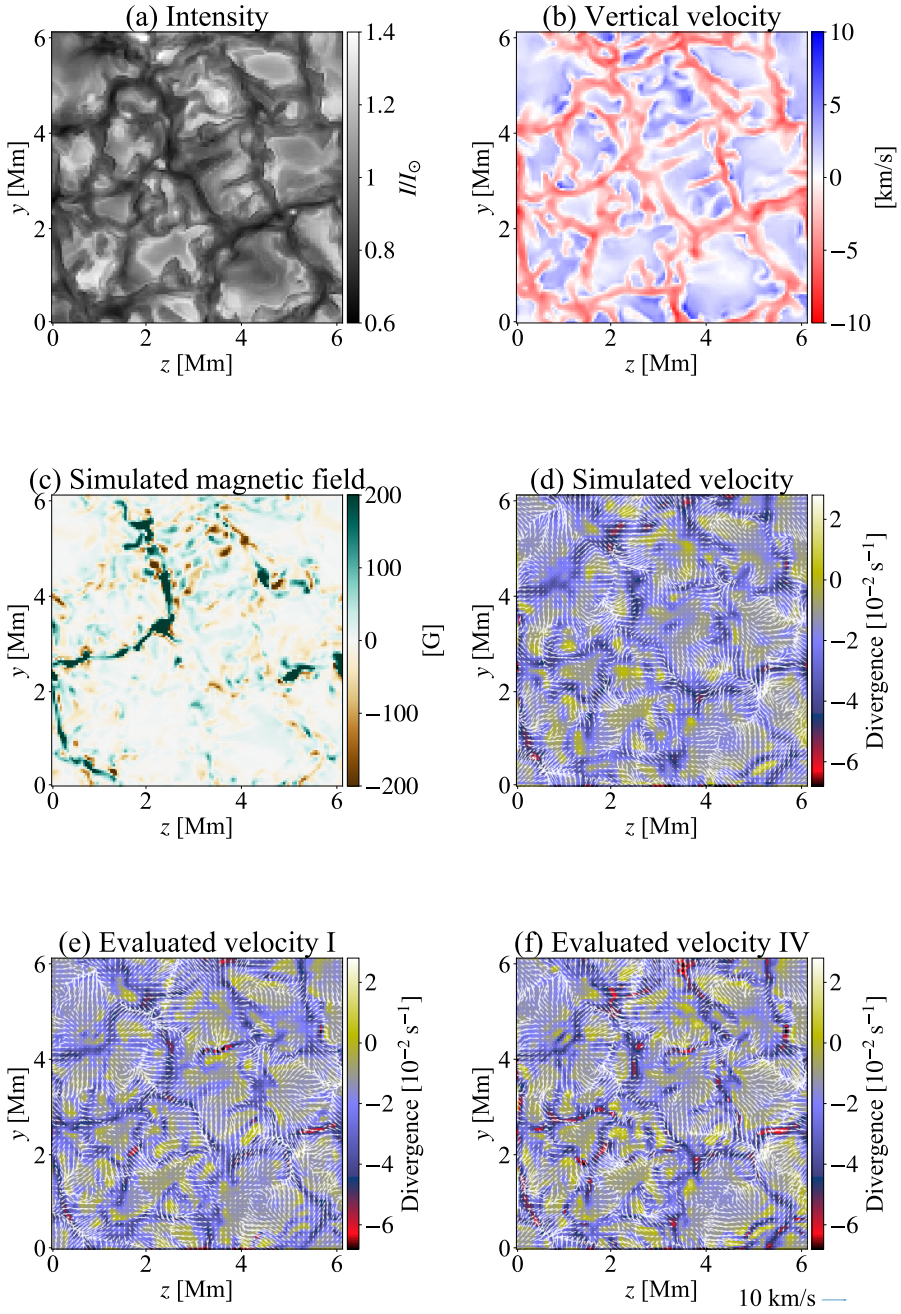


Figure3.2 The figure shows the input and output when evaluating the horizontal velocity field. Panels (a) intensity, (b) Vertical velocity field, (c) Vertical magnetic field, (d) Horizontal velocity field, (e) Horizontal velocity field evaluated using only intensity, (f) Horizontal velocity field evaluated using intensity and vertical velocity. In Panels (d), (e), and (f), the background shows the horizontal divergence of the velocity field, and the white arrows represent the velocity field. The legend for the arrows is displayed in the bottom right corner.

CC	R2 Score	MSE $(\mathrm{km\ s^{-1}})^2$	MAE $(\mathrm{km\ s^{-1}})$	Mean Angular Difference
0.83	0.69	1.29	0.96	29.7°

Table3.1 Evaluation metrics for horizontal velocity field evaluation using only intensity.

For the correlation coefficient and R2 score, values closer to 1 indicate better performance, while for the other metrics, values closer to 0 indicate better performance. Table 3.1 shows the results of evaluation using only intensity. The typical convective velocity of granulation is 3-4 $\mathrm{km\ s^{-1}}$, and since the error is around one $\mathrm{km\ s^{-1}}$, this corresponds to an approximate error of 30%.

Figure 3.3 shows the distribution of absolute errors between the horizontal velocity field from the simulation and the velocity field evaluated by the network. Panels (a) and (b) show the errors for the evaluation using only intensity and the evaluation using both intensity and vertical velocity, respectively. The areas with larger errors are located at the boundaries of the granulation, where the intensity is lower. In contrast, within the bright regions of the granulation, the errors appear to be smaller. This is likely because the dark regions of the granulation contain downward flows of cooler plasma, resulting in more complex flows. The figure also shows that using line-of-sight velocity for evaluation reduces the error over more area. Figure 3.4 shows the 2D histogram of the velocities from the simulation and the network. Panels (a) and (b) show the results when evaluated using only intensity and when using intensity and vertical velocity. Most points lie along the line $v_{\mathrm{sim}} = v_{\mathrm{eva}}$, but it can be seen that in regions of high velocity, the network tends to underestimate the values. The approximation lines and the average line for each simulated velocity show that when the network makes evaluation errors in the velocity field, it tends to underestimate the velocity.

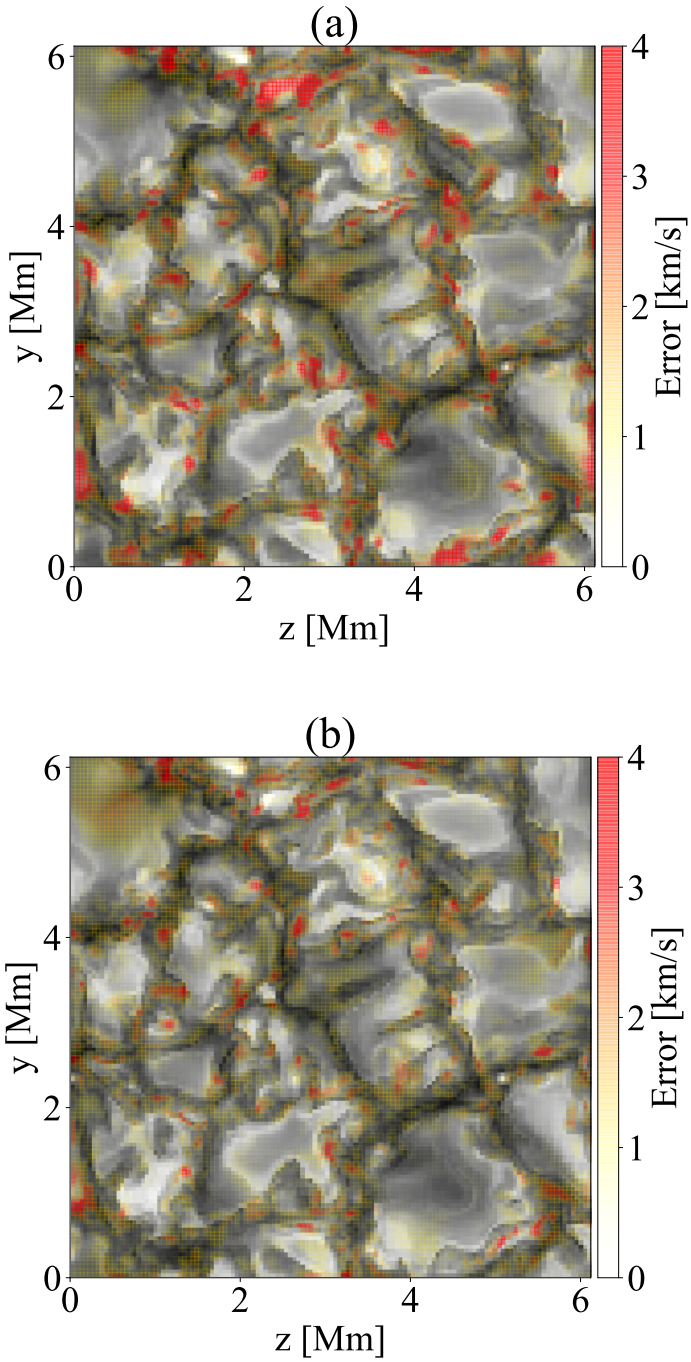


Figure3.3 Panels (a) and (b) show the distribution of errors between the evaluated velocity field and the simulation velocity field. Panel (a) shows the evaluation result using only intensity, while Panel (b) shows the evaluation result using intensity and vertical velocity field. The background represents the input intensity.

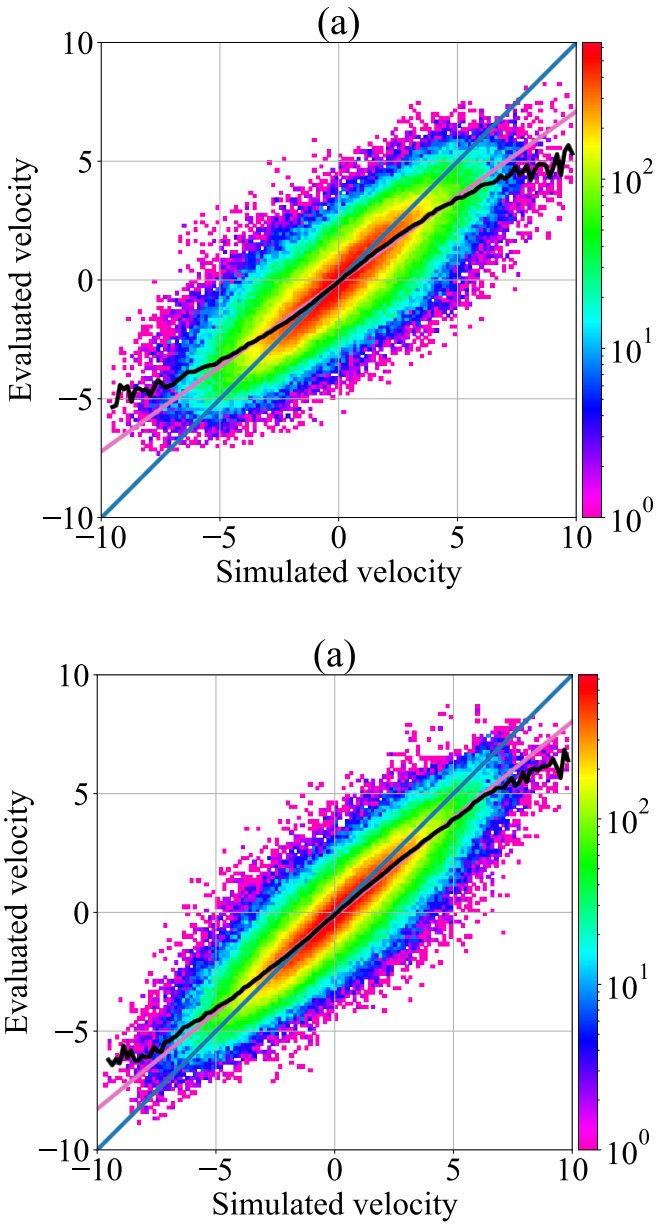


Figure3.4 The figure shows the 2D histograms of the velocity field evaluated by the network and the velocity field from the simulation. Panels (a) and (b) show the errors when evaluated using only intensity and when using intensity and vertical velocity, respectively. The color map indicates the number of pixels corresponding to each value. The blue line represents where the two values are equal, $v_{\mathrm{sim}} = v_{\mathrm{eva}}$. The pink line is the approximation line for both. In Panel (a), the approximation is $v_{\mathrm{eva}} = 0.710 v_{\mathrm{sim}} - 0.076$, while in Panel (b), it is $v_{\mathrm{eva}} = 0.813 v_{\mathrm{sim}} - 0.098$. The black line indicates the average evaluated velocity for each simulated velocity.

3.4 Dependence of Network Performance on Data Amount

When the training data lacks generality, the network overly adapts the training data, a phenomenon known as overfitting, which results in poor performance when used in real problems. However, there is no systematic way to evaluate the generality of training data before training, and the extent of overfitting can only be confirmed by actual training. Here, we compare the performance by training the network using 3,000, 30,000, and 120,000 datasets. The results of the training are shown in Table 3.2.

Figure 3.5 shows the learning curve for each amount of training data, indicating the evaluation values at each stage of learning. These results suggest that increasing the amount of data reduces overfitting in the network. Moreover, the performance improvement from 3,000 to 30,000 sets is more significant than the improvement from 30,000 to 120,000 sets, indicating that further increasing the data beyond 120,000 is likely to not dramatically enhance performance.

Number of Training Data	3,000	30,000	120,000
Correlation Coefficient	0.70	0.80	0.83
R2 Score	0.42	0.63	0.69
Mean Squared Error $[(\mathrm{km\,s^{-1}})^2]$	1.76	1.41	1.29
Mean Absolute Error $[\mathrm{km\,s^{-1}}]$	1.36	1.06	0.96
Mean Angular Difference [degree]	$45.0°$	$33.7°$	$29.7°$

Table3.2 Evaluation metrics for each amount of training data used. All results are for the horizontal velocity field evaluated using only intensity.

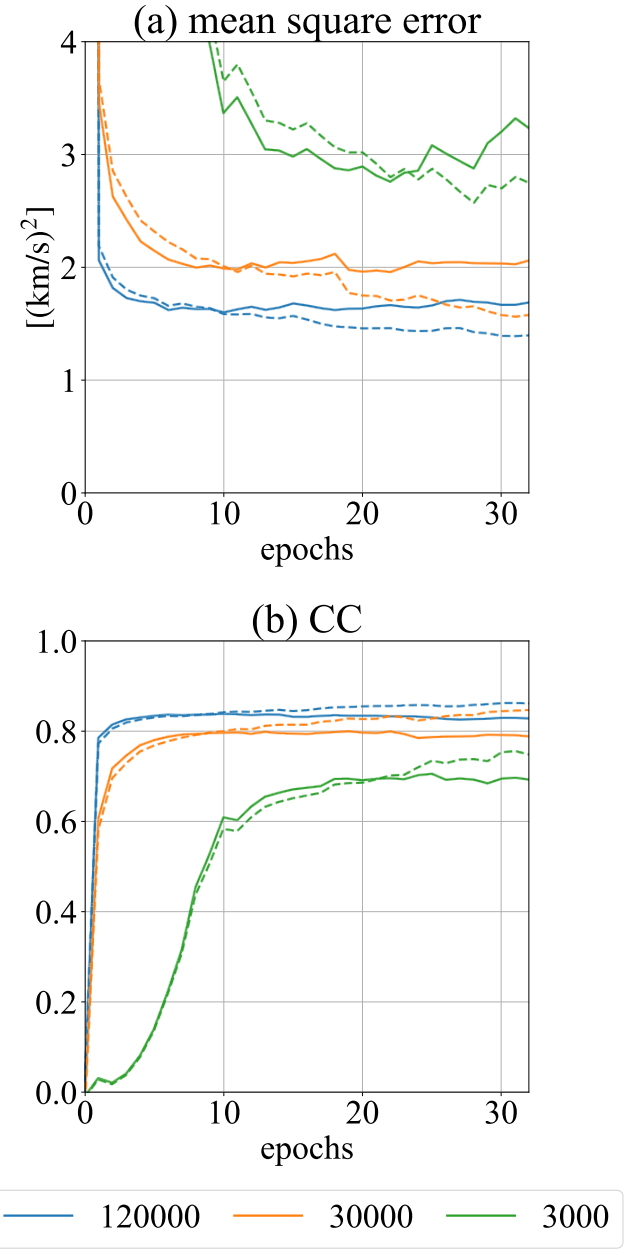


Figure3.5 The learning curve for each amount of training data, showing the evaluation metrics at each stage of learning. Panel (a) shows the mean absolute error, and Panel (b) shows the correlation coefficient. The solid lines represent the errors for the validation data, and the dashed lines represent the errors for the training data. The horizontal axis represents the number of epochs, which corresponds to the number of times the entire dataset is learned rather than the number of network updates. As learning progresses, the gap between the errors in the training data and the validation data increases, indicating the occurrence of overfitting.

3.5 Dependence of Network Performance on Input Quantities

In the previous section, we discussed the evaluation using only intensity and the evaluation using both intensity and vertical velocity. Here, we further examine how the input physical quantities, including the vertical magnetic field observable through the Zeeman effect, affect the evaluation of the horizontal velocity field. However, since intensity is the most easily and highly resolved observable quantity, it is used in all network evaluations. The number of input data channels is increased to 3, and unused data is masked with 0. All other settings are the same as in the previous section, with 120,000 sets of data. Examples of the intensity, vertical velocity field, and vertical magnetic field used are shown in Figure 3.2.

The vertical magnetic field has a typical value of a few G in most regions but can exceed 1,000 G in some areas, which indicates large kurtosis. Since such structures are challenging to learn, we processed the data using the following equation, which effectively compresses the magnetic field:

$$B' = \frac{B}{|B|} \log\left(1 + \frac{B}{B_{\rm cr}}\right), \tag{3.6}$$

where $B_{\rm cr}$ is the reference magnetic field, set to $B_{\rm cr} = 1$ G in this study. This allows weaker magnetic field structures in quiet regions to also influence learning.

The evaluation metrics for each input combination are shown in Table 3.3, and the learning curves for each input combination are shown in Figure 3.6. The results indicate that the evaluation performance improves when velocity is included. On the other hand, adding the magnetic field does not significantly enhance performance. This is likely because the vertical velocity field is strongly related to the horizontal velocity field and provides additional information necessary for evaluation. In contrast, the information provided by the vertical magnetic field is already included in either the intensity or vertical velocity field, making it less effective in the evaluation.

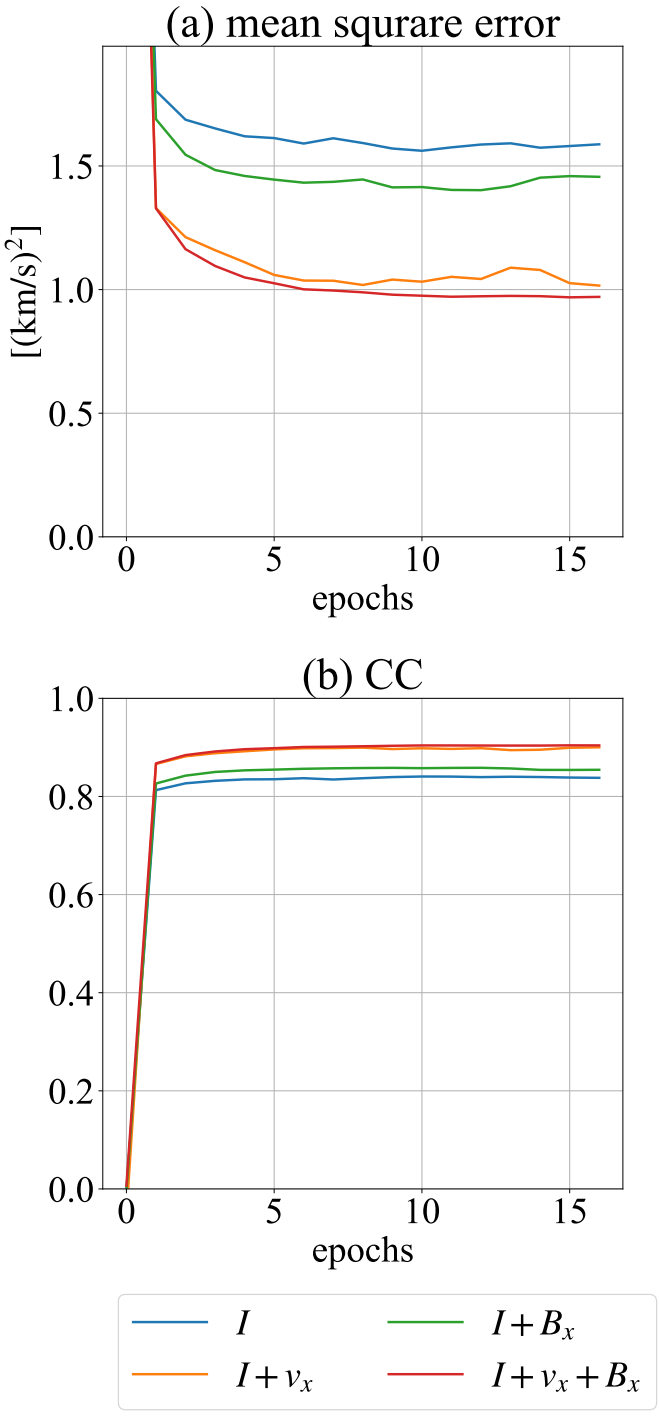


Figure3.6 The learning curve for each set of input data. Panel (a) shows the mean squared error, and Panel (b) shows the correlation coefficient.

Network Name	I	IV	IB	IVB
Input Data	I	I, v_x	I, B_x	I, v_x, B_x
Correlation Coefficient	0.84	0.90	0.86	0.90
R2 Score	0.71	0.81	0.74	0.81
Mean Squared Error $[(\mathrm{km\,s^{-1}})^2]$	1.56	1.02	1.39	0.96
Mean Absolute Error $[\mathrm{km\,s^{-1}}]$	0.92	0.74	0.88	0.72
Mean Angular Difference [degree]	28.6°	22.6°	26.9°	21.9°

Table3.3 Evaluation metrics for the evaluation performance with each set of input physical quantities. The effect of the change in the shape of the input data has resulted in a small change from the previous Table in the results based on intensity only.

3.6 Application to Observations

In this section, we apply the trained network to observational data. The observational data used are continuous light images taken through a green continuum filter centered at 555 nm by the Solar Optical Telescope (SOT) on the Hinode satellite (Kosugi et al., 2007; Tsuneta et al., 2007) on December 25, 2007. Figure 3.7 a shows the full image of the observational data used in this study. Linear interpolation is applied to match the resolution of the observational data with that of the simulation data used for training. The resolution-corrected data for the region enclosed by the rectangle in the left image is shown in Figure 3.7 b. The data are also normalized by the mean value of the entire dataset for input.

To match the Hinode observations, we convolve the simulated intensity with the Hinode point spread function (PSF) at 555 nm, as described in Welsch et al. (2004). The PSF is not applied to the output horizontal velocity field to evaluate a velocity field as close to the simulation as possible. However, leaving the network as-is causes overfitting, preventing accurate evaluations. Figure 3.8 shows the wavenumber distribution obtained by Fourier transforming the intensity data used in this study. The Fourier transform of the intensity $f(x_l, y_m)$ at position (x_l, y_m) is given by

$$F\left(k_x, k_y\right) = \sum_l \sum_m f(x_l, y_m) \exp\left(-2\pi i (k_x x_l + k_y y_m)\right). \tag{3.7}$$

The wavenumbers k_x and k_y range from $1/N$ to $1/N$ in increments up to $N/2$, where N is the number of pixels in the horizontal and vertical directions, and in this study, $N = 128$. The figure shows $F\left(k_x, k_y\right)$ as a function of the wavenumber magnitude $\sqrt{k_x^2 + k_y^2}$.

