REINFORCEMENT LEARNING BASED RESOLUTION IMPORVEMENT OF GEOPHYSICAL DATA

Thesis Report
Submission

MASTER OF SCIENCE & TECHNOLOGY

In

Applied Geophysics

By

Himanshu Jain

Admission no. 20MC0039

To

Swarandeep Sahoo

Assistant Professor

Department of Applied Geophysics

INDIAN INSTITUTE OF TECHNOLOGY

(INDIAN SCHOOL OF MINES) DHANBAD, 826004

15 Dec, 2022