In large-scale ranges, the spectra of the observation and simulation data match, but in

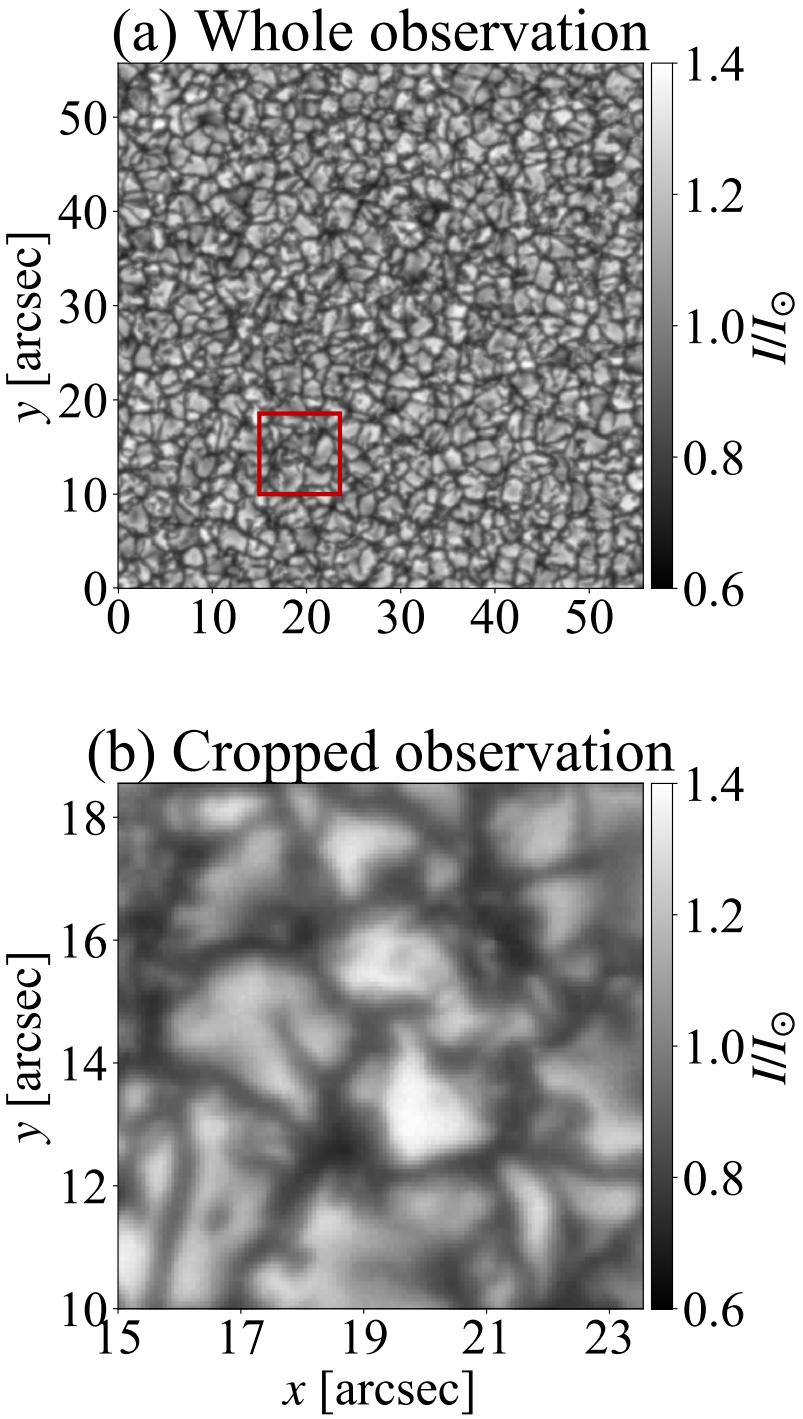


Figure3.7 An example of the observational data from the Hinode used in this study. The left image shows the entire observation region and the right image shows the portion extracted for network input, which is linearly interpolated. The extracted region is indicated by the red rectangle in the left image.

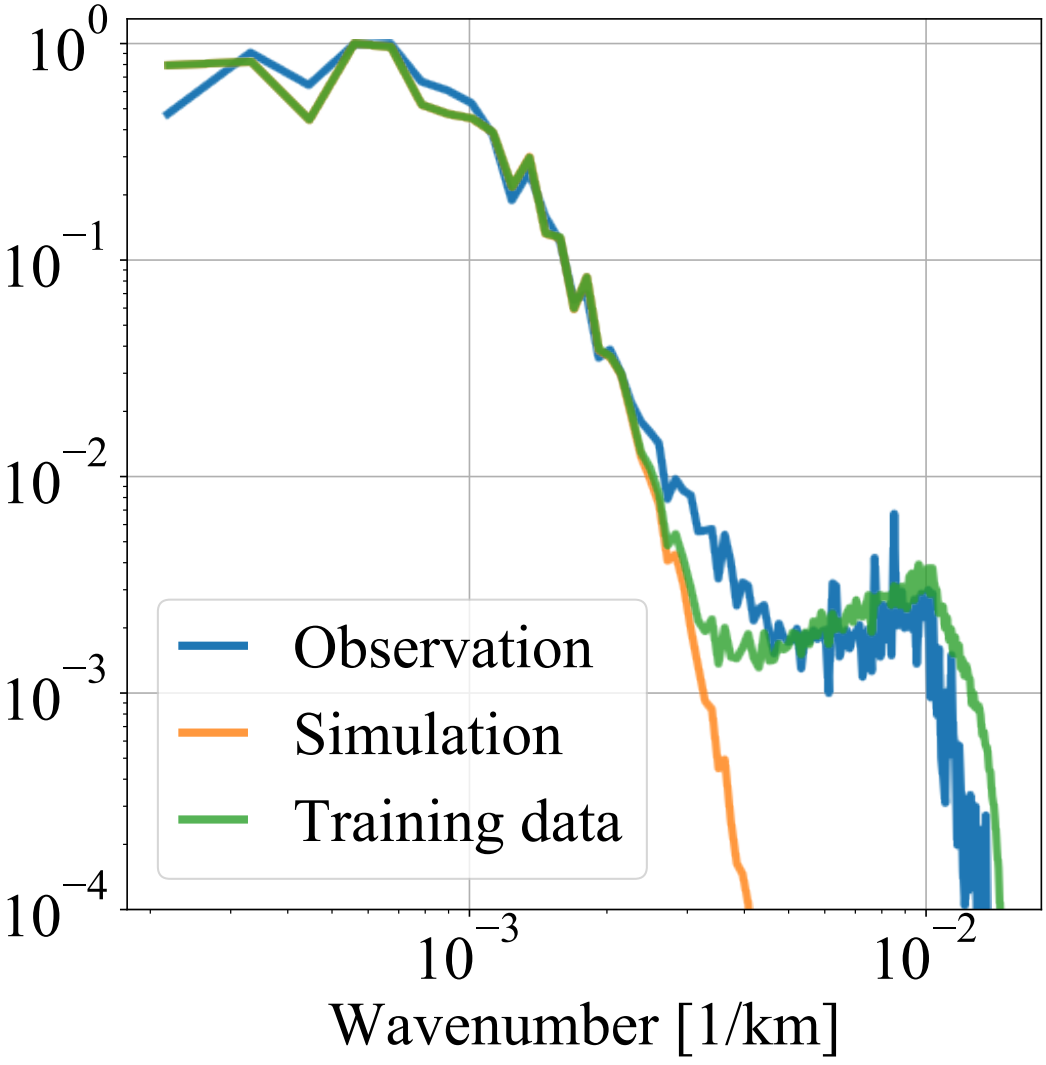


Figure3.8 Power spectrum of the intensity is shown. The blue, orange, and green lines represent the spectra for the observation and simulation training data, respectively. The horizontal axis represents the wavenumber, and the vertical axis represents the intensity normalized by the maximum value.

small-scale ranges, they do not. Random noise is added to the simulated intensity input to suppress small-scale structures and prevent the network from learning small-scale structures.

Random noise is added to intensity normalized with the maximum value by generating Gaussian-distributed random numbers with a mean of zero and standard deviation of 1.6×10^{-3} in the wavenumber distribution and then performing an inverse Fourier transform. The inverse Fourier transform is given by

$$F\left(x,y\right) = \sum_{l}\sum_{m} f(k_{x_l}, k_{y_m}) \exp\left(2\pi i(k_x x_l + k_y y_m)\right). \tag{3.8}$$

The wavenumber distribution after adding noise is shown by the green line. No noise is added to the output horizontal velocity field data, as it is expected to evaluate a horizontal velocity field as close to the simulation as possible. Other training settings remained unchanged.

We name the network trained with only PSF applied to the intensity "Network IP" and the network trained with PSF and noise "Network IPN." Figure 3.9 shows the result of applying this network to the observational image. In Panel (c), where the result of the Network IP, the image is distorted and does not seem to evaluate accurately. On the other hand, the Network IPN in Panel (b) is able to make accurate evaluations. This suggests that the small-scale structures in the intensity data significantly affect the evaluation, and removing them with noise enables the accurate evaluation of the horizontal velocity field. Figure 3.10 shows the results of applying the networks to simulation data that are different from the training data. In this case, accurate evaluations are possible. Therefore, this issue of inaccurate evaluations may be resolved by conducting observations with a higher resolution and a lower noise that matches the small-scale structures in the observational data with those in the simulation.

The performance of the network declines when PSF and noise are applied compared to when they are not. The correlation coefficient decreases to 0.64, and the R2 score drops to 0.42. Figure 3.11 shows the 2D histogram of the horizontal velocity from the simulation and the horizontal velocity field evaluated by the network trained with noise. It can be seen that the network tends to underestimate the velocity field even more than in the simulation.

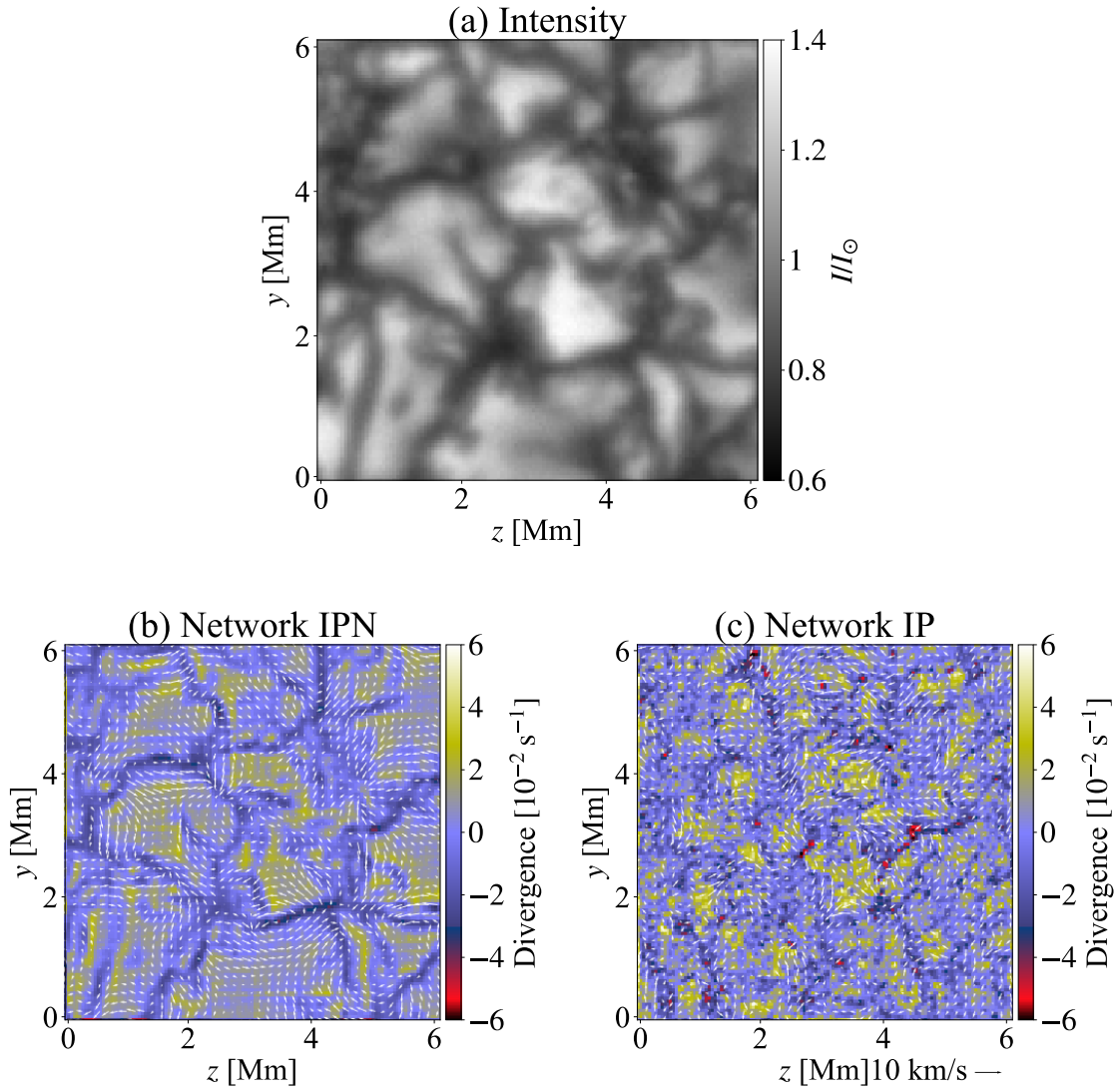


Figure3.9 The images show (a) the intensity observed by the Hinode, (b) the results of the Network IPN, and (c) the results of the Network IP. Panel (c) shows significant distortion in the image, indicating overfitting.

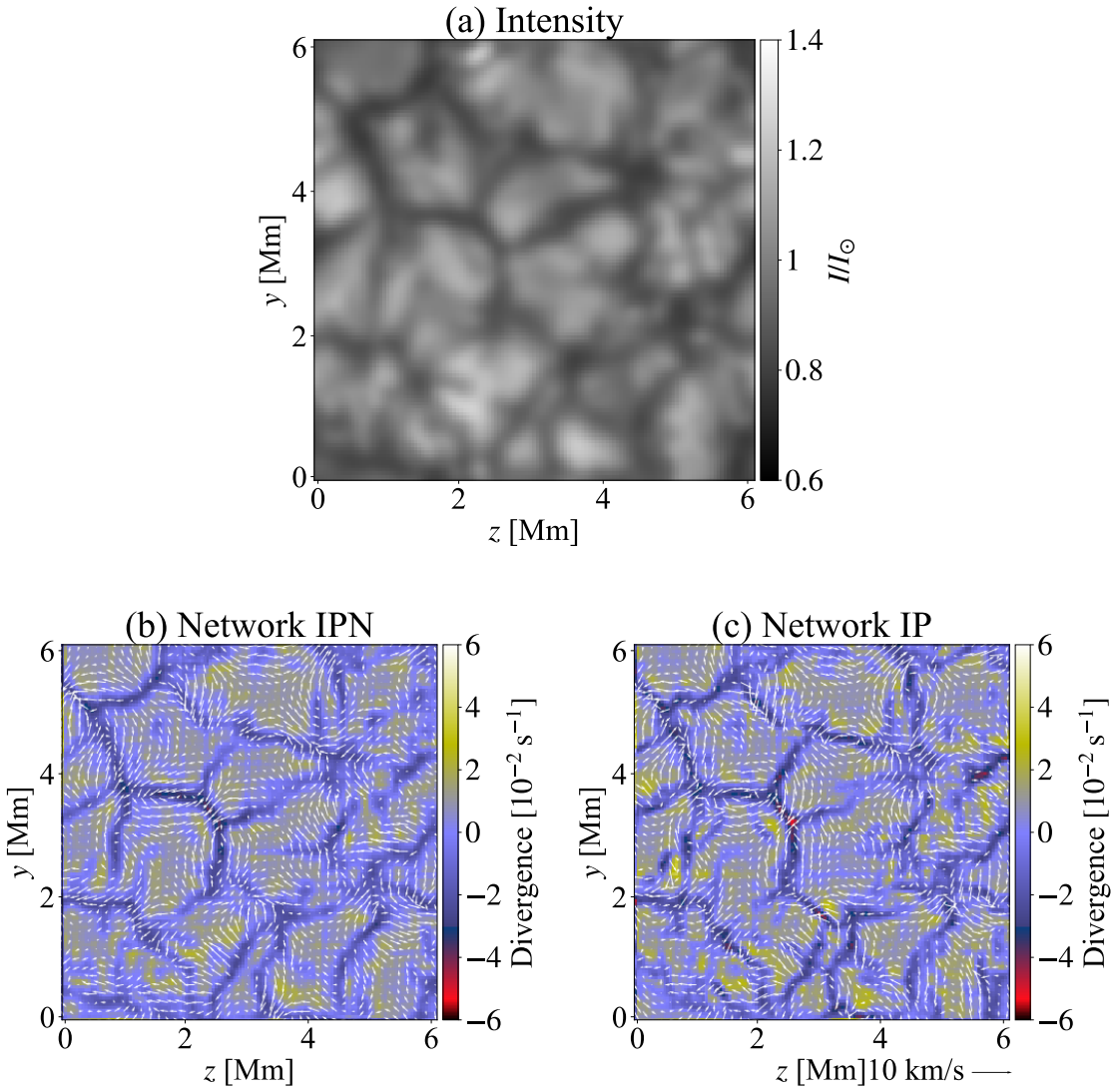


Figure3.10 The images show (a) the input intensity from the simulation, (b) the results of the Network IPN, and (c) the results of the Network IP. In the simulation data, the results in Panel (c) are not distorted, and accurate evaluations are possible.

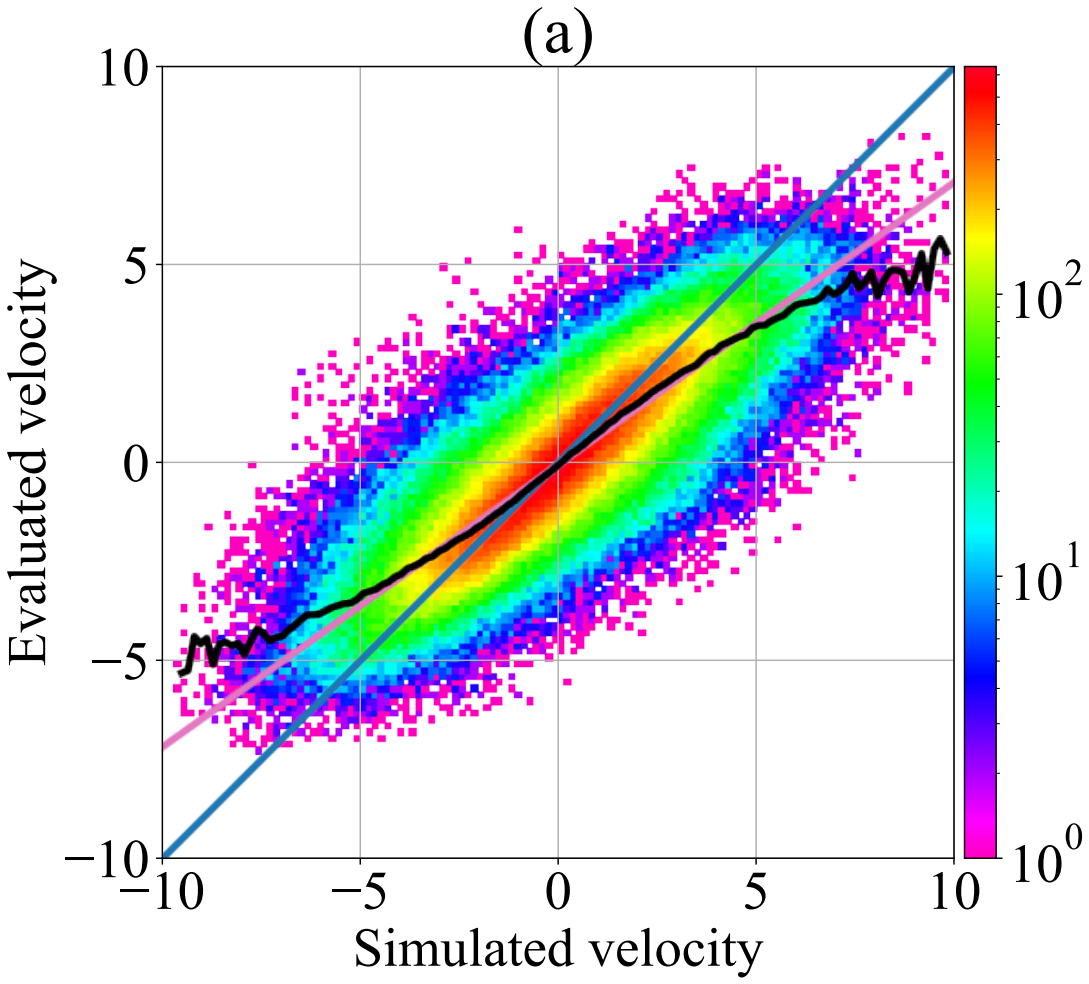


Figure3.11 2D histogram of the horizontal velocity from the simulation and the horizontal velocity field evaluated by the network IPN. Panels (a) and (b) show the errors when evaluated using only intensity and when using intensity and vertical velocity, respectively. The colormap represents the number of pixels corresponding to each value. The blue line represents the line where the two values are equal, $v_{\mathrm{sim}} = v_{\mathrm{eva}}$. The pink line is the approximation line, with $v_{\mathrm{eva}} = 0.485 v_{\mathrm{sim}} - 0.123$. The black line indicates the average evaluated velocity for each simulation velocity.

3.7 Comparison with LCT

We compare the network developed in this study with LCT. For LCT, we use the LCT code developed by Fisher and Welsch (2008). The full width at half maximum of the Gaussian filter used for the LCT is set to 1200 km. The LCT is applied 20 times with continuous data at 30-second intervals using either the observational data from the Hinode or the simulated intensity data, and the average is taken. The result corresponds to a 10-minute average of the horizontal velocity field.

First, we apply it to the simulated intensity data. Figure 3.12 shows the result. The velocity field evaluated by the network is relatively similar to the simulation, but the LCT result does not match either. The correlation coefficient between the LCT result and the simulation is 0.14, and between the LCT and the network evaluation is 0.16. Both values are low. The correlation between the network and the simulation is 0.91. Since small-scale, short-lived structures are smoothed out in the time average, the correlation coefficient of the neural network improves.

Next, we apply it to the observational data. The result is shown in Figure 3.13. In this case, the correlation coefficient between the LCT result and the network evaluation is -0.03, showing no agreement at all. The observational data and the simulation data used in this study both have a resolution of about 50 km, and the typical velocity of granulation is 3-4 $\mathrm{km\,s^{-1}}$. Therefore, in an image with a 30-second interval, the structures only move about two pixels. As a result, the LCT struggles to extract displacement in structures of the scale of granulation. The average velocity evaluated by the LCT is also smaller, indicating that the velocity is not fully captured during displacement extraction. Malherbe et al. (2018) reported that the LCT is poor at detecting granulation flow. Additionally, the LCT is inherently difficult to evaluate when the movement of the pattern extends outside the field of view. Since the network developed in this study evaluates from snapshots, it does not face this problem.

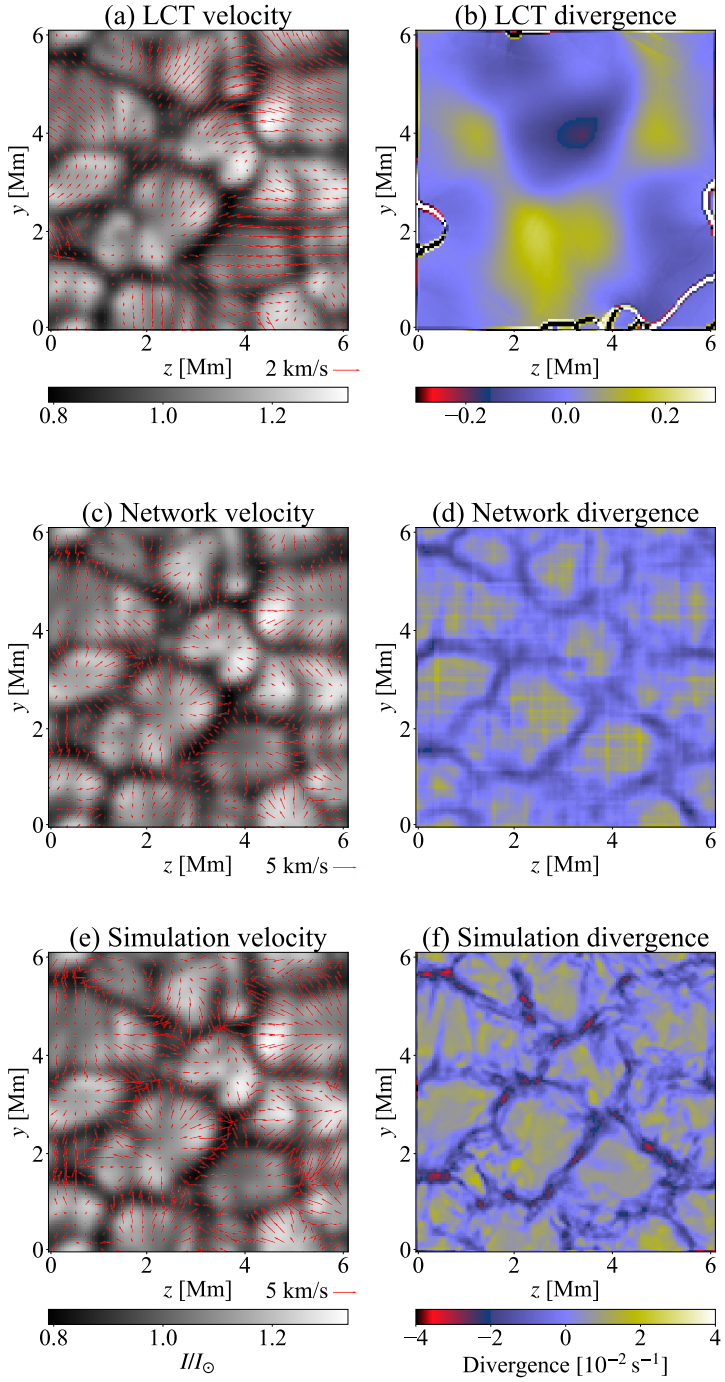


Figure3.12 The figure shows the velocity evaluated by the LCT and the network using simulation data. The background of the left column shows the intensity, and the red arrows indicate the horizontal velocity. The right column shows the horizontal divergence of the velocity field. The top row (a, b) shows the LCT results, the middle row (c, d) shows the network evaluation results, and the bottom row (e, f) shows the true velocity field from the simulation.

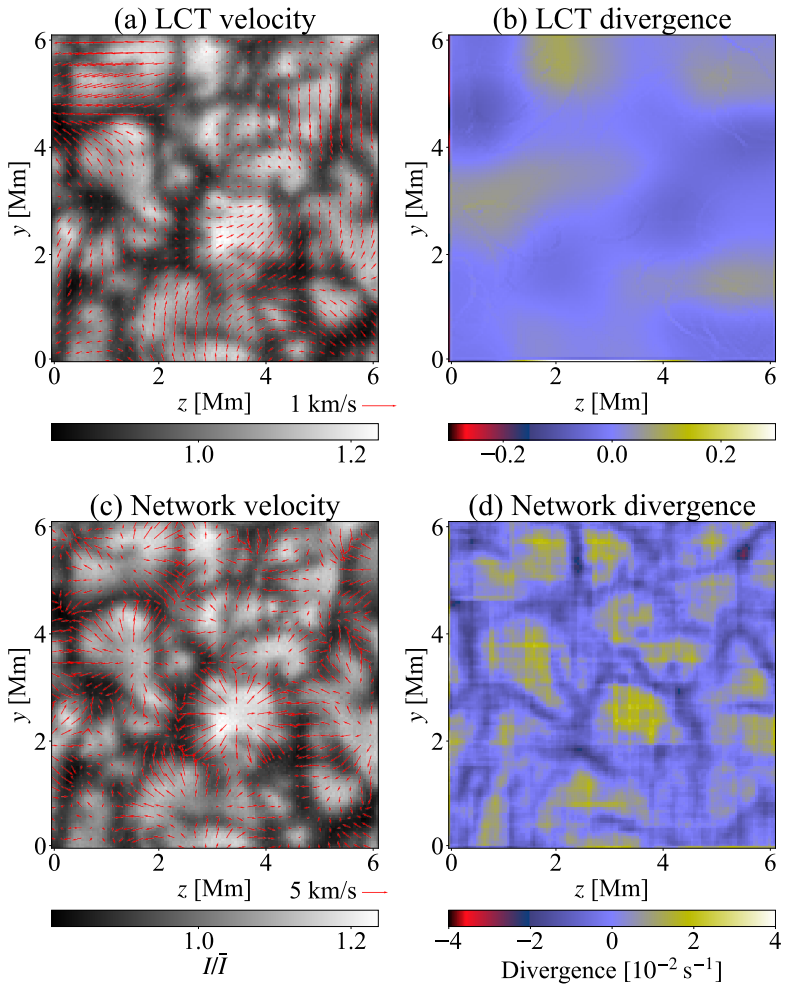


Figure3.13 The figure shows the velocity evaluated by the LCT and the network using the observational data from the Hinode. The display format is the same as in Figure 3.12. The intensity is normalized by the average intensity $\bar{I}$ over the entire simulation region.

3.8 Comparison with Previous Studies

We compare our study with previous research that estimated the solar velocity field using similar methods. DeepVel and DeepVelU estimate the horizontal velocity using two consecutive intensity images of LoS velocity or LoS magnetic field. The method by Ishikawa et al. (2021) estimates the velocity from three consecutive frames. These approaches serve as alternatives to the LCT. In contrast, our network can estimate the velocity field from a single snapshot, which the LCT cannot handle, and obtain convective information without relying on temporal evolution.

The training data for DeepVel consists of 30,000 sets of 50×50 pixel images, while DeepVelU uses 2,000 sets of 48×48 pixel images. The method by Ishikawa et al. (2021) uses 350 sets of 128×128 pixel images. Our dataset consists of 120,000 images of 128×128 pixels, which is 20 times more data than that used in DeepVel, the study with the largest dataset. This increase in data volume enables velocity estimation from a single snapshot. The correlation coefficient between the estimated velocity and the simulation velocity in the previous studies is 0.84 for DeepVel and 0.95 for both DeepVelU and Ishikawa et al. (2021). In our method, the correlation coefficient is 0.84 when using simulation data directly and 0.64 when adjusted with noise.

To enable velocity estimation from a single snapshot, we add noise, resulting in lower performance compared to methods using two or more input data. In the evaluation of horizontal velocity fields, if consecutive data are available, increasing the input improves performance. Therefore, it is effective to choose the approach based on the available data.

Chapter 4

Evaluation of Internal Velocity Fields

This chapter describes the method for evaluating the distribution of vertical upflow velocity fields inside the Sun based on physical quantities such as radiation intensity, which are relatively easy to observe.

4.1 Simulation Setting

To evaluate the internal structure of the Sun, we conduct simulations on a scale that includes supergranulation, which are thought to be structures caused by thermal convection deep in the Sun. Since supergranulations have a typical size of approximately 30 Mm, the computational domain in the horizontal direction is set to 49.152 Mm. In the depth direction, it extends 23.876 Mm and includes the upper layers of the convection zone, up to 700 km above the solar surface. The horizontal grid spacing is 96 km, resulting in a 512 $\times$ 512 grid. In the vertical direction, a non-uniform grid is used with a minimum spacing of 48 km. Near the photosphere, where accurate radiative transfer is important, the upper 4.608 Mm of the computational box consists of a uniform grid with 96 grids at 48 km intervals. Beyond that, 256 grids are used, where the grid spacing Δx increases by a constant value δx according to the following equation, except for the two grids at both ends:

$$\Delta x = n\delta x \tag{4.1}$$

where n is the number of grids from the upper part of the uniform grid region. Setting δx so that the depth becomes 23.876 Mm gives $\delta x = 2.4$ km, and the grid spacing at the bottom of the computational box is $\Delta x = 90$ km. The radiative transfer equation is solved using two

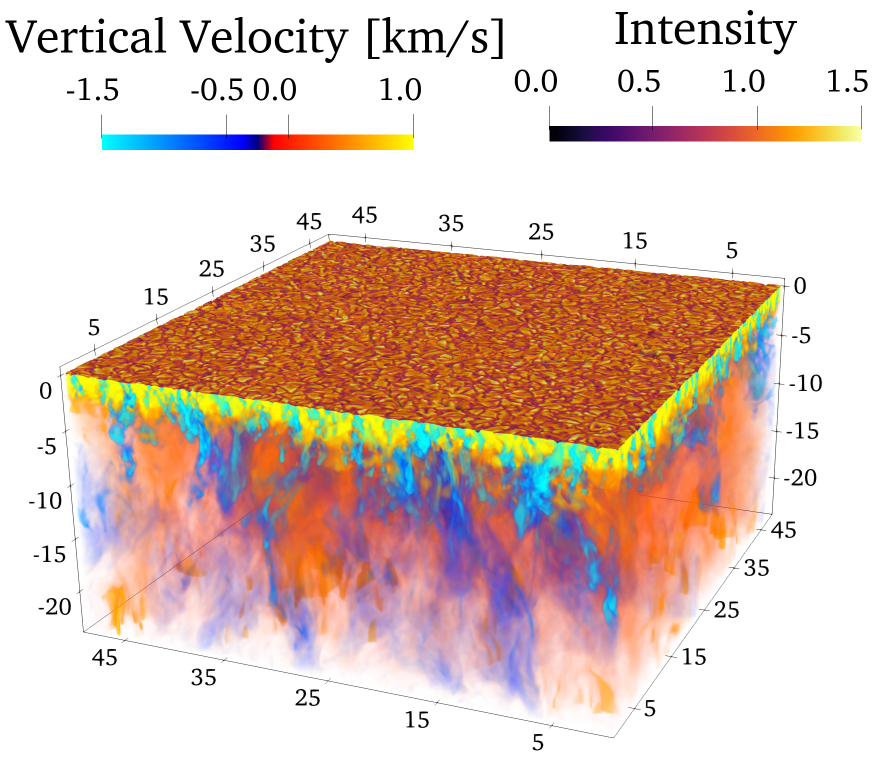


Figure4.1 3D volume rendering of the wide-range simulation data prepared for network training. The color within the box represents vertical velocity, and the surface map on the top of the box shows intensity.

rays.

We run five simulations with initial magnetic fields of 10 G, 13 G, 15 G, 20 G, and 25 G. Four of the cases output data at 5-minute intervals, based on the typical lifetime of surface granulations, while one case output data at 45-second intervals for validation using helioseismology.

The sound speed at a depth of 25 Mm is approximately 60 $\mathrm{km\ s^{-1}}$, so according to the CFL condition, The time step Δt is calculated as $90\,(\mathrm{km})/60\,(\mathrm{km\ s^{-1}})$, giving approximately 1.5 seconds. To ensure stability, the time step is taken to be approximately 1 second. A total of 30 days of simulation is performed, with a total of around 3 million steps. In total, approximately 5,000 data sets are obtained. A sample of the simulation data is shown in Figure 4.1.

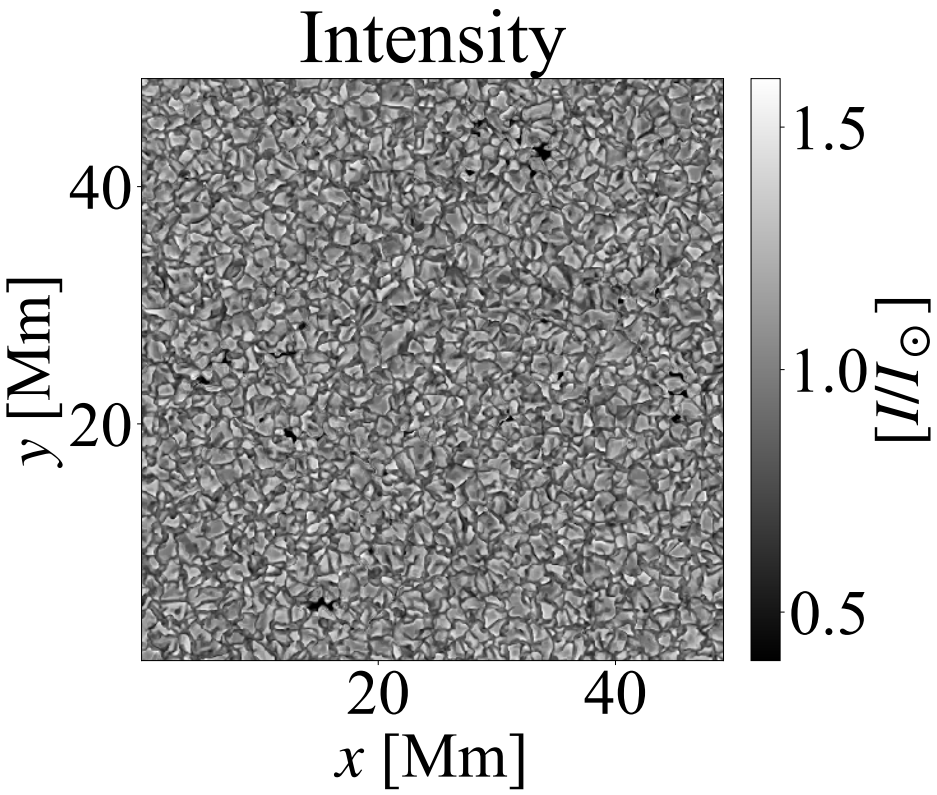


Figure4.2 Figure of emergent intensity from the top of the computational box. An example of the wide-range simulation data prepared for network training.

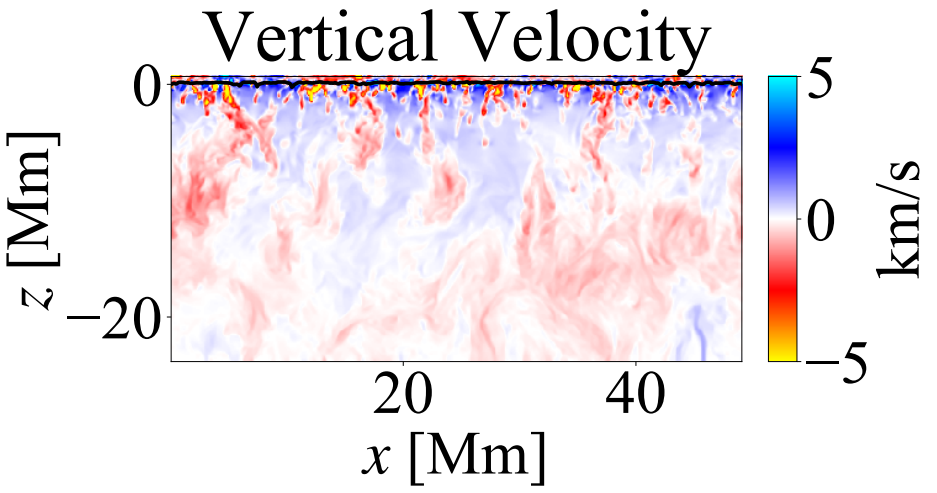


Figure4.3 An example of the wide-range simulation data prepared for network training. A horizontal slice of the vertical velocity. The black line is the plane with optical depth $\tau = 1$.

4.2 Training Setup

The network architecture is shown in Figure 4.4. This architecture is based on the U-net structure. The encoder-decoder structure is typically used when the input and output image sizes are the same. In this study, the input data size is 128 × 128, while the output data size is 512 × 512, so the decoder structure is extended to match the size of the output images.

In the encoder, each convolutional layer performs two convolutions with a kernel size of 3 × 3. Afterward, downsampling is done using max pooling over a 2 × 2 region. The number of kernels doubles with each downsampling, starting with 256 in the first layer. In the decoder, after similar convolutions, upsampling is done by expanding 1 pixel into a 2 × 2 block. Skip connections are made by concatenating tensors in the kernel direction.

The 3,000 sets of simulation images are quadrupled through 90-degree rotations, resulting in approximately 12,000 sets used for training. Furthermore, 1,000 images are reserved for validation and another 1,000 for testing. Adam optimizer (Kingma and Ba, 2014) is used for optimization, and the hyperparameters follow the paper of Adam unless otherwise noted. The mini-batch size is 4. Training is conducted for eight epochs, and the result with the highest correlation coefficient in the evaluation data is adopted as the final outcome. This code is implemented using TensorFlow, and training is performed using an Nvidia RTX 2080 Ti GPU.

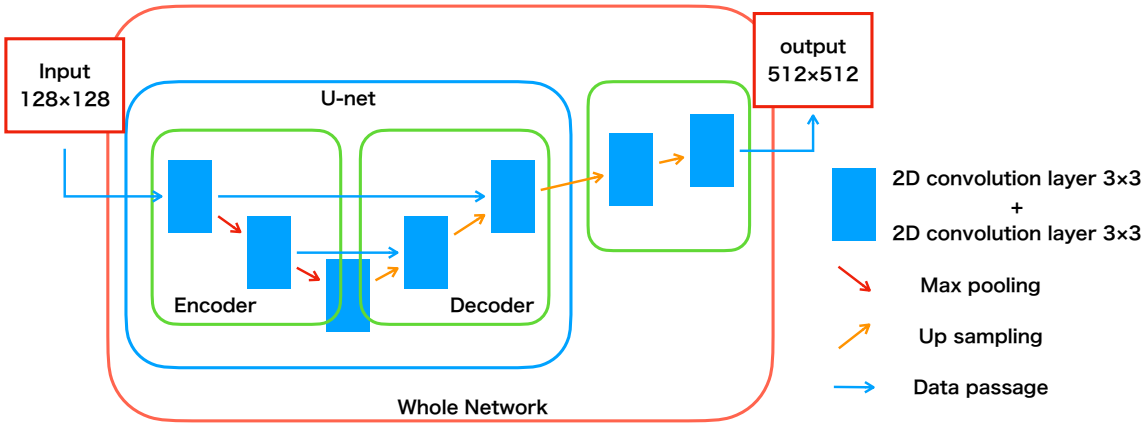


Figure4.4 Diagram of the network architecture.

4.3 Adjustment of Simulation Results to Match Observation

In this study, the neural network is trained for application to the SDO data. The PSF of the SDO, as described in Yeo et al. (2014), is used to reproduce the blurring effect in the observational data on the simulation data. Additionally, the observational data from SDO has a resolution of 0.03 degrees in both latitude and longitude, corresponding to 0.35 Mm at the equator. Since the simulation has a grid spacing of 0.096 Mm, to match the resolution, the values of 4×4 grids are averaged to create one grid, resulting in a resolution of 0.384 km and an image of 128×128 pixels.

As in chapter 3, the small-scale structures in the observational data and simulation data do not match. Since this mismatch negatively affects learning, random noise with a magnitude of 5% of the standard deviation of the intensity after Fourier transformation is added to erase small-scale structures at small wavenumbers. The inverse Fourier transformed data is then used as training data.

The processed input data consists of intensity, vertical velocity field, and vertical magnetic field obtained from simulations. Three images are input every hour, totaling nine images of 128×128 pixels, which simulate the physical quantities at SDO's resolution. The output is the vertical velocity field at various depths obtained from the simulation. Since the output data is not preprocessed for observational noise and resolution, the simulation data results are used as-is with a size of 512×512 pixels.

4.4 Training Results

The nine panels in Figure 4.5 show the input data to the network. The top row shows the intensity, the middle row shows the line-of-sight velocity field, and the bottom row shows the line-of-sight magnetic field. The central column corresponds to the data at the same time as the velocity field to be evaluated, with the left column showing data from one hour earlier and the right column from one hour later. Each data set is normalized with a mean of 0 and a standard deviation of 1 for efficient learning.

Figure 4.6 and Figure 4.7 show the color map of the simulation data, considered the ground truth, and the evaluated velocity field at a depth of 12 Mm. Although the small-scale structures have been smoothed out, the large-scale convective structures seem to be roughly reproduced. The correlation coefficients between the evaluated velocity and the true simulation are 0.28 at a depth of 5 Mm, 0.38 at 12 Mm, and 0.37 at 18 Mm. The lower correlation at 5 Mm is due

to the smaller convective scales, which are harder to match.

Figure 4.8 shows the coherence spectra at each depth. The coherence spectrum is a metric used to evaluate the degree of agreement for each scale of image structure, as used by Ishikawa et al. (2022). The coherence spectrum is obtained as follows: a 2D Fourier transform is applied to both images to obtain X and Y.

$$\hat{X}(k_x, k_y) = \frac{1}{N^2} X(x, y) \exp[-i(k_x x + k_y y)] \tag{4.2}$$

The sum of $\hat{X}^2$ over an interval with width Δk around each k is calculated to obtain the power spectrum for a specific interval.

$$E_X(k) = \frac{1}{2\Delta k} \sum_{\sqrt{k_x^2+k_y^2} \in [k-\Delta k/2, k+\Delta k/2]} \left| \hat{X}(k_x, k_y) \right|^2 \tag{4.3}$$

Similarly, the cross-spectrum is obtained by summing X, Y over the same interval. The coherence spectrum is then calculated by the following equation:

$$S_{XY}(k) = \frac{1}{2\Delta k} \sum_{\sqrt{k_x^2+k_y^2} \in [k-\Delta k/2, k+\Delta k/2]} \left\langle \hat{X}^* \hat{Y} \right\rangle (k_x, k_y) \tag{4.4}$$

$$\gamma_{XY}(k) = \frac{S_{XY}(k)}{\sqrt{\langle E_X \rangle (k) \langle E_Y \rangle (k)}} \tag{4.5}$$

This value corresponds to the correlation coefficient at specific spatial scales, with values closer to 1 indicating greater agreement and values closer to 0 indicating no correlation. The wavenumber is divided into 256 bins, with intervals of 135.36 $[\mathrm{km}^{-1}]$ chosen as δk. From this figure, it is evident that large-scale structures agree with a value of about 0.7. It is also consistent with the intuitive expectation that deeper layers are more difficult to evaluate.

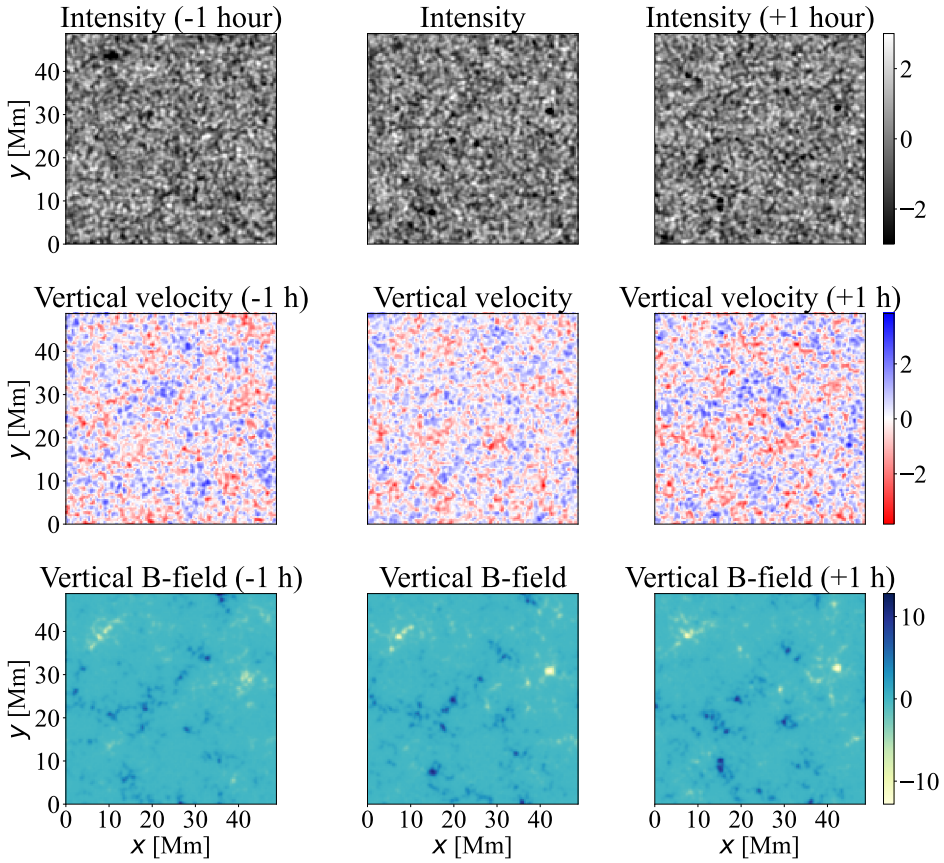


Figure4.5 The panels show the input data by simulation for the neural network. These images represent the values normalized and processed.

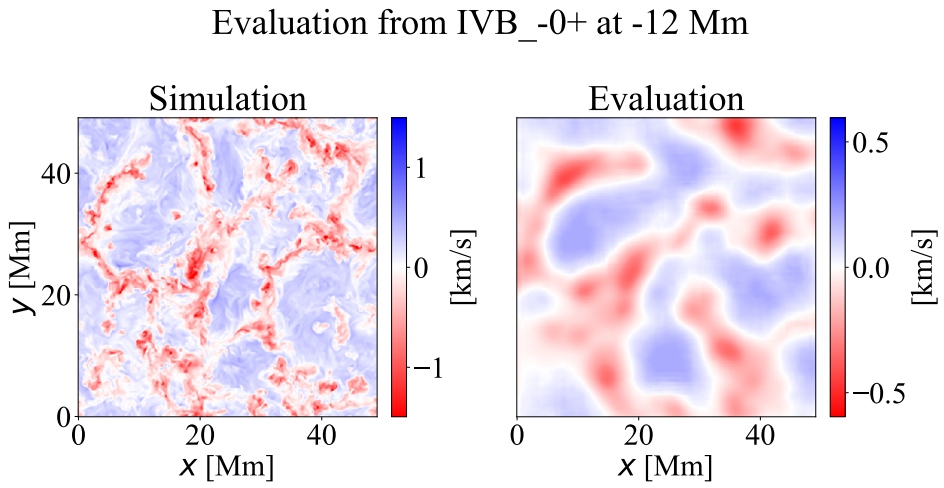


Figure4.6 The panels show the result of the simulated and the evaluated velocity fields at 12 Mm depth. IVB_-0+ is a label that indicates the use of three physical quantities and three time steps as inputs, and it will be used in Chapter 4.6.

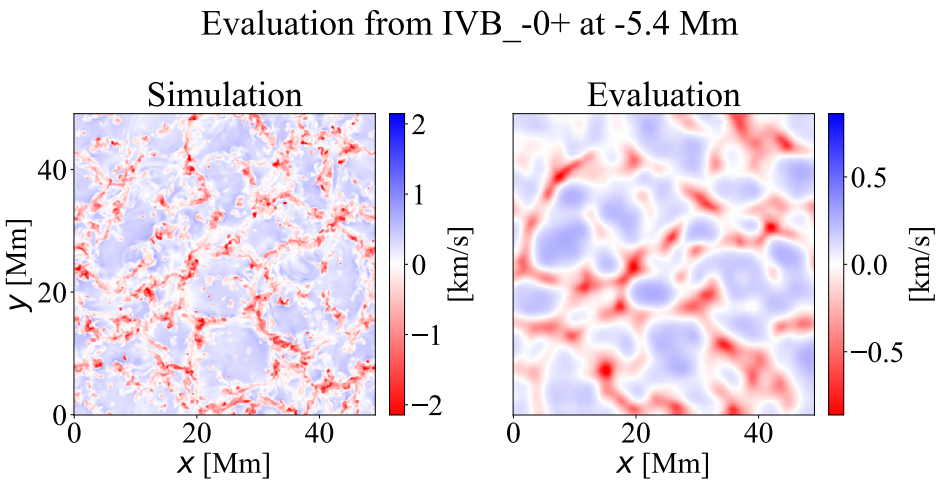


Figure4.7 These panels show the result of the simulated and the evaluated velocity fields at 5.4 Mm depth.

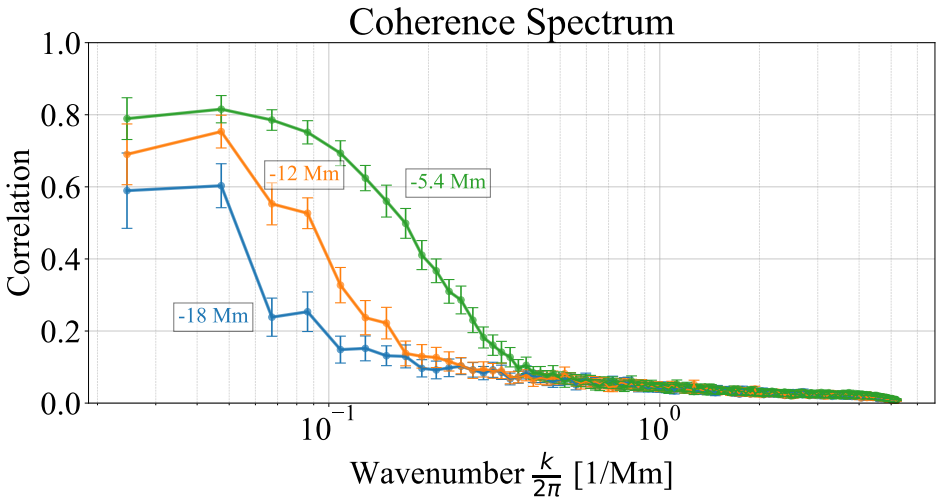


Figure4.8 This panel shows the coherence spectrum between the evaluated and simulated velocities in the deep convection zone. The error bars represent the 99% confidence interval.

4.5 Comparison with Helioseismology

To validate the accuracy of the internal structure evaluated by the neural network in this study, a comparison is made using helioseismology, a traditional method for probing the internal structure of the Sun. Helioseismology is performed according to the method of Gizon and Birch (2004) (See Appendix C.2). However, since uncertainties arise when performing inversion to calculate internal velocities, this study only compares the travel time. A distance

of 12 Mm between two points is chosen, and the filter corresponding to a distance of 8.7–14.5 Mm for i=3, as shown in Gizon and Birch (2004), page 30, Table 1, is used. The travel time shift is calculated in 256 directions, equally spaced by 360 degrees. Helioseismology is calculated using 24 hours of data. For comparison, the neural network velocities are averaged every hour over 24 hours of data.

4.5.1 Comparison with Simulation Data

Helioseismology calculations are performed using simulation data to compare the two methods. The simulation used for helioseismology is different from the one used for training and validation, with data output at a 45-second cadence for approximately one day of simulation. Figure 4.9 shows an input sample at the central time of the 24-hour data. The left panel of figure 4.13 shows the time-averaged internal velocity field from the simulation over one day, which serves as the ground truth. The middle panel shows the divergence of the travel time obtained from helioseismology. The right panel shows the internal structure evaluated from the surface intensity, vertical velocity field, and magnetic field, calculated at hourly intervals and averaged over 24 hours. The correlation coefficients are -0.31 between the simulation and helioseismology, 0.59 between the simulation and the neural network, and -0.35 between the neural network and helioseismology. The R2 scores based on the simulation were 0.18 for the network and -0.38 for helioseismology. The coherence spectra in Figure 4.11 indicate that the neural network's evaluations are effective even at smaller scales.

4.5.2 Comparison with Observational Data

Figure 4.12 shows the observational data. The data used are from the SDO (Pesnell et al., 2012), recorded between 00:00:45 and 23:59:45 on April 5, 2011. The results are based on 1 day of images taken at a 45-second cadence, totaling 1919 images. The travel time is calculated using a window corresponding to a distance of 12 Mm, and its horizontal divergence is calculated. Convergence corresponds to downflows, while divergence corresponds to upflows. Figure 4.13 shows the horizontal divergence of the travel time from helioseismology and the evaluated upflow velocity field from the neural network. Since helioseismology evaluates the average flow, the network's evaluations are also averaged over time. The average is based on 24 sets of evaluations made from 24 sets of data (each set covering 1 hour) over the course of a day (note that the edge data cannot be evaluated due to the 3-hour set requirement). The calculation time for travel time over one day of data is 30 seconds, while

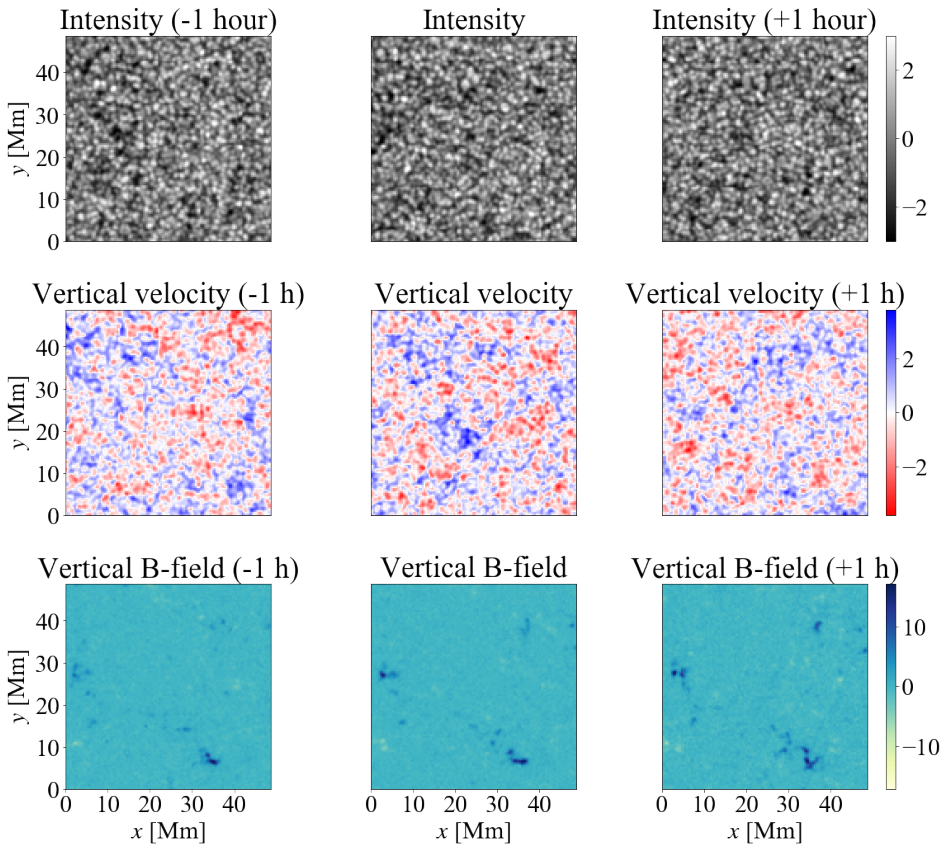


Figure4.9 These panels show the input data by simulation for the neural network. These images are input data and have been normalized.

the neural network evaluation takes 3 seconds. Visually, large-scale structures also appear to match. The correlation coefficient is -0.46, with a negative sign due to the relationship between divergence and velocity fields. Figure 4.14 shows the coherence spectra between the two. At scales of about 20 Mm, a correlation of approximately 0.7 indicates a good match between the two. However, small-scale structures show little agreement.

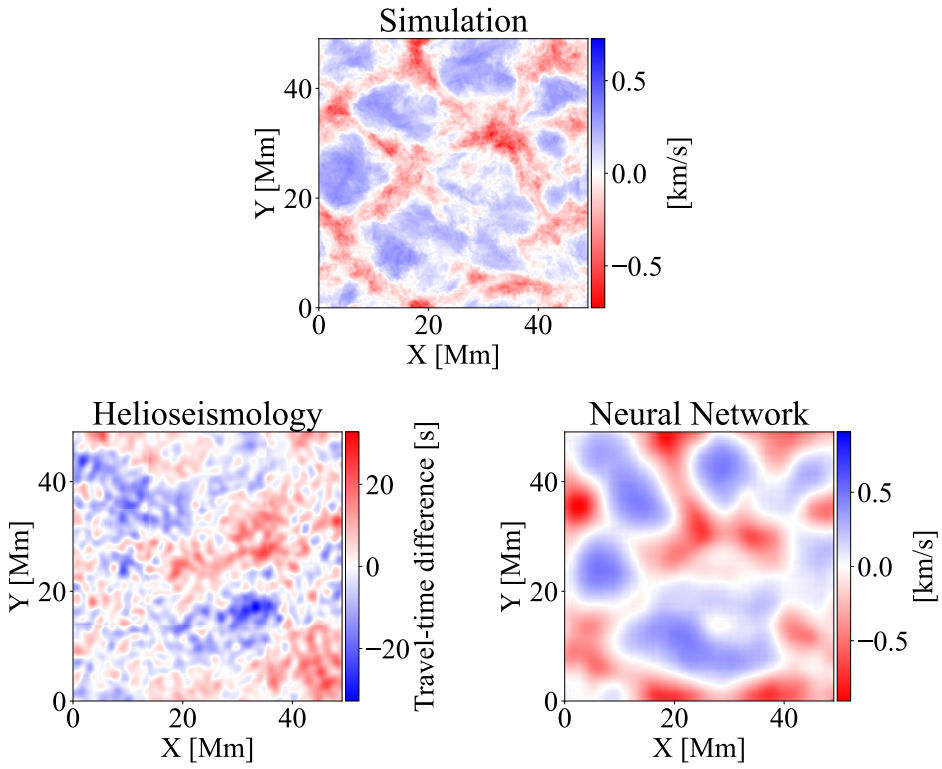


Figure4.10 These panels compare the simulated velocity, results from the helioseismology, and the evaluated velocity field.

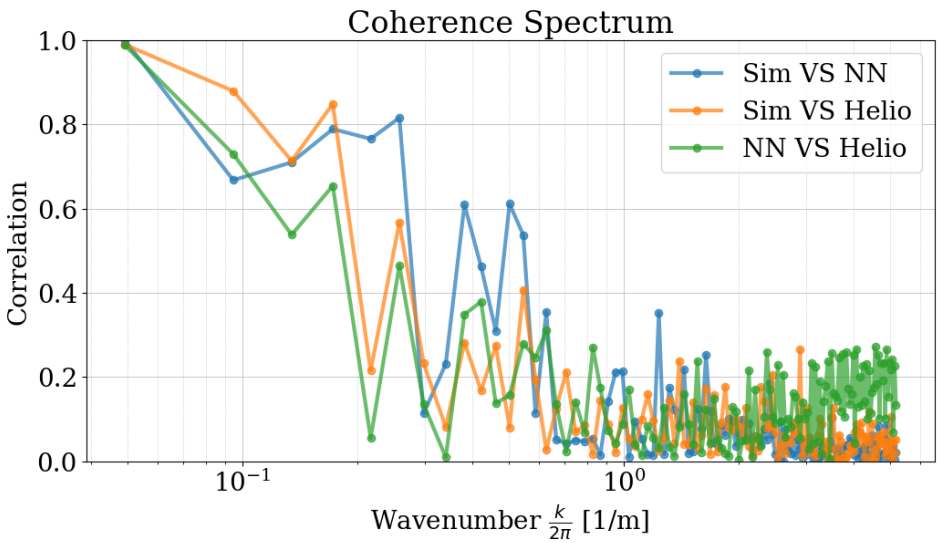


Figure4.11 This depicts the coherence spectra.

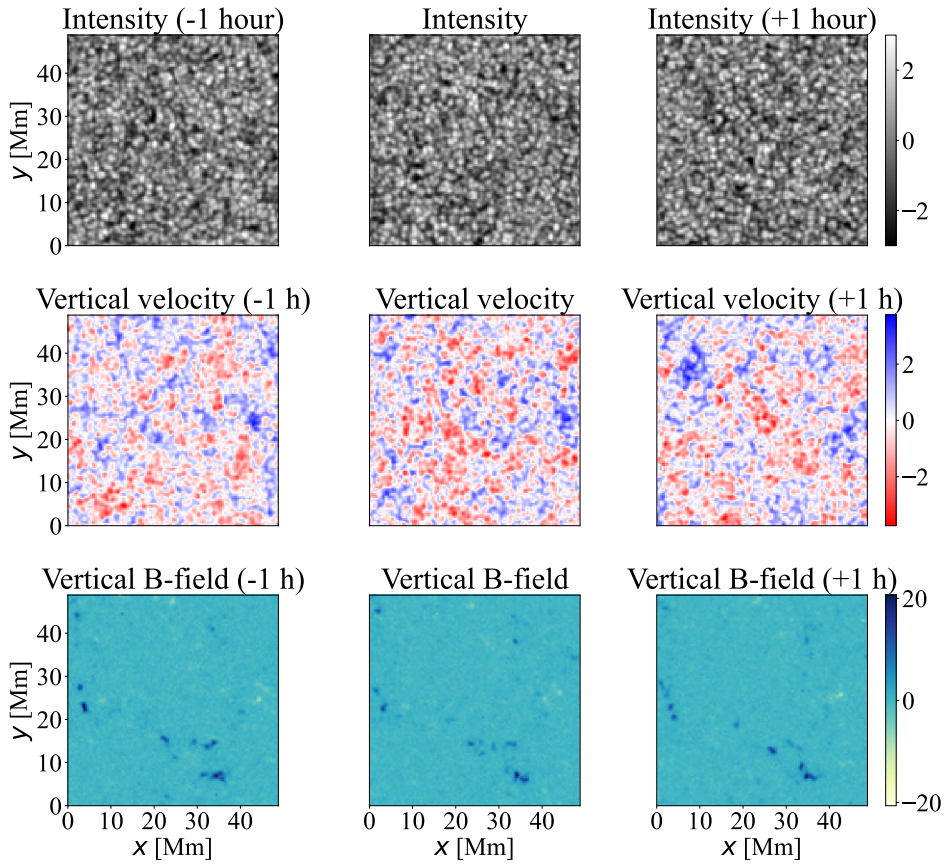


Figure4.12 This panel shows the input data by observation for the neural network. These images represent the values normalized and processed.

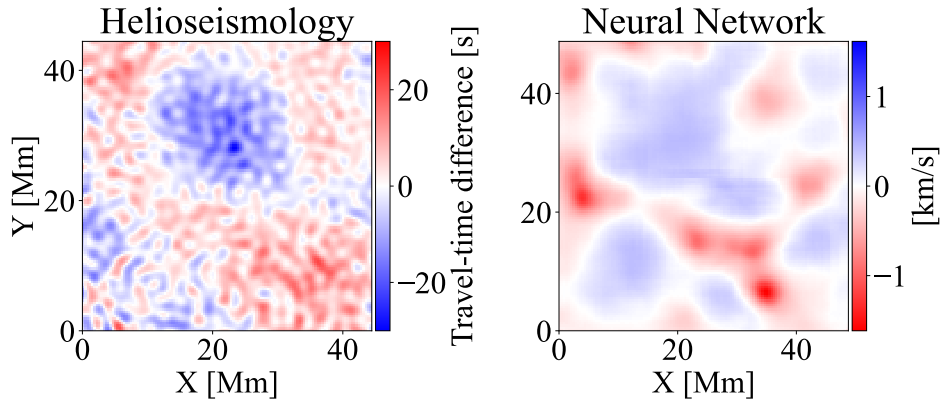


Figure4.13 This panel compares the results from the helioseismology and the evaluated velocity field.

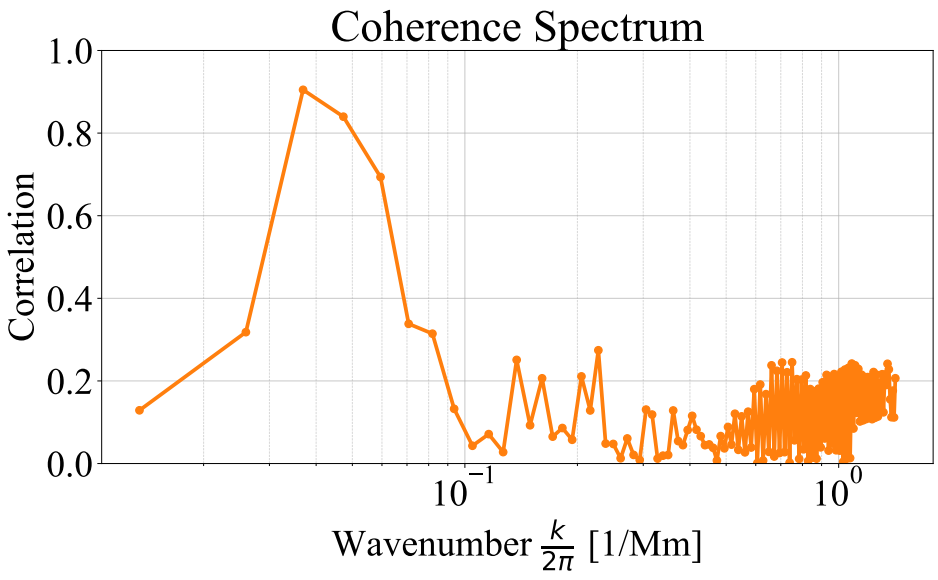


Figure4.14 The bottom right panel depicts the coherence spectra between helioseismology and the evaluation.

4.6 Dependence of Network Performance on Input Quantities

We compare the network performance with different types of input quantities. A subset of physical quantities is replaced with zero-valued data of the same size and used as input for training. The size of the training data and input data, as well as the network structure, remain unchanged. Among the physical quantities, three types are examined: intensity, vertical velocity field, and vertical magnetic field, either individually or in combinations of two.

Additionally, the time of the input data is also considered. When applying this method in actual observations, using data one hour after the target time involves utilizing future information for evaluation, requiring a wait time of at least one hour. Therefore, comparisons are made among three patterns: 1. data from the target time, 2. data from the target time and one hour prior, and 3. data as described in the previous chapter. Comparisons are conducted for two depth levels: 12 Mm and 5.4 Mm.

Labels for inputs are described as follows: models using intensity are labeled "I," those using vertical velocity are labeled "V," and those using vertical magnetic field are labeled "B." Additionally, the time frame of the data is indicated: models using the same time as the target are labeled "0," those using data one hour earlier are labeled "-," and those using data one hour later are labeled "+." For example, "IB_-0" indicates a model trained using intensity

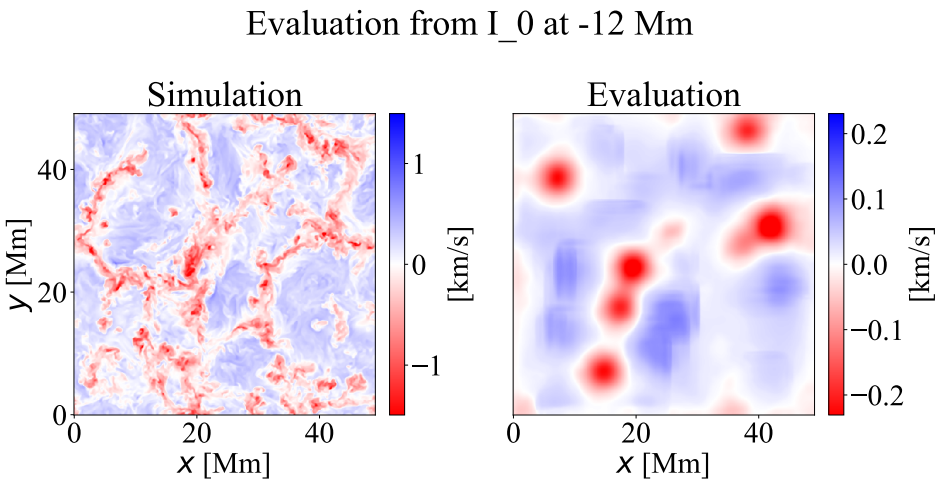


Figure4.15 Simulation data at a depth of 12 Mm and the velocity field evaluated solely from intensity at the target time

and magnetic field without the velocity field and using data from one hour prior and the target time, but not from one hour later.

Next, an example of the evaluation results is presented. The input data is shown in Figure 4.5, the same as in the previous section, with the physical quantities not used for training replaced by zero values. Figures 4.15 and 4.16 show the results of training for depths of 12 Mm and 5.4 Mm, using only snapshots of intensity, which provide the least amount of information. The correlation coefficients are 0.172 and 0.132, respectively. While most regions do not match, some areas with strong downflows appear to be detected. Despite being for different depths, both of these evaluation results indicate downflows in the same regions. These regions align with areas of strong magnetic fields visible in the magnetic field diagram in Figure 4.5 and appear as faint black dots in the intensity image.

The evaluation of internal structures using intensity suggests that the network may internally transform intensity into surface magnetic fields or equivalent information before making its evaluation.

Figures 4.17 and 4.18 show the results of training using snapshots of magnetic fields. The correlation coefficients are 0.321 and 0.314, indicating improved performance compared to using intensity. As shown in the magnetic field in Figure 4.5, it is that the regions evaluated as downflows correspond fairly well to regions of strong magnetic fields.

Figures 4.19 and 4.20 show the results for all input combinations. Table 4.6 and Figure 4.21 present the correlation coefficients for each combination. Learning with only intensity shows a large performance variation depending on depth. Learning with only velocity indicates

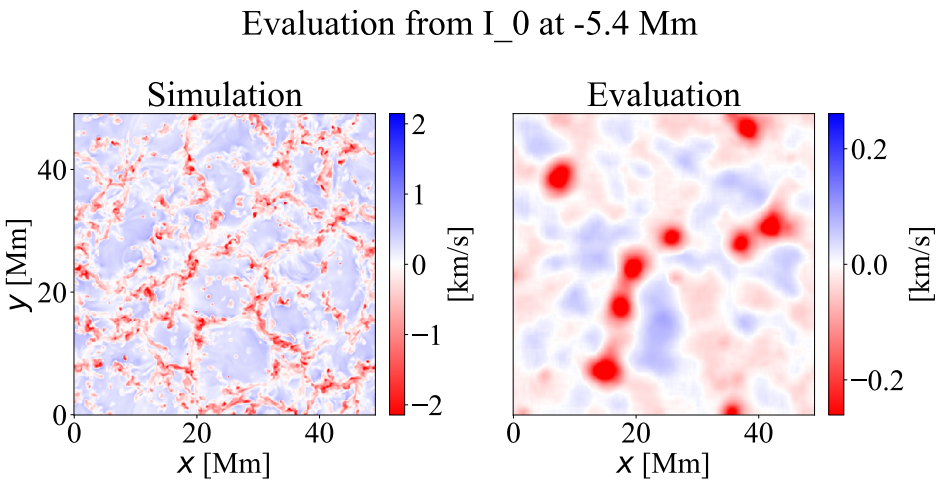


Figure4.16 Simulation data at a depth of 5.4 Mm and the velocity field evaluated solely from intensity at the target time

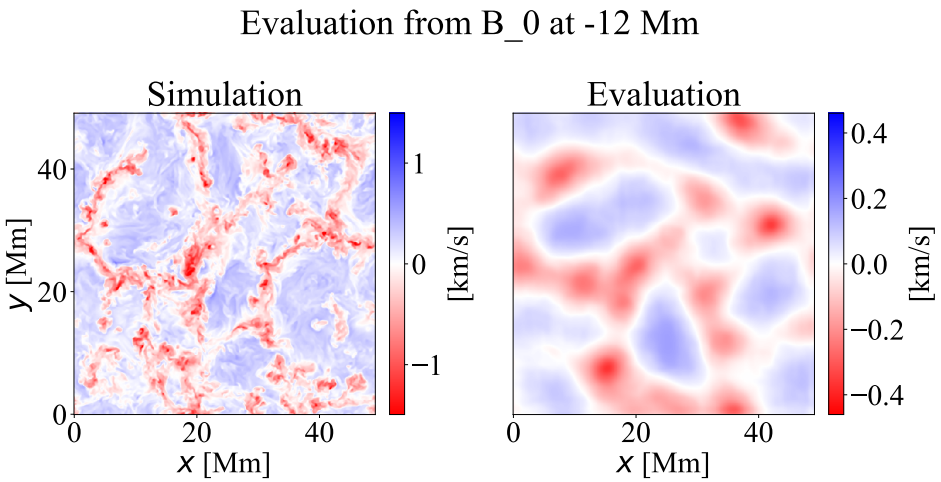


Figure4.17 Simulation data at a depth of 12 Mm and the velocity field evaluated solely from magnetic fields at the target time

significant performance improvement when time-shifted data is added. For snapshots, the evaluation of intensity and velocity is reversed at a depth of 12 Mm, with intensity becoming advantageous. Combining intensity and velocity fields improves performance, whereas combining magnetic field data with others does not result in performance improvement. In all combinations, using data from different times enhances performance. Adding data from one hour earlier increases the correlation coefficient by about 0.05, and adding data from one hour later further improves performance slightly, though the improvement is smaller when velocity is included.

Figures 4.22 and 4.23 show the coherence spectra at depths of 12 Mm and 5.4 Mm. Figure

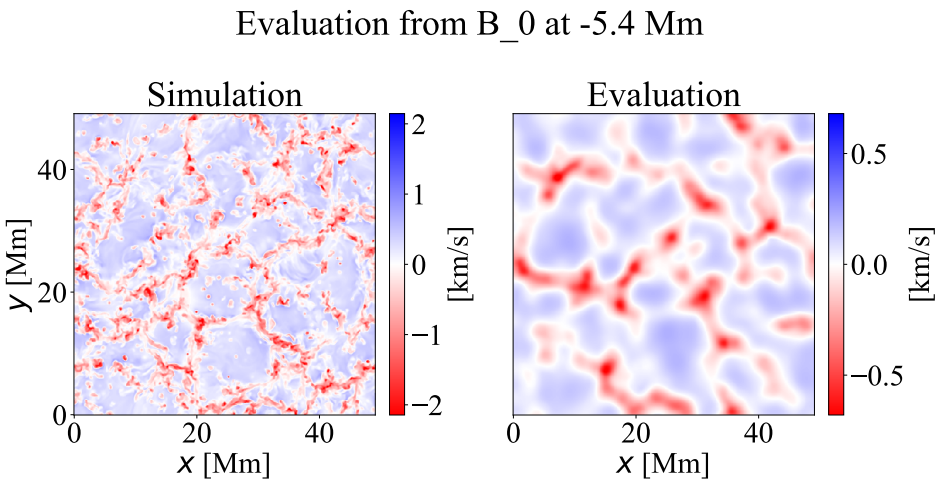


Figure4.18 Simulation data at a depth of 5.4 Mm and the velocity field evaluated solely from magnetic fields at the target time

	I	V	IV	B	IB	VB	IVB
0	0.172	0.169	0.255	0.321	0.320	0.320	0.320
-0	0.191	0.216	0.284	0.364	0.358	0.360	0.364
-0+	0.204	0.223	0.290	0.373	0.373	0.368	0.378

Table4.1 Correlation coefficient at a depth of -12 Mm (First Table)

	I	V	IV	B	IB	VB	IVB
0	0.132	0.151	0.242	0.312	0.314	0.314	0.314
-0	0.155	0.215	0.279	0.359	0.357	0.361	0.361
-0+	0.162	0.223	0.282	0.365	0.367	0.371	0.371

Table4.2 Correlation coefficient at a depth of -5.4 Mm (Second Table)

4.22 compares the performance for each input physical quantity when using data from all time points, while Figure 4.23 compares the performance for different input time configurations. In Figure 4.22, the performance improves across all scales in the same order as in Figure 4.21. Results using only the magnetic field and those using all parameters are generally similar, but the results using all parameters are slightly better. Figure 4.23 shows the performance for different input time configurations for several physical quantities. The performance improves as more step points are included for all cases. However, for inputs using intensity (I) or intensity and velocity (IV), the improvement from adding data one hour after the target time is minimal, and in some scales, the results are reversed.

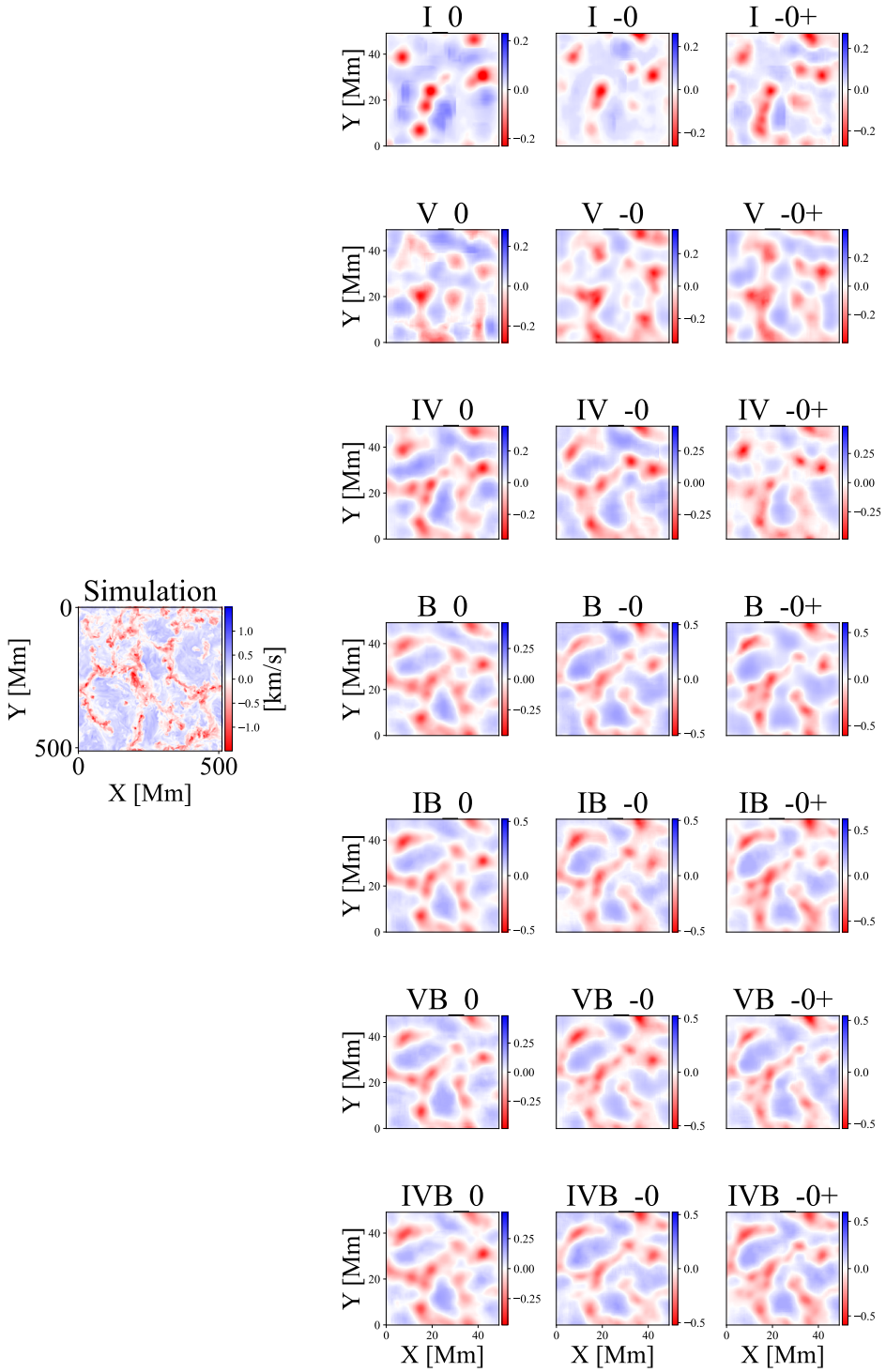


Figure4.19 Evaluated velocity field at a depth of 12 Mm. The color bar unit is $\mathrm{km\ s^{-1}}$.

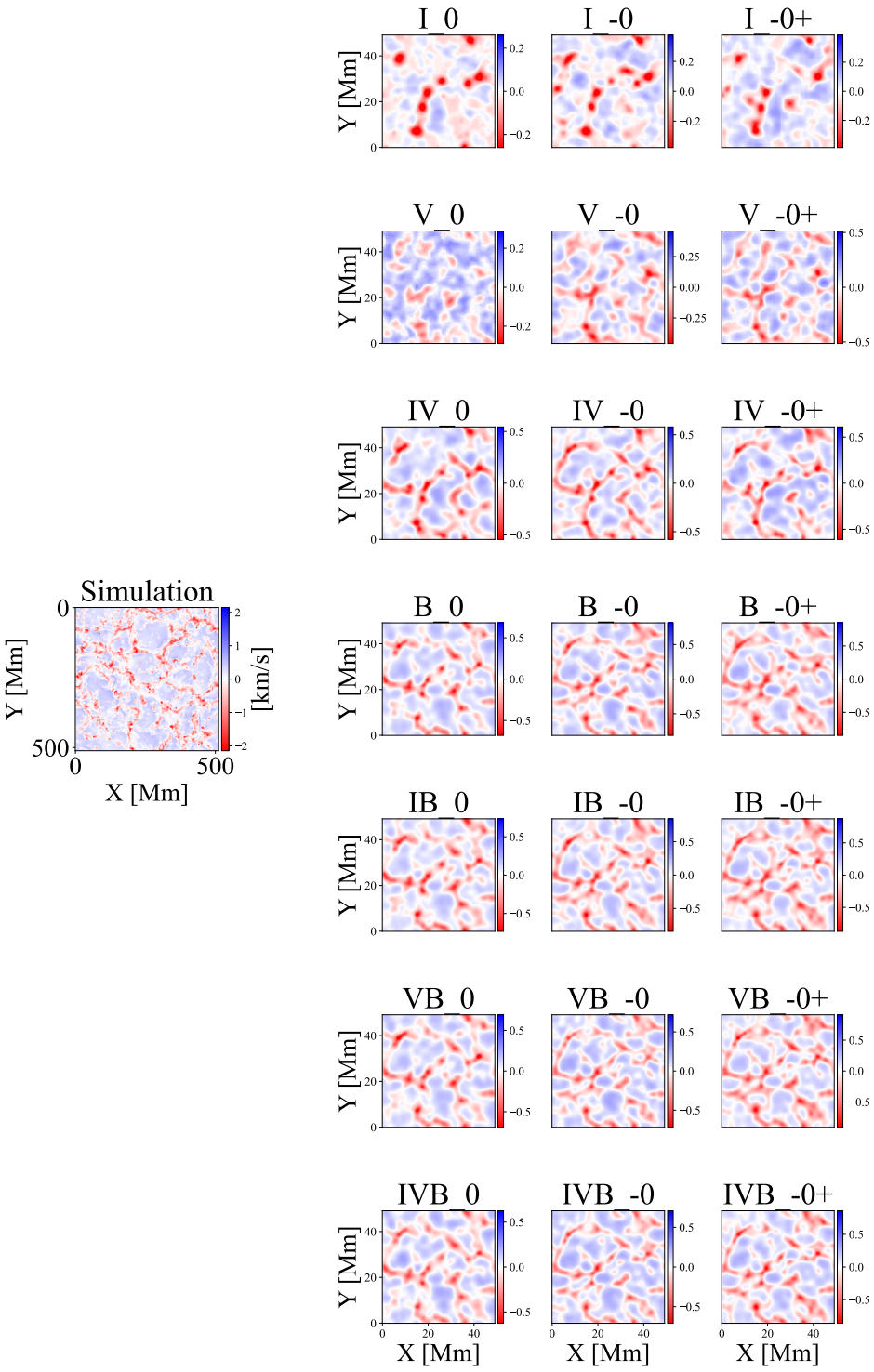


Figure4.20 Evaluated velocity field at a depth of 5.4 Mm. The color bar unit is $\mathrm{km\ s^{-1}}$.

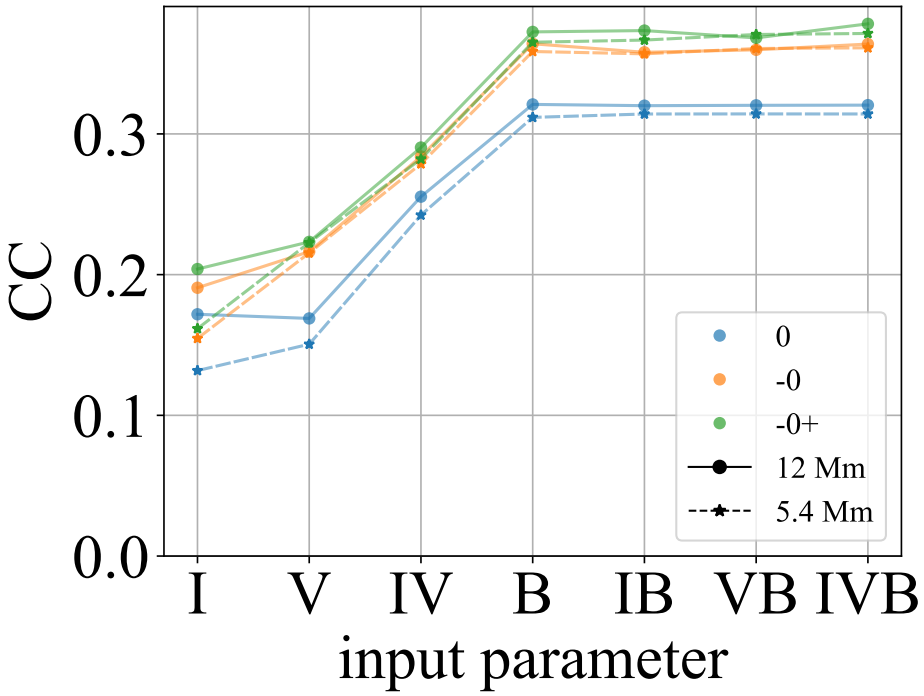


Figure4.21 Graph of correlation coefficients for each input. Results at a depth of 12 Mm from Table 4.6 are shown as solid lines, while results at a depth of 5.4 Mm are shown as dashed lines.

The difficulty of intensity in evaluating shallow regions can be attributed to the challenges in evaluating smaller scales, particularly at a depth of 5.4 Mm, where the convection scale is not effectively captured. Intensity relies on structures formed by the concentration of magnetic fields visible on the solar surface to infer the internal state, which limits its ability to evaluate smaller scales. When velocity fields are used as input, the evaluation likely relies on the relationship between surface and internal flows without relying on magnetic fields. Strong magnetic field lines are thought to correlate with internal flows, which explains the improved performance when using magnetic fields for evaluations. However, the lack of improvement when using intensity or velocity as input suggests that most of the information about the internal velocity field contained in intensity and surface velocity is already included in the magnetic field. Furthermore, while generally increasing the amount of input data for a neural network enhances evaluation performance, introducing irrelevant information or data whose content overlaps with other inputs can cause part of the network's capacity to be used for processing unrelated data, potentially reducing overall performance.

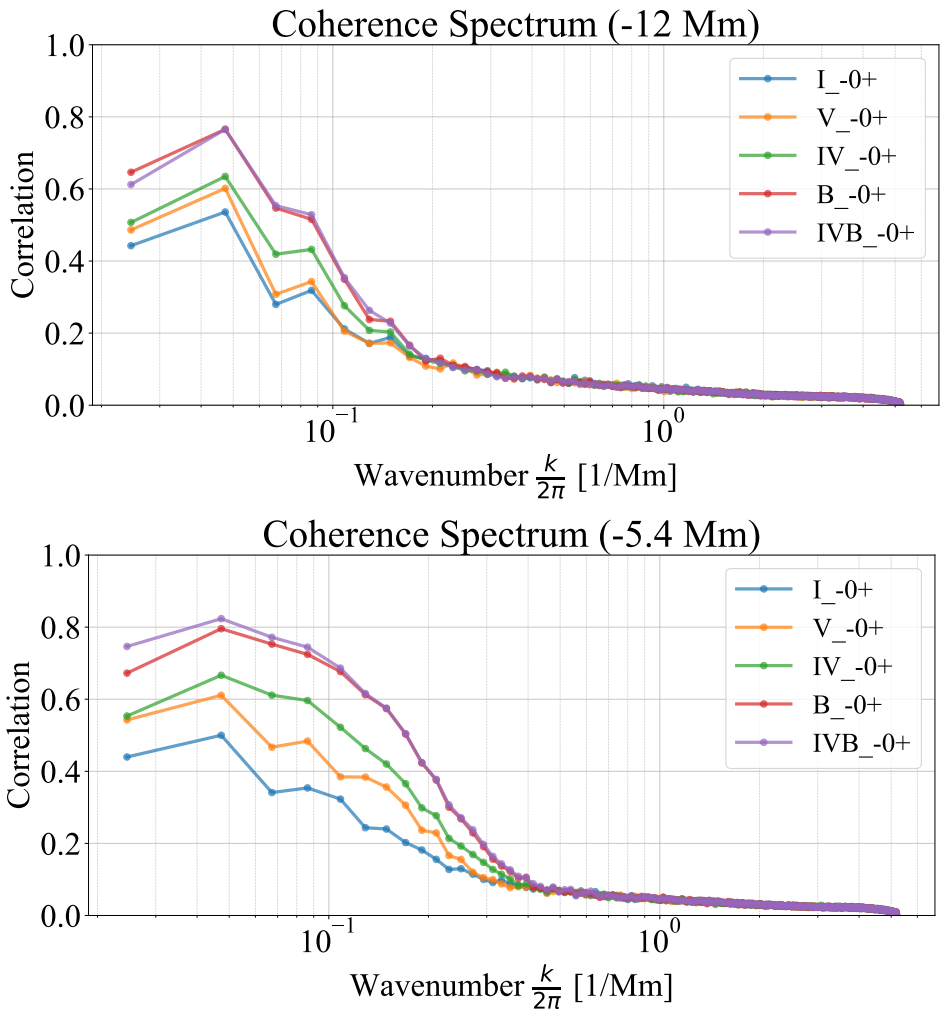


Figure4.22 Coherence spectra between the evaluated velocity field and the simulation results. The results when three time points (target time and one hour before and after) are used as input. Blue, orange, green, yellow, and purple lines represent evaluations using intensity, velocity, intensity and velocity, magnetic field, and all three parameters, respectively.

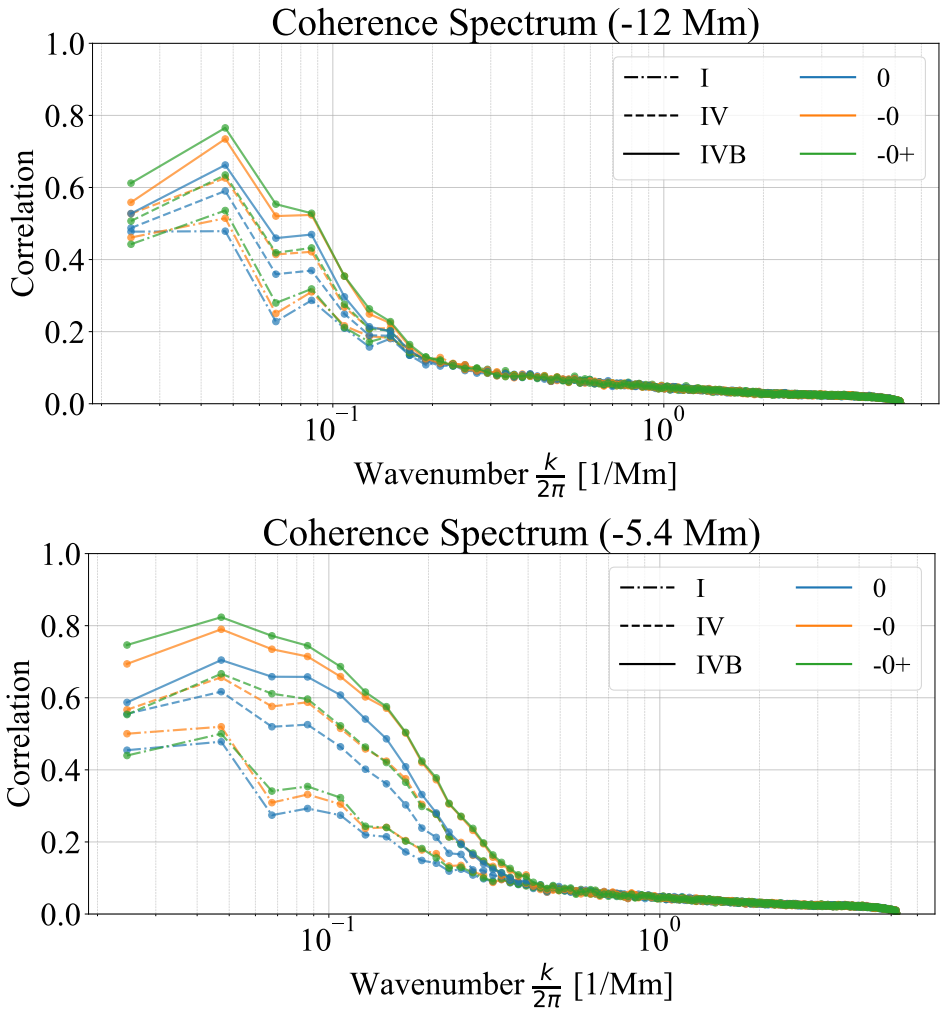


Figure4.23 Coherence spectra between the evaluated velocity field and the simulation results. The results of changing the input time points are shown for cases using only intensity, intensity velocity, and all three parameters. Blue lines indicate results using only the target time; orange lines indicate results using the target time and one hour before, and green lines indicate results using the target time and one hour before and after. Dotted lines represent results using intensity, dashed lines represent results using intensity and velocity, and solid lines represent results using all parameters.

Chapter 5

Summary and Discussion

5.1 Surface Velocity Field

We develop a method using neural networks to evaluate horizontal velocity fields, which are difficult to observe, by inputting relatively easy-to-observe physical quantities such as intensity, line-of-sight velocity, and line-of-sight magnetic field.

Training a neural network requires ground truth data. However, compared to intensity, line-of-sight velocity, and line-of-sight magnetic field, it is difficult to obtain high-accuracy observation data for horizontal velocity fields. Therefore, we prepare data for intensity, line-of-sight velocity, and line-of-sight magnetic fields that correspond to horizontal velocity fields using magnetohydrodynamic simulations. The magnetohydrodynamic code R2D2 is used for the simulations.

The network's evaluation performance improves by adding the vertical velocity field to the network that evaluates the horizontal velocity field using only intensity. On the other hand, adding the vertical magnetic field does not significantly improve network performance. Vertical and horizontal velocities are strongly related due to the basic convective structure of upflows and downflows. On the other hand, the magnetic field does not directly relate to thermal convection, and the information associated with the magnetic field in the horizontal velocity field can be sufficiently covered by the intensity and horizontal velocity field.

When evaluating the horizontal velocity field, the evaluation performance is highest when using intensity, vertical velocity field, and vertical magnetic field, with a correlation coefficient of 0.90 and an R2 score of 0.81. The comparison of the results shows that the overall shape of the evaluated field agrees well with the simulation. However, there is a tendency for larger errors in evaluating the velocity field at the granulation boundaries. When the network incorrectly evaluates the horizontal velocity field, it often underestimates the actual velocity.

In this study, the velocity field evaluation is performed by dividing it into two components, horizontal and vertical. Since the simulation area is symmetric concerning each horizontal direction, the average value of the horizontal velocity field is 0 $\mathrm{km\ s^{-1}}$. When the network fails to evaluate physical quantities, it tends to output values close to this average value of 0 to minimize error, resulting in underestimation. By evaluating quantities where the average is not 0, such as the magnitude of vectors, this tendency to underestimate can be addressed. Therefore, changing the approach from evaluating horizontal and vertical components separately to evaluating quantities like magnitude and angle might improve performance.

Fluid rising to the solar surface cools as opacity decreases in the upper layers, reducing its temperature and increasing its density, which eventually halts the upward motion. With nowhere to go, the fluid spreads outward, avoiding surrounding regions. Subsequently, the cooler fluid begins to converge and sink. The intensity is determined by the surface temperature, which is driven by density and pressure. On the other hand, the upward velocity is directly related to horizontal flows through fluid divergence and convergence. This "distance of information" likely contributes to the performance improvement when velocity fields are used as input. While surface magnetic fields are driven by horizontal flows, they are concentrated at the boundaries of granules due to the frozen-in condition of magnetic field lines and do not appear at the centers of granules. Thus, they do not lead to a significant improvement in overall evaluation metrics such as correlation coefficients.

When applying the trained network by simulation to observation data, the network outputs highly distorted results. This happens because the network overfits the fine features specific to the simulation data used for training and cannot adjust well to the observational data. The differences between small-scale structures in the simulation and observational data, due to the precision of observations, lead to discrepancies when the network tries to evaluate the horizontal velocity field, incorporating these structures. To make the simulation data more similar to the observational data, we apply the point spread function (PSF) of the Hinode and add random noise to eliminate small-scale structures. Training the network with this modified data leads to a correlation coefficient of 0.64 and an R2 score of 0.42 between the simulated and evaluated velocity fields.

If the network can incorporate small-scale structure information more effectively, the network's performance may improve. By further improving observation accuracy and capturing finer structures, the evaluation performance can also be enhanced.

When comparing the network's evaluation results to the LCT (Local Correlation Tracking), another method for evaluating horizontal velocity fields, the results differ significantly. The network shows much better agreement with the simulation than the LCT. The LCT tracks

visible changes and cannot track flow that does not accompany visible changes. Furthermore, the LCT struggles to evaluate velocity fields in granulation due to their characteristic structure. Given the conditions of this study, the neural network's evaluation of horizontal velocity fields outperforms the LCT.

Observations with Hinode are efficiently conducted by adjusting the observation cadence based on the target and purpose. However, methods like DeepVel rely on training with data of specific time intervals, making it difficult to evaluate from data with intervals significantly different from the trained ones. The method proposed in this study allows for flexibility in handling any cadence, enabling more information to be extracted from observational data. Since evaluations can be made from a single snapshot, this method is advantageous for real-time global solar observations aimed at space weather forecasting, as it allows evaluations even before a second image is acquired, avoiding bottlenecks in observation. Additionally, for high-resolution ground-based observations, where weather conditions may cause data gaps, this method can help mitigate the effect of such data loss.

5.2 Internal Velocity Field

In this study, a neural network is trained to evaluate the speed of upward flows inside the Sun from three observable surface quantities: intensity, line-of-sight velocity, and line-of-sight magnetic field. The training data used are generated by the radiation-magnetohydrodynamics simulation code R2D2, which simulates internal thermal convection velocities. While the network is able to roughly reproduce the internal velocity field at a depth of 12 Mm, reproducing small-scale structures proves difficult. The correlation coefficient at a depth of 5 Mm, where the convection structure becomes smaller, is lower than that at 12 Mm. Additionally, when compared with helioseismology using observational data, a correlation in large-scale structures is observed, indicating that the network's evaluated velocity field is not significantly inconsistent with reality. In comparison with helioseismology using simulation data, the neural network is slightly better at reproducing small-scale structures.

In shallower regions, evaluations are easier because of the stronger connection with surface physical quantities. However, as the scale of thermal convection becomes smaller, it becomes difficult to evaluate at effective scales. In this study, a time interval of 1 hour is adopted, but the time it takes for internal thermal convection to affect the surface or for surface convection to sink and affect the interior varies by depth. Therefore, there may be an appropriate time interval for each depth. Entering data at finer time intervals could improve performance, but increasing the time by an order of magnitude would not result in dramatic improvements,

and computational costs would increase. Therefore, appropriate inputs should be selected according to the purpose. In this study, simulations that match the resolution of SDO data are used as input data. However, even in the learning results where the raw simulation data are used as input, the correlation coefficient for depths below 5 Mm shows little improvement. Thus, even if the accuracy of observation data improves due to advances in observation technology, the current method is unlikely to achieve significant performance improvements.

Even when trained with high-resolution data, the accuracy does not improve, suggesting that significant performance improvements cannot be expected with this method, even if observational technology advances. Rather than focusing on capturing detailed structures, it would be more effective to learn on a wider scale to evaluate internal velocity fields on a global scale and predict the emergence of rising magnetic fields.

Helioseismology, which investigates the Sun's internal structure, requires the analysis of several hours of data. Therefore, when using helioseismology to understand the Sun's internal structure, it is unavoidable to encounter a delay of several hours. Additionally, the computational cost is high, requiring powerful computers to efficiently obtain global data. By extending this research, the use of neural networks for efficient processing can reduce the required data and computation time, thereby improving the monitoring of sunspot formation and other phenomena that impact Earth.

This study focuses on learning the structure of quiet regions, making it unsuitable for application to complex structures such as sunspots. Moreover, since the training data are based on the data reproduced observation data taken perpendicular to the surface, the method cannot be applied to regions observed obliquely, such as the solar limb. By improving the training dataset to include such data, the system aims to utilize real-time observations of the entire Sun to evaluate subsurface structures.

5.3 Future Prospects

By further developing this research, we aim to build a system that monitors the three-dimensional plasma motion throughout the solar convection zone. Explosive phenomena that influence the Earth through the solar wind and other processes are mainly caused by sunspot activity (Singh et al., 2010; Webb and Howard, 2012). Therefore, knowing in advance about sunspot formation and its parameters could be a useful tool for space weather forecasting.

Magnetic flux tubes inside the Sun rise due to magnetic pressure pushing out the plasma, reducing their density, which is called magnetic buoyancy. As a result, sunspots appear on the surface. Several simulations have been conducted to reproduce this emergence process.

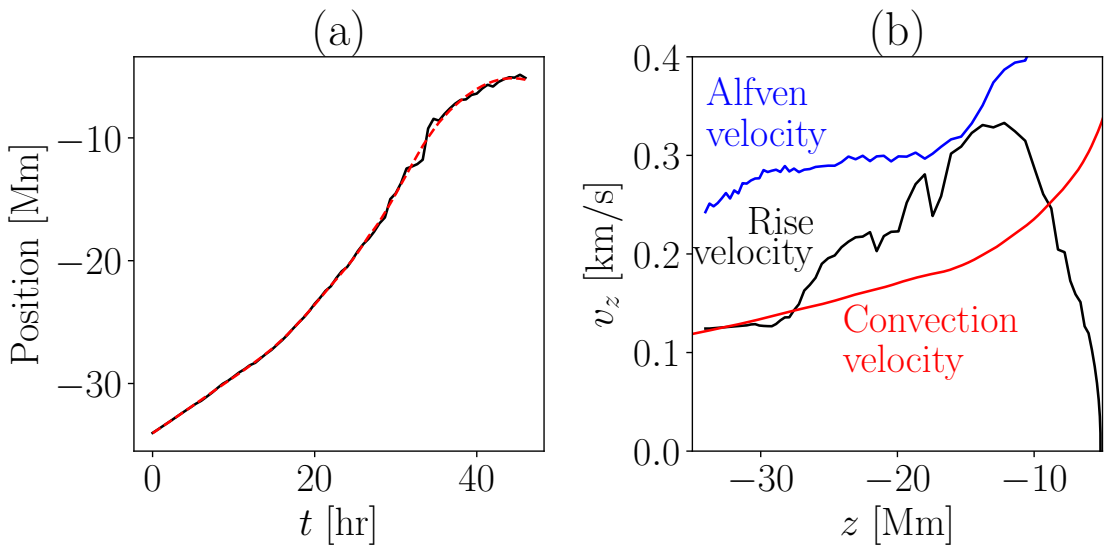


Figure5.1 Cited from Figure 9 of Hotta and Iijima (2020). The left panel shows the position of the flux tube over time, while the right panel presents the velocity at each depth.

Hotta and Iijima (2020) carried out a large-scale simulation that included the entire solar convection zone down to a depth of 200 Mm, placing a flux tube of about 10 Mm radius in the convective structure and examining its behavior. The results are shown in Figure 1.5. The flux tube, initially placed parallel to the surface, rises where there are upflows and sinks where there are downflows. Since the flux tube is influenced not only by convection but also by magnetic buoyancy, it moves faster than typical convective flows. Figure 5.1 shows (a) the temporal evolution of the flux tube position and (b) the velocity at different depths. Around a depth of 20 Mm, (b) shows that the upward velocity of the flux tube (black line) is several hundred $\mathrm{m\ s^{-1}}$ faster than the convective velocity (red line). Ilonidis et al. (2011) applied helioseismology to eight hours of observational data and investigated the solar interior. Figure 5.2 shows the result. They reported that, at depths of 42 Mm to 75 Mm, anomalies in travel time spanning approximately 35 Mm were detected 4–10 hours before emerging magnetic fields appeared. Based on this, they estimated the rise speed of magnetic fields to be 300–600 $\mathrm{m\ s^{-1}}$.

Since emerging magnetic field regions are on the order of tens of Mm, and velocity differences are around several hundred $\mathrm{m\ s^{-1}}$, they fall within the detectable range of our proposed method. By incorporating emerging magnetic field data into training, it may become possible to capture flux tube emergence-related flows.

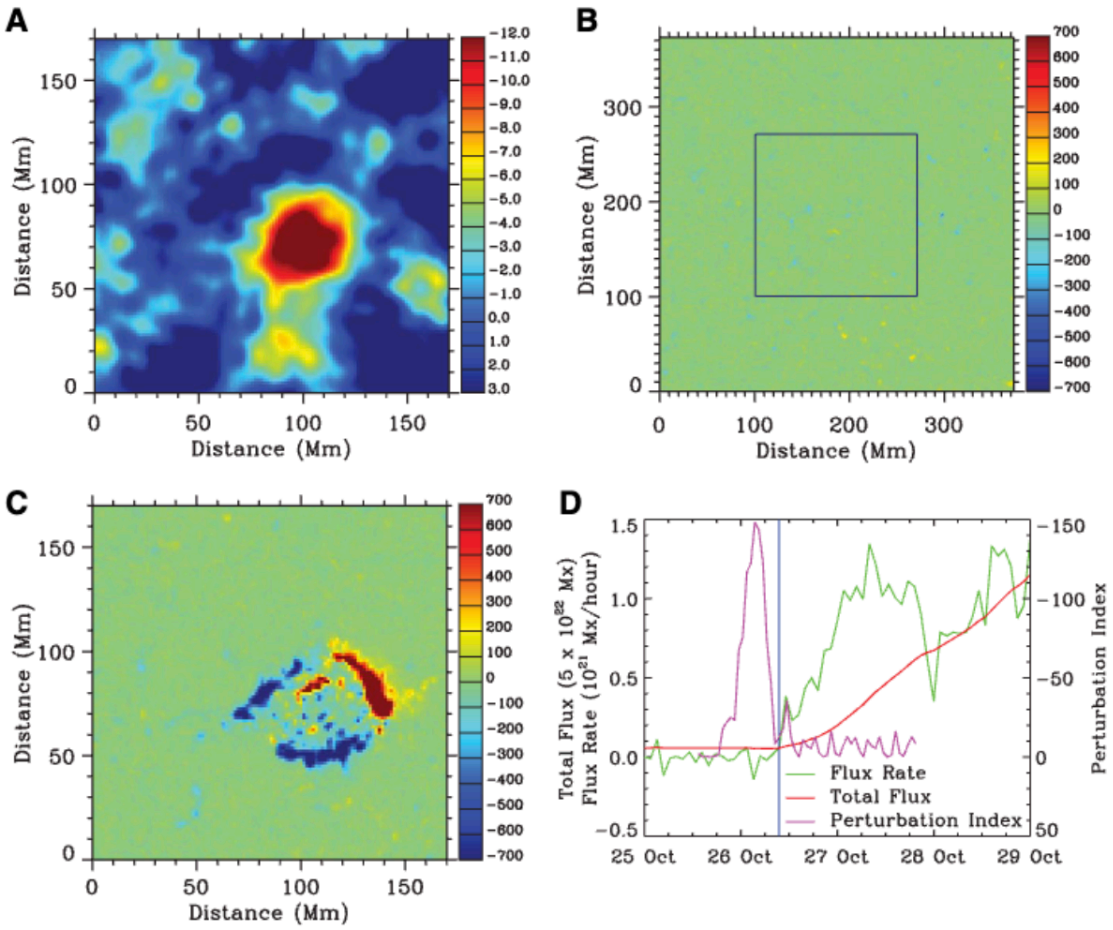


Figure5.2 Cited from Figure 2 of Ilonidis et al. (2011). The figure presents the distribution of travel time (a), the surface magnetic field at that moment (b), the surface magnetic field 24 hours later (c), and the time evolution of each value (d).

Additionally, horizontal flows associated with magnetic field emergence have been speculated. Fan et al. (2013) conducted a simulation replicating the entire Sun to examine flux tube motion. The results show that as a flux tube rises, it moves away from the rotational axis. Due to angular momentum conservation, it attempts to maintain its angular velocity, resulting in a retrograde flow of approximately 300 $\mathrm{m\ s^{-1}}$ in the opposite direction of rotation. Rempel and Cheung (2014) investigates the effects of artificially introducing a retrograde flow into a flux tube. Their findings indicate that the retrograde flow created an asymmetry in the time evolution of the flux tube on the rotation-facing and opposite sides, affecting the formation time of sunspots.

Detecting such flows could help identify precursors to sunspot formation and provide insights into their development time and behavior. Although the retrograde flow differs from

the target of this study as it represents a flow parallel to the horizontal surface, Chapter 3 demonstrates that horizontal velocity can be inferred from line-of-sight velocity and other surface observations. This suggests that deep learning-based methods could also be effective in detecting horizontal flows within the solar interior.

AppendixA

Fourth-Order Accurate Differentiation

A.1 Spatial Differentiation

The magnetohydrodynamic equations include spatial derivatives, but since they cannot be analytically solved, they are numerically computed using finite differences. The fourth-order accurate central difference used in this study for the spatial derivative of a physical quantity u at the i-th cell is expressed by the following equation:

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{1}{12\Delta x}\left(-u_{i+2} + 8u_{i+1} - 8u_{i-1} + u_{i-2}\right) \tag{A.1}$$

A.2 Temporal Differentiation

The temporal differentiation is performed as follows. When the equation to be solved for the physical quantity $\boldsymbol{U}$ is expressed as:

$$\frac{\partial \boldsymbol{U}}{\partial t} = \boldsymbol{R}(\boldsymbol{U}) \tag{A.2}$$

By calculating the following sequence, the physical quantity $\boldsymbol{U}_1$ at time $t + \Delta t$ can be

obtained from $\boldsymbol{U}_0$ at time t:

$$\boldsymbol{U}_{\frac{1}{4}} = \boldsymbol{U}_0 + \frac{\Delta t}{4}\boldsymbol{R}(\boldsymbol{U}_0) \tag{A.3}$$

$$\boldsymbol{U}_{\frac{1}{3}} = \boldsymbol{U}_0 + \frac{\Delta t}{3}\boldsymbol{R}(\boldsymbol{U}_{\frac{1}{4}}) \tag{A.4}$$

$$\boldsymbol{U}_{\frac{1}{2}} = \boldsymbol{U}_0 + \frac{\Delta t}{2}\boldsymbol{R}(\boldsymbol{U}_{\frac{1}{3}}) \tag{A.5}$$

$$\boldsymbol{U}_1 = \boldsymbol{U}_0 + \Delta t\boldsymbol{R}(\boldsymbol{U}_{\frac{1}{2}}) \tag{A.6}$$

AppendixB

Network structure and Update

This section explains the network structure and the specific method of optimization.

B.1 Gradient Descent Method

Network parameters are optimized to minimize the error between the network output and the expected correct output data. In models with fewer variables, brute-force parameter search or least squares methods using second-order derivatives are often used. However, because neural networks have a large number of parameters, the gradient descent method is used.

In the gradient descent method, the first-order partial derivative is taken for each parameter, and the parameters are fine-tuned in the direction that reduces the error function. Let the change in network parameter w be represented as Δw, then it can be expressed as:

$$\Delta w = -\beta \frac{\partial L}{\partial w} \tag{B.1}$$

Here, L is the error function that represents the degree of agreement between the network's output and the correct data. β is called the learning rate and controls the update amount.

B.2 Neural Network structure and Gradient Calculation

A neural network takes the input at a layer as $\boldsymbol{x}(n)$ and outputs $\boldsymbol{x}(n+1)$. The function determining the j-th element of the output in that layer is denoted as $f_j^{(n)}$, and the parameters of the function are represented by $\boldsymbol{w}(n)$, which can be expressed as a recurrence relation:

$$\boldsymbol{x}_j^{(n+1)} = f_j^{(n)}(\boldsymbol{x}^{(n)}, \boldsymbol{w}^{(n)}) \tag{B.2}$$

Let N be the number of layers, with the final output denoted as $x^{(N)}$. The error function

between the network's output and the expected correct data is expressed as $L(\boldsymbol{x}^{(N)})$.

$\frac{\partial L}{\partial x^{(n)}}$ represents a vector whose elements are the partial derivatives with respect to each component.

This allows $\frac{\partial L}{\partial \boldsymbol{x}^{(n)}}$ to be computed in reverse order from the output layer. Using $\frac{\partial x_j^{(n+1)}}{\partial \boldsymbol{x}^{(n)}}$, we have:

$$\frac{\partial L}{\partial \boldsymbol{x}^{(n)}} = \sum_j \frac{\partial L}{\partial x_j^{(n+1)}} \frac{\partial x_j^{(n+1)}}{\partial \boldsymbol{x}^{(n)}} \tag{B.3}$$

By preparing formulas for $\frac{\partial L}{\partial \boldsymbol{x}^{(n)}}$ and $\frac{\partial \boldsymbol{x}^{(n)}}{\partial \boldsymbol{w}^{(n)}}$ from the network structure in advance, we can compute $\frac{\partial L}{\partial \boldsymbol{w}^{(n)}}$ to obtain the gradient of each parameter. This is known as backpropagation Rumelhart et al. (1986). The network is updated using this gradient.

Below, we present the specific calculation formulas and their partial derivatives for the network structure used in this study.

B.2.1 Skip Connection

Skip connection is a method that connects the outputs from different layers to the next layer He et al. (2016). There are several connection methods, such as addition, but in this study, we simply concatenate the vectors. Denoting the layer numbers as $n > m$:

$$\boldsymbol{x}^{(n+1)} = [\boldsymbol{x}^{(n)}, \boldsymbol{x}^{(m)}] \tag{B.4}$$

In backpropagation, we can compute:

$$\frac{\partial L}{\partial \boldsymbol{x}^{(m)}} = \sum_j \left(\frac{\partial L}{\partial x_j^{(n+1)}} \frac{\partial x_j^{(n+1)}}{\partial \boldsymbol{x}^{(m)}} + \frac{\partial L}{\partial x_j^{(m+1)}} \frac{\partial x_j^{(m+1)}}{\partial \boldsymbol{x}^{(m)}} \right) \tag{B.5}$$

B.2.2 Error Function

The error function used in this study is the mean squared error. The value $\boldsymbol{x}^{(N)}$ in the final N-th layer, which has a size of I, is compared with the ground truth data $\boldsymbol{y}$ prepared for training, and it is expressed as:

$$L(\boldsymbol{x}^{(N)}) = \frac{1}{I} \sum_{i=1}^{I} (x_i^{(N)} - y_i)^2 \tag{B.6}$$

The partial derivative is:

$$\frac{\partial L}{\partial x_i^{(N)}} = \frac{2}{I}(x_i^{(N)} - y_i) \tag{B.7}$$

B.2.3 Activation Function

The ReLU function used in this study is expressed as Nair and Hinton (2010). The output $\boldsymbol{x}^{(n+1)}$ and the input $\boldsymbol{x}^{(n)}$ are vectors of the same size.

$$x_i^{(n+1)} = \begin{cases} x_i^{(n)} & \text{if } x_i^{(n)} \geq 0 \\ 0 & \text{if } x_i^{(n)} < 0 \end{cases} \tag{B.8}$$

In backpropagation, we have:

$$\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}} = \begin{cases} \delta_{ij} & \text{if } x_i^{(n)} \geq 0 \\ 0 & \text{if } x_i^{(n)} < 0 \end{cases} \tag{B.9}$$

Here, δ_{ij} is the Kronecker delta.

B.2.4 Fully Connected Network

A fully connected layer is expressed by the following equation. Let W be the weight tensor and $\boldsymbol{b}$ the bias vector:

$$\boldsymbol{x}^{(n+1)} = W\boldsymbol{x}^{(n)} + \boldsymbol{b} \tag{B.10}$$

For each parameter, we have:

$$\frac{\partial x_j^{(n+1)}}{\partial w_{ij}^{(n)}} = x_i^{(n)} \tag{B.11}$$

$$\frac{\partial x_j^{(n+1)}}{\partial b_i^{(n)}} = \delta_{ij} \tag{B.12}$$

For backpropagation, the partial derivative is:

$$\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}} = w_{ij} \tag{B.13}$$

This formula can also be applied to tensors with multiple dimensions. In such cases, for a specific dimension's i-th element, the tensor, which is one dimension smaller, uses the scalar weight w_{ij} and bias b_j:

$$\boldsymbol{x}_j^{(n+1)} = w_{ij}\boldsymbol{x}_i^{(n)} + b_j \tag{B.14}$$

For w_{ij}, we sum all elements of $\boldsymbol{x}_i^{(n)}$:

$$\frac{\partial x_j^{(n+1)}}{\partial w_{ij}^{(n)}} = \delta_{ij} \sum_k x_{i,k}^{(n)} \tag{B.15}$$

B.2.5 Convolutional Networks

Convolutional neural networks can be expressed as follows LeCun et al. (1998). Here, x is a two-dimensional matrix with dimensions $K \times L$.

$$x_{kl}^{(n+1)} = \sum_{i,j} w_{ij} x_{k+i,l+j}^{(n)} \tag{B.16}$$

This means the computation is performed only over the range of neighboring i and j. In this study, a 3×3 convolution is used, so $i, j = -1, 0, 1$.

The partial derivative is:

$$\frac{\partial x_{kl}^{(n+1)}}{\partial w_{ij}^{(n)}} = x_{k+i,l+j}^{(n)} \tag{B.17}$$

Also,

$$\frac{\partial x_{kl}^{(n+1)}}{\partial x_{k+i,l+j}^{(n)}} = w_{ij} \tag{B.18}$$

holds, but outside the range of $i, j = -1, 0, 1$, it becomes 0.

In many cases, the input to the layer consists of multiple sets of images. In particular, for input data, many image recognition problems use RGB channels, whereas this study uses two-dimensional distributions of multiple physical quantities. In this case, the computation is performed channel-wise as in Equation B.14.

B.2.6 Max Pooling

Max pooling is an operation where $x(n)$ is a matrix with size $K \times L$, and the matrix has a size of $s \times s$. If s is not an integer, $K \times L$ is extended to be a multiple of s. Often, zero

padding is used for this extension, but in this study, the input and output sizes are adjusted to be multiples of s.

$$x_{kl}^{(n+1)} = \max_{0 \leq i < s, 0 \leq j < s} x_{sk+i,sl+j}^{(n)} \tag{B.19}$$

The values $(i(k,l), j(k,l))$ that gives the maximum are recorded, and the backpropagation is calculated as:

$$\frac{\partial x_{kl}^{(n+1)}}{\partial x_{sk+i,sl+j}^{(n)}} = \begin{cases} 1 & \text{if } i = i(k,l), j = j(k,l) \\ 0 & \text{otherwise} \end{cases} \tag{B.20}$$

B.2.7 Upsampling

Upsampling is a layer that expands the value of one pixel into an $s \times s$ region, increasing the image size.

$$x_{sk+i,sl+j}^{(n+1)} = x_{kl}^{(n)} \quad \text{for } 0 \leq i, j < s \tag{B.21}$$

The backpropagation is:

$$\frac{\partial x_{sk+i,sl+j}^{(n+1)}}{\partial x_{kl}^{(n)}} = \begin{cases} 1 & \text{for } 0 \leq i, j < s \\ 0 & \text{otherwise} \end{cases} \tag{B.22}$$

B.2.8 Batch Normalization

Batch normalization is a technique that normalizes the input of each layer, preventing extreme values in the intermediate layers, stabilizing the learning process, and accelerating convergence Ioffe and Szegedy (2015).

The forward pass for batch normalization is expressed as follows. Let the input be $\boldsymbol{x}^{(n)}$ and the output be $\boldsymbol{x}^{(n+1)}$.

First, calculate the mean and variance of the mini-batch.

$$\mu = \frac{1}{m} \sum_{k=1}^{m} x_k^{(n)} \tag{B.23}$$

$$\sigma^2 = \frac{1}{m} \sum_{k=1}^{m} \left(x_k^{(n)} - \mu \right)^2 \tag{B.24}$$

Here, m is the size of the mini-batch.

Next, normalize the input:

$$x_j^{(n+1)} = \frac{x_j^{(n)} - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{B.25}$$

Here, ϵ is a small constant for numerical stability.

To compute $\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}}$, the following calculations are performed.

Let $s = \sqrt{\sigma^2 + \epsilon}$ for the denominator:

$$x_j^{(n+1)} = \frac{x_j^{(n)} - \mu}{s} \tag{B.26}$$

Since μ and s depend on all $x_i^{(n)}$, the partial derivative of $x_j^{(n+1)}$ with respect to all $x_i^{(n)}$ must be taken.

The partial derivative of $x_j^{(n+1)}$ with respect to $x_i^{(n)}$ is:

$$\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}} = \frac{1}{s}\left(\delta_{ij} - \frac{\partial \mu}{\partial x_i^{(n)}}\right) - \frac{x_j^{(n)} - \mu}{s^2}\frac{\partial s}{\partial x_i^{(n)}} \tag{B.27}$$

Here, δ_{ij} is the Kronecker delta.

The partial derivative of μ is:

$$\frac{\partial \mu}{\partial x_i^{(n)}} = \frac{1}{m} \tag{B.28}$$

Next, the partial derivative of s is found by first calculating the partial derivative of σ^2:

$$\frac{\partial \sigma^2}{\partial x_i^{(n)}} = \frac{2}{m}\left(x_i^{(n)} - \mu\right) - \frac{2}{m}\sum_{k=1}^{m}\left(x_k^{(n)} - \mu\right)\frac{\partial \mu}{\partial x_i^{(n)}} = \frac{2}{m}\left(x_i^{(n)} - \mu\right) \tag{B.29}$$

Since $\sum_{k=1}^{m}\left(x_k^{(n)} - \mu\right) = 0$, the second term becomes zero.

The partial derivative of s is:

$$\frac{\partial s}{\partial x_i^{(n)}} = \frac{1}{2s}\frac{\partial \sigma^2}{\partial x_i^{(n)}} = \frac{1}{s}\cdot\frac{1}{m}\left(x_i^{(n)} - \mu\right) \tag{B.30}$$

Using these, $\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}}$ becomes:

$$\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}} = \frac{1}{s}\left(\delta_{ij} - \frac{1}{m}\right) - \frac{\left(x_j^{(n)} - \mu\right)}{s^2}\cdot\frac{1}{s}\cdot\frac{1}{m}\left(x_i^{(n)} - \mu\right) \tag{B.31}$$

Simplifying:

$$\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}} = \frac{1}{s}\left(\delta_{ij} - \frac{1}{m} - \frac{\left(x_j^{(n)} - \mu\right)\left(x_i^{(n)} - \mu\right)}{m\left(\sigma^2 + \epsilon\right)}\right) \tag{B.32}$$

Finally:

$$\frac{\partial x_j^{(n+1)}}{\partial x_i^{(n)}} = \frac{1}{\sqrt{\sigma^2 + \epsilon}}\left(\delta_{ij} - \frac{1}{m} - \frac{\left(x_j^{(n)} - \mu\right)\left(x_i^{(n)} - \mu\right)}{m\left(\sigma^2 + \epsilon\right)}\right) \tag{B.33}$$

Thus, the partial derivative of the output $x_j^{(n+1)}$ with respect to the input $x_i^{(n)}$ in the batch normalization layer is obtained, which can be used in backpropagation for learning.

Generally, scaling and shifting are applied after normalization:

$$x_j^{(n+2)} = \gamma x_j^{(n+1)} + \beta \tag{B.34}$$

Here, γ and β are trainable parameters.

The partial derivatives of $x_j^{(n+2)}$ with respect to γ, β, and the input $x_j^{(n+1)}$ are as follows.

Partial derivative with respect to γ:

$$\frac{\partial x_j^{(n+2)}}{\partial \gamma} = x_j^{(n+1)} \tag{B.35}$$

Partial derivative with respect to β:

$$\frac{\partial x_j^{(n+2)}}{\partial \beta} = 1 \tag{B.36}$$

Partial derivative with respect to the input $x_j^{(n+1)}$:

$$\frac{\partial x_j^{(n+2)}}{\partial x_j^{(n+1)}} = \gamma \tag{B.37}$$

Using these partial derivatives, the gradients for backpropagation can be calculated.

B.3 Adam

Simple gradient descent has a drawback. As the learning process progresses and the error function approaches its local minimum, the parameters may overshoot the optimal values and keep oscillating near the minimum without properly settling into the valley. To solve this,

the momentum method was devised, where the change in parameters Δw^t at the tth step is updated as:

$$\Delta w^t = -(1-\rho)\,\beta\frac{\partial L}{\partial w} + \rho\Delta w^{t-1} \tag{B.38}$$

Here, ρ is a parameter that determines the influence of momentum. If the parameters oscillate near the bottom of the error function's valley, the sign of the derivative changes from the previous step. By adding the previous update amount to the current derivative of the error function, the momentum cancels out when the direction of the decrease in the error function differs from the previous step, solving this issue.

Gradient descent has another drawback. If the learning rate β is too large, the updates overshoot multiple local minima, preventing convergence. Conversely, if it is too small, learning progresses too slowly. Moreover, in cases where the gradient of the error is large, convergence is faster, while in cases where it is small, convergence is slower, leading to inefficiency due to varying optimization speeds for different parameters. AdaGrad was developed to solve this problem. The AdaGrad update equation is expressed using the cumulative sum of gradient magnitudes s^t:

$$s^t = s^{t-1} + \left(\frac{\partial L}{\partial w}\right)^2 \tag{B.39}$$

The update equation becomes:

$$\Delta w^t = -\frac{\beta}{\sqrt{s^t}+\epsilon}\frac{\partial L}{\partial w} \tag{B.40}$$

The ϵ is a small amount to avoid zero division. By dividing the network's update amount by the cumulative gradient, the update amount decreases over time, allowing finer adjustments. Additionally, larger gradients result in larger reductions, correcting the optimization speed differences among parameters.

AdaGrad also has limitations. When approaching saddle points where the gradient is small but not a local minimum, the update speed slows down, preventing the parameters from reaching the local minimum. RMSprop modifies AdaGrad by updating s^t as:

$$s^t = \sigma s^{t-1} + (1-\sigma)\left(\frac{\partial L}{\partial w}\right)^2 \tag{B.41}$$

By applying a parameter σ smaller than 1 when updating s^t, the influence of earlier updates is reduced, allowing the learning rate to increase and progress when learning stagnates. The update Δw^t follows the same equation as B.40.

The optimization method used in this study, Adam, combines all of these techniques and is currently one of the most widely used methods. In equation form, it is expressed as:

$$m^t = -(1-\rho)\beta\frac{\partial L}{\partial w} + \rho\Delta w^{t-1}, \tag{B.42}$$

$$s^t = \sigma s^{t-1} + (1-\sigma)\left(\frac{\partial L}{\partial w}\right)^2, \tag{B.43}$$

$$\hat{m}^t = \frac{m^t}{1-\rho^t}, \tag{B.44}$$

$$\hat{s}^t = \frac{s^t}{1-\sigma^t}, \tag{B.45}$$

$$\Delta w^t = -\frac{\alpha}{\sqrt{\hat{s}^t}+\epsilon}\hat{m}^t. \tag{B.46}$$

Here, m^t is momentum. Additionally, the parameters ρ, σ, α, and β are called hyperparameters, which cannot be automatically determined. These must be manually adjusted according to the problem and the learning results. In Adam, bias-corrected terms are introduced to address the underestimation of m^t and s^t in the early stages of training. By applying these bias corrections, the estimates of the first and second moments at the early steps become more accurate, leading to a more stable convergence.

AppendixC

Observation method

C.1 Local correlation tracking(LCT)

C.1.1 Fourier Local correlation tracking(FLCT)

In this study, we use the FLCT code provided by Fisher and Welsch (2008). The calculation method for this technique is described below. Two images, $I_1(x, y)$ and $I_2(x, y)$, taken with a time difference δ, at the same location, are prepared. To prevent misjudgments regarding structures that are far away but similar in shape, a Gaussian filter σ centered at (x_i, y_i) is applied.

$$S_1(x, y) = I_1(x, y)e^{-\left[(x-x_i)^2+(y-y_i)^2\right]/\sigma^2} \tag{C.1}$$

$$S_2(x, y) = I_2(x, y)e^{-\left[(x-x_j)^2+(y-y_j)^2\right]/\sigma^2} \tag{C.2}$$

The cross-correlation at two points separated by $(\delta x, \delta y)$ is then calculated.

$$C(\delta x, \delta y) = \iint dx\, dy\, S_1^*(x, y)S_2(x - \delta x, y - \delta y) \tag{C.3}$$

The $(\delta x, \delta y)$ that maximizes this value is defined as the tentative displacement distance of the region centered at (x_i, y_j). Fisher and Welsch (2008) speeds up this cross-correlation calculation using a fast Fourier transformation. The displacement is then determined with sub-pixel accuracy by using a second-order Taylor expansion centered at (x_m, y_n). The

quadratic approximation function of $C(x, y)$, denoted as f_r, is

$$f_r(x,y) \equiv C(x_m, y_n) + \frac{\partial f}{\partial x}(x - x_m) + \frac{\partial f}{\partial y}(y - y_n) + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(x - x_m)^2 + \frac{\partial^2 f}{\partial x \partial y}(x - x_m)(y - y_n) + \frac{1}{2}\frac{\partial^2 f}{\partial y^2}(y - y_n)^2 \tag{C.4}$$

The peak of this function can be analytically derived as:

$$x_{\max} - x_m = \frac{\frac{\partial^2 f}{\partial y^2}\frac{\partial f}{\partial x} - \frac{\partial^2 f}{\partial x \partial y}\frac{\partial f}{\partial y}}{\left(\frac{\partial^2 f}{\partial x^2}\right)\left(\frac{\partial^2 f}{\partial y^2}\right) - \left(\frac{\partial^2 f}{\partial x \partial y}\right)^2} \tag{C.5}$$

$$y_{\max} - y_n = \frac{\frac{\partial^2 f}{\partial x^2}\frac{\partial f}{\partial y} - \frac{\partial^2 f}{\partial x \partial y}\frac{\partial f}{\partial x}}{\left(\frac{\partial^2 f}{\partial x^2}\right)\left(\frac{\partial^2 f}{\partial y^2}\right) - \left(\frac{\partial^2 f}{\partial x \partial y}\right)^2} \tag{C.6}$$

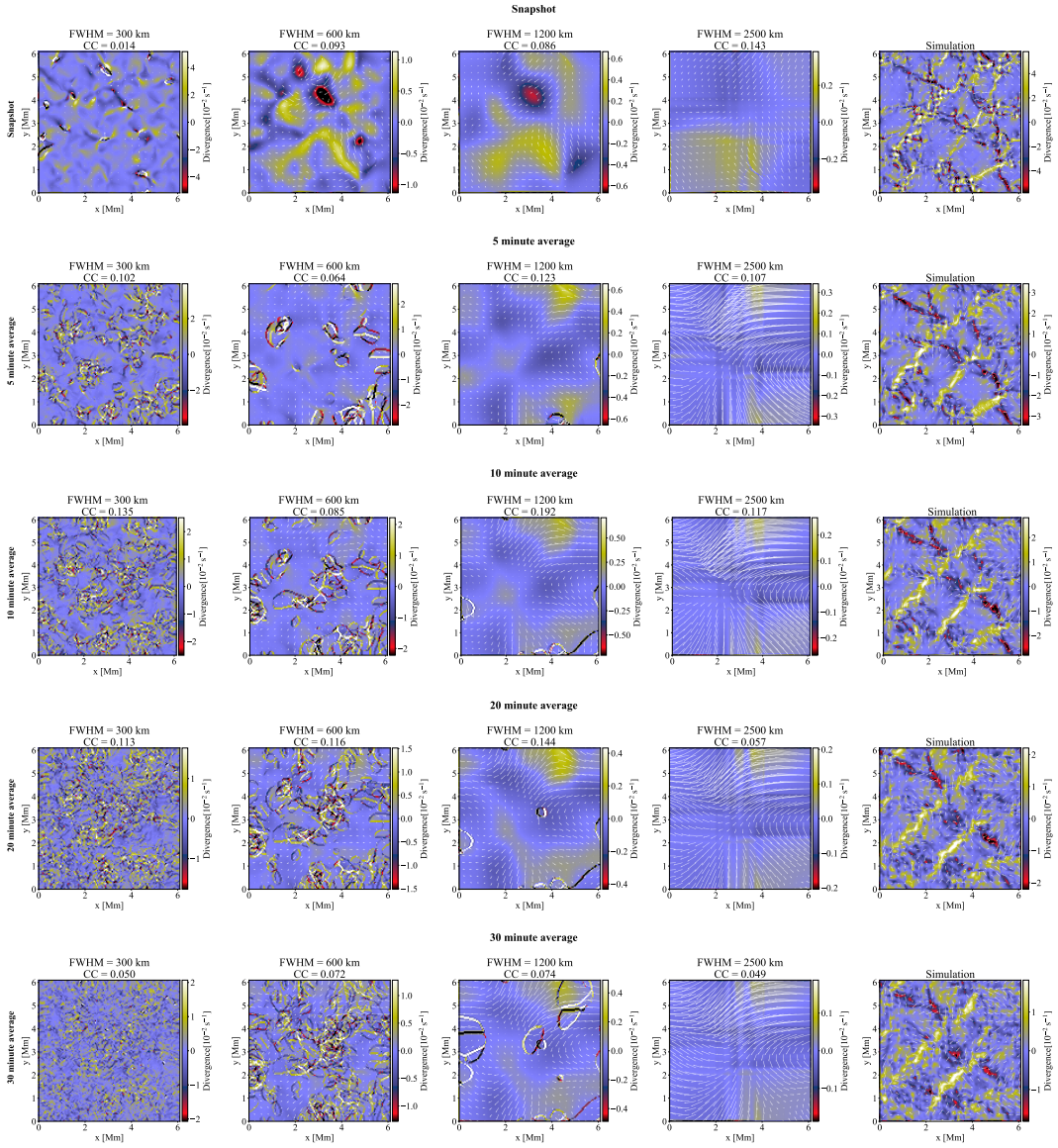
The partial derivatives appearing in these equations are calculated using a second-order finite difference method.

C.1.2 LCT Parameter Survey

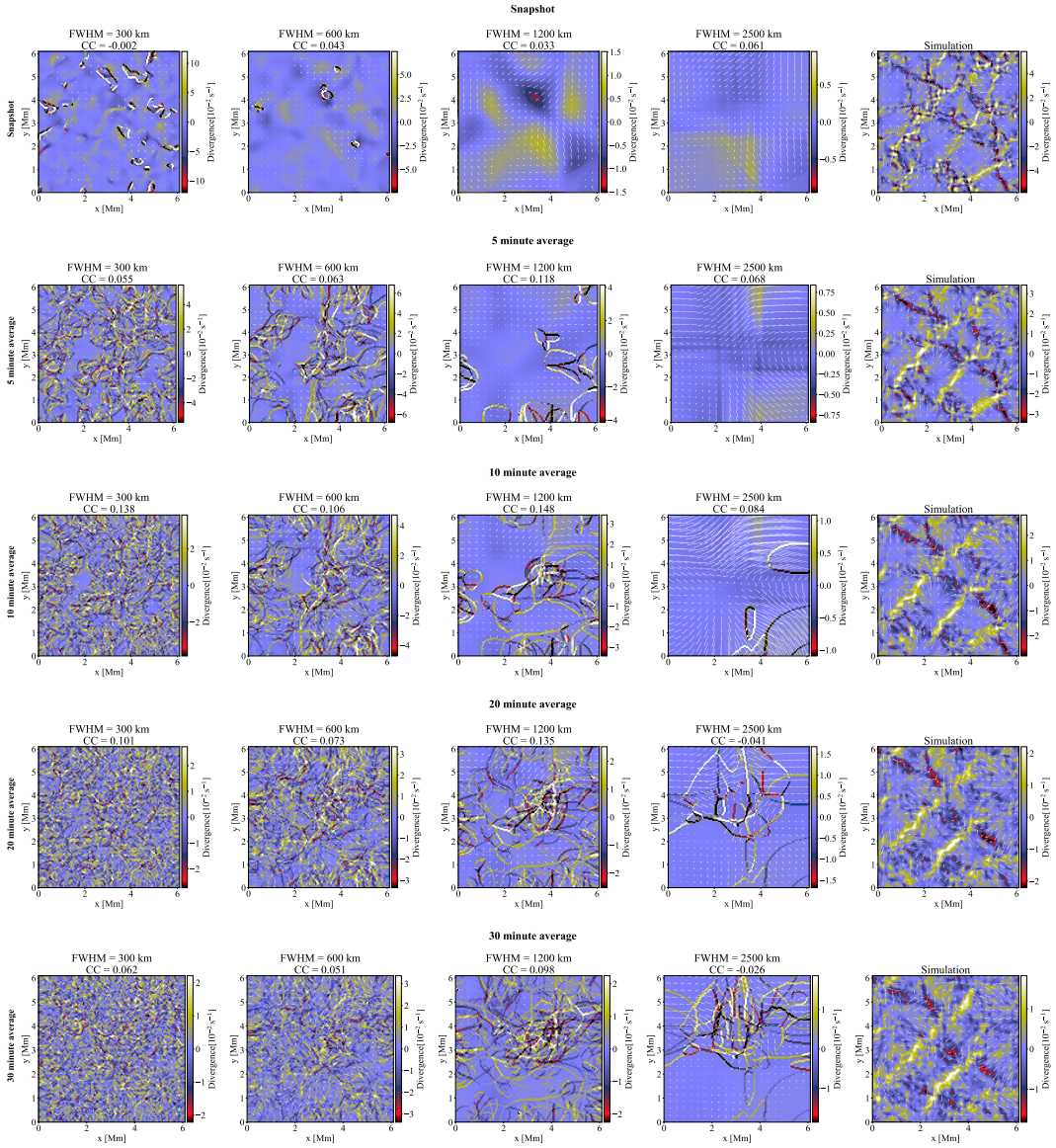
This appendix presents the parameter survey for LCT (Local Correlation Tracking).

Figures (image30), (image60), and (image120) show the horizontal divergence of the velocity field obtained by LCT with 30-second, 60-second, and 120-second intervals, respectively. The first, second, third, and fourth rows represent time-averaged data for 5 minutes, 10 minutes, 20 minutes, and 30 minutes, respectively. The first, second, third, and fourth columns represent results with FWHM values of 300, 600, 1200, and 2500 km, respectively. The rightmost column shows the simulated velocity field. The correlation coefficient (CC) calculated using the simulated velocity is shown at the top of each panel. It should be noted that the correlation coefficient is calculated from the velocity field, not from the horizontal divergence.

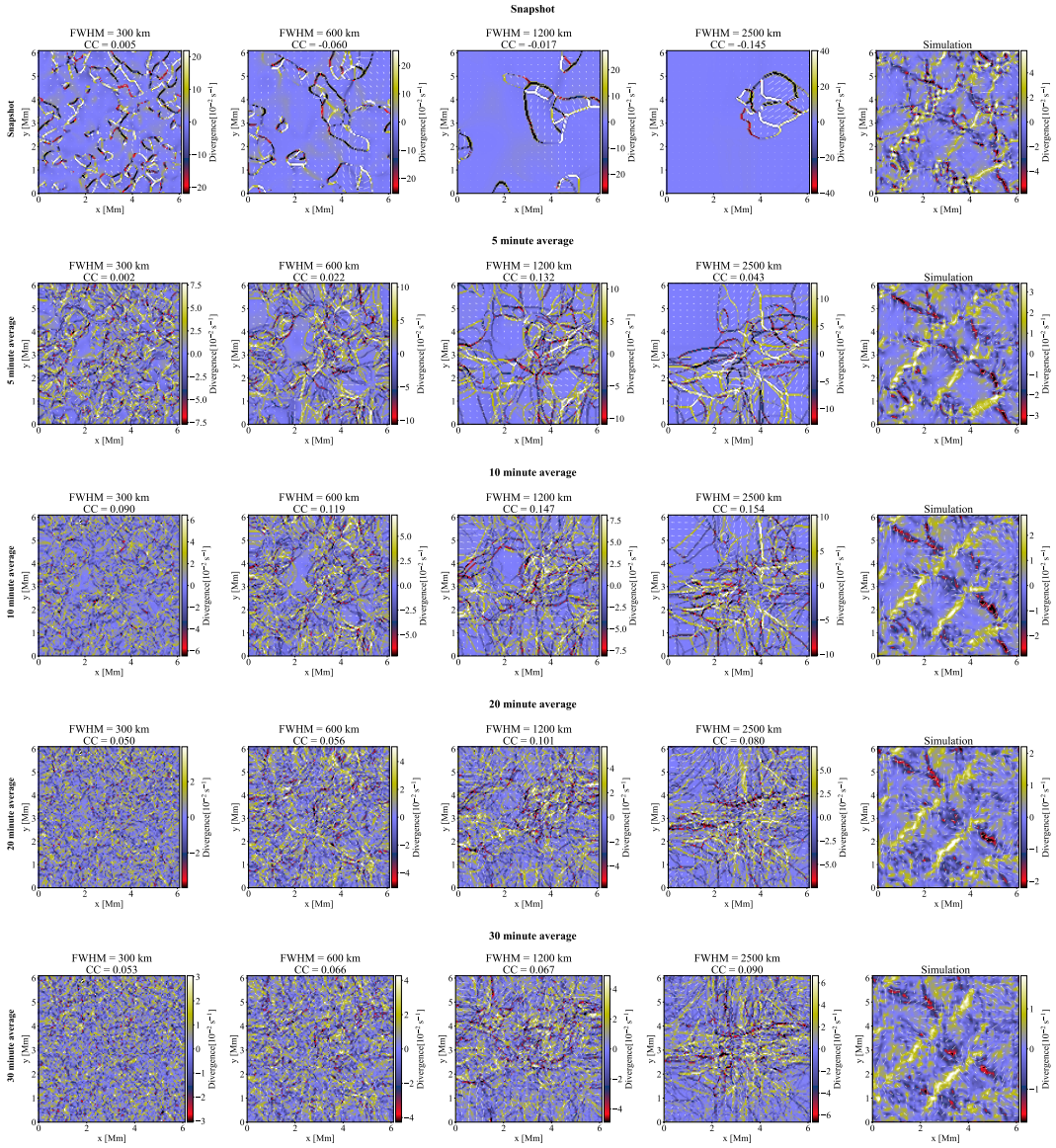
In the third row and third column of Figure 14, the highest CC value (0.192) was obtained when the interval was 30 seconds, the averaging time was 10 minutes, and the FWHM was 1200 km, so this parameter was adopted.



FigureC.1 These figures show the divergence of the horizontal velocity field obtained using LCT with a 30 s interval. Each row contains images averaged over the same time; each column represents images with the same FWHM. The rightmost column presents the time-averaged horizontal velocity field of the simulation. Above each image, the correlation coefficient (CC) with the corresponding simulated velocity is shown.



FigureC.2 These figures show the divergence of the horizontal velocity field obtained using LCT with a 60 s interval. Each row contains images averaged over the same time; each column represents images with the same FWHM. The rightmost column presents the time-averaged horizontal velocity field of the simulation. Above each image, the correlation coefficient (CC) with the corresponding simulated velocity is shown.



FigureC.3 These figures show the divergence of the horizontal velocity field obtained using LCT with a 120 s interval. Each row contains images averaged over the same time; each column represents images with the same FWHM. The rightmost column presents the time-averaged horizontal velocity field of the simulation. Above each image, the correlation coefficient (CC) with the corresponding simulated velocity is shown.

C.2 Helioseismology

We use the method of Gizon and Birch (2004) for calculating travel times. To detect waves passing through a specific depth, a filter is applied to the observational image using the sound speed at the target depth for a specific wavenumber and frequency.

$$F(\omega, k) = \exp\left[\frac{-\left(\omega/k - v_i\right)^2}{2\delta v_i^2}\right] \tag{C.7}$$

Where ω/k is the phase speed at the target depth and δv is determined as the allowable range for the target.

After applying the filter, the cross-correlation is calculated by shifting the oscillations between two points over time. Let $\phi(\boldsymbol{x}_i, t)$ be the intensity of the oscillation at time t at x_i, and the cross-correlation is computed as follows:

$$C(\boldsymbol{x}_1, \boldsymbol{x}_2, \Delta t) = \sum_i \phi(\boldsymbol{x}_1, t_i)\phi(\boldsymbol{x}_2, t_i + \Delta t) \tag{C.8}$$

The time when this cross-correlation is the highest is the simplest travel time. However, what is necessary for understanding the internal structure is not the travel time itself but the travel time shift, which is the difference from the typical travel time obtained from simulations or models.

Naturally, this cross-correlation contains observational noise. Therefore, a parameter ϵ, representing the amount of noise, is introduced to define a smooth cross-correlation. Let C^{ref} represent the cross-correlation in an ideal state obtained from simulations or models:

$$C^{\epsilon}(\boldsymbol{x}_1, \boldsymbol{x}_2, t) = \epsilon C(\boldsymbol{x}_1, \boldsymbol{x}_2, t) + (1 - \epsilon)\, C^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t) \tag{C.9}$$

When there is no noise, $\epsilon = 1$ and C^{ϵ} matches the observation. Conversely, when the noise is large, $\epsilon \ll 1$.

Using C^{ϵ}, the difference between the observed cross-correlation and the ideal cross-correlation C^{ref} is defined as follows:

$$X_{\pm}(\boldsymbol{x}_1, \boldsymbol{x}_2, t) = \int_{-\infty}^{\infty} dt' f(\pm t') \left[C^{\epsilon}(\boldsymbol{x}_1, \boldsymbol{x}_2, t') - C^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t' \pm \epsilon t)\right]^2 \tag{C.10}$$

Here, $f(\pm t')$ is a function used to separate the waves traveling from x_1 to x_2 and from x_2 to x_1, and a step function is used.

When $X_{\pm}(\boldsymbol{x}_1, \boldsymbol{x}_2, t)$ is minimized, the observation and model are considered most consistent, and the corresponding time $t = \tau_{\pm}$ is the travel time shift.

By calculating the derivative of $X_{\pm}(\boldsymbol{x}_1, \boldsymbol{x}_2, t)$ with respect to t:

$$\begin{aligned}\frac{\partial}{\partial t} X_{\pm}(\boldsymbol{x}_1,\boldsymbol{x}_2, t) = \int_{-\infty}^{\infty} dt'\, f(\pm t')\, \dot{C}^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t' \pm \epsilon t) \\ \times \left[\epsilon C(\boldsymbol{x}_1, \boldsymbol{x}_2, t') + (1-\epsilon)\, C^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t') - C^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t' \pm \epsilon t)\right]\end{aligned} \tag{C.11}$$

we compute the time $t = \tau$ at which the derivative is zero. When the noise is sufficiently large, i.e., $\epsilon \ll 1$, expanding ϵ to the first order yields the following expression for $\tau_{\pm}$:

$$\tau_{\pm} = \mp \frac{\int_{-\infty}^{\infty} dt' f(\pm t') \left[C(\boldsymbol{x}_1, \boldsymbol{x}_2, t') - C^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t')\right] \dot{C}^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t')}{\int_{-\infty}^{\infty} dt' f(\pm t') \left[\dot{C}^{\mathrm{ref}}(\boldsymbol{x}_1, \boldsymbol{x}_2, t')\right]^2} \tag{C.12}$$

Acknowledgments

I would like to express my sincere gratitude to Professor Hideyuki Hotta for his invaluable guidance and support throughout this research.

I am also deeply grateful to the National Astronomical Observatory, Associate Professor Yukio Katsukawa, and Dr. Ryotaro Ishikawa for their valuable advice.

I would like to express my deep gratitude to Dr. Yosuke Matsumoto, Dr. Masamune Oguri, Dr. Kotaro Kyutoku, Dr. Tomoyuki Hanawa, and Dr. Ryoji Matsumoto for their valuable advice during the laboratory seminar.

The numerical calculations in this study were performed using the XC50 system operated by CfCA at the National Astronomical Observatory of Japan.

Bibliography

Iñigo Arregui and Tom Van Doorsselaere. Coronal heating. In Srivastava, A. K. and Goossens, M. and Arregui, I., editor, *Magnetohydrodynamic Processes in Solar Plasmas*. Elsevier, 2024. doi: 10.48550/arXiv.2409.13318. Chapter 10, 1st Edition - May 31, 2024.

A. Asensio Ramos, I. S. Requerey, and N. Vitas. DeepVel: Deep learning for the estimation of horizontal velocities at the solar surface. *A&A*, 604:A11, July 2017. doi: 10.1051/0004-6361/201730783.

Luis Bellot Rubio and David Orozco Suárez. Quiet Sun magnetic fields: an observational view. *Living Reviews in Solar Physics*, 16(1):1, December 2019. doi: 10.1007/s41116-018-0017-1.

J. Christensen-Dalsgaard, W. Dappen, S. V. Ajukov, E. R. Anderson, H. M. Antia, S. Basu, V. A. Baturin, G. Berthomieu, B. Chaboyer, S. M. Chitre, A. N. Cox, P. Demarque, J. Donatowicz, W. A. Dziembowski, M. Gabriel, D. O. Gough, D. B. Guenther, J. A. Guzik, J. W. Harvey, F. Hill, G. Houdek, C. A. Iglesias, A. G. Kosovichev, J. W. Leibacher, P. Morel, C. R. Proffitt, J. Provost, J. Reiter, Jr. Rhodes, E. J., F. J. Rogers, I. W. Roxburgh, M. J. Thompson, and R. K. Ulrich. The Current State of Solar Modeling. *Science*, 272 (5266):1286–1292, May 1996. doi: 10.1126/science.272.5266.1286.

K. DeGrave, D. C. Braun, A. C. Birch, A. D. Crouch, and B. Javornik. Validating Forward Modeling and Inversions of Helioseismic Holography Measurements. *ApJ*, 863(1):34, August 2018. doi: 10.3847/1538-4357/aacffd.

Yuhong Fan, Nicholas Featherstone, and Fang Fang. Three-Dimensional MHD Simulations of Emerging Active Region Flux in a Turbulent Rotating Solar Convective Envelope: the Numerical Model and Initial Results. *arXiv e-prints*, art. arXiv:1305.6370, May 2013. doi: 10.48550/arXiv.1305.6370.

G. H. Fisher and B. T. Welsch. FLCT: A Fast, Efficient Method for Performing Local Correlation Tracking. In R. Howe, R. W. Komm, K. S. Balasubramaniam, and G. J. D. Petrie, editors, *Subsurface and Atmospheric Influences on Solar Activity*, volume 383 of

Astronomical Society of the Pacific Conference Series, page 373, January 2008.

Fukushima. Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol Cybern.*, 36(4):193–202, 1980. doi: 10.1007/BF00344251.

L. Gizon and A. C. Birch. Time-Distance Helioseismology: Noise Estimation. *ApJ*, 614(1): 472–489, October 2004. doi: 10.1086/423367.

Laurent Gizon, Robert H. Cameron, Majid Pourabdian, Zhi-Chao Liang, Damien Fournier, Aaron C. Birch, and Chris S. Hanson. Meridional flow in the Sun's convection zone is a single cell in each hemisphere. *Science*, 368(6498):1469–1472, June 2020. doi: 10.1126/science.aaz7119.

Deborah A. Haber, Bradley W. Hindman, Juri Toomre, Richard S. Bogart, Rasmus M. Larsen, and Frank Hill. Evolving Submerged Meridional Circulation Cells within the Upper Convection Zone Revealed by Ring-Diagram Analysis. *ApJ*, 570(2):855–864, May 2002. doi: 10.1086/339631.

H. Hamedivafa. Structure of proper motions in a sunspot penumbra. *Iranian Journal of Astronomy and Astrophysics*, 2(1):39–57, July 2015. ISSN 2251-8912. doi: 10.22128/IJAA.2015.16. URL `https://doi.org/10.22128/IJAA.2015.16`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR*, page 1, June 2016. doi: 10.1109/CVPR.2016.90.

H. Hotta and H. Iijima. On rising magnetic flux tube and formation of sunspots in a deep domain. *MNRAS*, 494(2):2523–2537, April 2020. doi: 10.1093/mnras/staa844.

H. Hotta and S. Toriumi. Formation of superstrong horizontal magnetic field in delta-type sunspot in radiation magnetohydrodynamic simulations. *MNRAS*, 498(2):2925–2935, August 2020. doi: 10.1093/mnras/staa2529.

H. Hotta, H. Iijima, and K. Kusano. Weak influence of near-surface layer on solar deep convection zone revealed by comprehensive simulation from base to surface. *Science Advances*, 5(1):2307, January 2019.

Stathis Ilonidis, Junwei Zhao, and Alexander Kosovichev. Detection of Emerging Sunspot Regions in the Solar Interior. *Science*, 333(6045):993, August 2011. doi: 10.1126/science.1206253.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv e-prints*, art. arXiv:1502.03167, February 2015. doi: 10.48550/arXiv.1502.03167.

Ryohtaroh T. Ishikawa, Motoki Nakata, Yukio Katsukawa, Youhei Masada, and Tino L.

Riethmüller. Multi-Scale Deep Learning for Estimating Horizontal Velocity Fields on the Solar Surface. *arXiv e-prints*, art. arXiv:2111.12518, November 2021.

Ryohtaroh T. Ishikawa, Motoki Nakata, Yukio Katsukawa, Youhei Masada, and Tino L. Riethmüller. Multi-scale deep learning for estimating horizontal velocity fields on the solar surface. *A&A*, 658:A142, February 2022. doi: 10.1051/0004-6361/202141743.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv e-prints*, art. arXiv:1412.6980, December 2014.

T. Kosugi, K. Matsuzaki, T. Sakao, T. Shimizu, Y. Sone, S. Tachikawa, T. Hashimoto, K. Minesugi, A. Ohnishi, T. Yamada, S. Tsuneta, H. Hara, K. Ichimoto, Y. Suematsu, M. Shimojo, T. Watanabe, S. Shimada, J. M. Davis, L. D. Hill, J. K. Owens, A. M. Title, J. L. Culhane, L. K. Harra, G. A. Doschek, and L. Golub. The Hinode (Solar-B) Mission: An Overview. *Sol. Phys.*, 243(1):3–17, June 2007. doi: 10.1007/s11207-007-9014-6.

Timothy P. Larson and Jesper Schou. Global-Mode Analysis of Full-Disk Data from the Michelson Doppler Imager and the Helioseismic and Magnetic Imager. *Sol. Phys.*, 293(2): 29, January 2018. doi: 10.1007/s11207-017-1201-5.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, May 2015. doi: 10.1038/nature14539.

J. M. Malherbe, T. Roudier, R. Stein, and Z. Frank. Dynamics of Trees of Fragmenting Granules in the Quiet Sun: Hinode/SOT Observations Compared to Numerical Simulation. *Sol. Phys.*, 293(1):4, January 2018. doi: 10.1007/s11207-017-1225-x.

Hiroyuki Masaki and Hideyuki Hotta. Detection of solar internal flows with numerical simulation and machine learning. *PASJ*, 76(6):L33–L38, December 2024. doi: 10.1093/pasj/psae093.

Hiroyuki Masaki, Hideyuki Hotta, Yukio Katsukawa, and Ryohtaroh T. Ishikawa. Solar horizontal flow evaluation using neural network and numerical simulations with snapshot data. *PASJ*, 75(6):1168–1182, December 2023. doi: 10.1093/pasj/psad063.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814. Citeseer, 2010.

Laurence J. November and George W. Simon. Precise Proper-Motion Measurement of Solar Granulation. *ApJ*, 333:427, October 1988. doi: 10.1086/166758.

W. Dean Pesnell, B. J. Thompson, and P. C. Chamberlin. The Solar Dynamics Observatory (SDO). *Sol. Phys.*, 275(1-2):3–15, January 2012. doi: 10.1007/s11207-011-9841-3.

M. Rempel and M. C. M. Cheung. Numerical Simulations of Active Region Scale Flux Emergence: From Spot Formation to Decay. *ApJ*, 785(2):90, April 2014. doi: 10.1088/0004-637X/785/2/90.

François Rincon and Michel Rieutord. The Sun's supergranulation. *Living Reviews in Solar Physics*, 15(1):6, December 2018. doi: 10.1007/s41116-018-0013-5.

Forrest J. Rogers, Fritz J. Swenson, and Carlos A. Iglesias. OPAL Equation-of-State Tables for Astrophysical Applications. *ApJ*, 456:902, January 1996. doi: 10.1086/176705.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv e-prints*, art. arXiv:1505.04597, May 2015.

Rosenblatt. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

Rumelhart. Hinton GE and Williams RJ: Learning internal representations by error propagation. *Parallel Distributed Processing*, 1:318–362, 1986.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. doi: 10.1038/323533a0.

Takashi Sakurai, Masaoki Kojima, Kentaro Kosugi, and Kazunari Shibata, editors. **太陽** *[The Sun]*, volume 10 of **現代の天文学** *[Modern Astronomy]*. Nihon Hyoronsha, Tokyo, Japan, 2nd edition, dec 2018. ISBN 978-4-535-60760-6. URL `https://www.nippyo.co.jp/shop/book/7898.html`.

M. J. Seaton, Y. Yan, D. Mihalas, and A. K. Pradhan. Opacities for stellar envelopes. *MNRAS*, 266:805, February 1994. doi: 10.1093/mnras/266.4.805.

A. K. Singh, Devendraa Siingh, and R. P. Singh. Space Weather: Physics, Effects and Predictability. *Surveys in Geophysics*, 31(6):581–638, December 2010. doi: 10.1007/s10712-010-9103-1.

H. C. Spruit, A. Nordlund, and A. M. Title. Solar convection. *ARA&A*, 28:263–301, January 1990. doi: 10.1146/annurev.aa.28.090190.001403.

R. F. Stein and Å. Nordlund. Simulations of Solar Granulation. I. General Properties. *ApJ*, 499(2):914–933, May 1998. doi: 10.1086/305678.

Robert F. Stein. Solar Surface Magneto-Convection. *Living Reviews in Solar Physics*, 9(1):4, December 2012. doi: 10.12942/lrsp-2012-4.

Michael Stix. *The sun: an introduction*. Springer, 2002.

K. Takahata, H. Hotta, Y. Iida, and T. Oba. Relationship between magnetic field properties and statistical flow using numerical simulation and magnetic feature tracking on solar photosphere. *MNRAS*, 503(3):3610–3616, May 2021. doi: 10.1093/mnras/stab710.

Benoit Tremblay and Raphaël Attie. Inferring Plasma Flows at Granular and Supergranular Scales with a New Architecture for the DeepVel Neural Network. *Frontiers in Astronomy and Space Sciences*, 7:25, June 2020. doi: 10.3389/fspas.2020.00025.

Benoit Tremblay, Thierry Roudier, Michel Rieutord, and Alain Vincent. Reconstruction of Horizontal Plasma Motions at the Photosphere from Intensitygrams: A Comparison Between DeepVel, LCT, FLCT, and CST. *Sol. Phys.*, 293(4):57, April 2018. doi: 10.1007/s11207-018-1276-7.

Saku Tsuneta, Y. Suematsu, K. Ichimoto, T. Shimizu, Y. Katsukawa, S. Nagata, D. Orozco Suárez, B. Lites, D. Shine, T. Tarbell, and A. Title. Magnetic Landscape Of Solar Polar Region With Solar Optical Telescope Aboard Hinode. In *American Astronomical Society Meeting Abstracts #210*, volume 210 of *American Astronomical Society Meeting Abstracts*, page 94.05, May 2007.

M. Verma, M. Steffen, and C. Denker. Evaluating local correlation tracking using CO5BOLD simulations of solar granulation. *A&A*, 555:A136, July 2013. doi: 10.1051/0004-6361/201321628.

A. Vögler, S. Shelyag, M. Schüssler, F. Cattaneo, T. Emonet, and T. Linde. Simulations of magneto-convection in the solar photosphere. Equations, methods, and results of the MURaM code. *A&A*, 429:335–351, January 2005. doi: 10.1051/0004-6361:20041507.

David F. Webb and Timothy A. Howard. Coronal Mass Ejections: Observations. *Living Reviews in Solar Physics*, 9(1):3, December 2012. doi: 10.12942/lrsp-2012-3.

B. T. Welsch, G. H. Fisher, W. P. Abbett, and S. Regnier. ILCT: Recovering Photospheric Velocities from Magnetograms by Combining the Induction Equation with Local Correlation Tracking. *ApJ*, 610(2):1148–1156, August 2004. doi: 10.1086/421767.

K. L. Yeo, A. Feller, S. K. Solanki, S. Couvidat, S. Danilovic, and N. A. Krivova. Point spread function of SDO/HMI and the effects of stray light correction on the apparent properties of solar surface phenomena. *A&A*, 561:A22, January 2014. doi: 10.1051/0004-6361/201322502.

Junwei Zhao and Alexander G. Kosovichev. Helioseismic Observation of the Structure and Dynamics of a Rotating Sunspot Beneath the Solar Surface. *ApJ*, 591(1):446–453, July 2003. doi: 10.1086/375343.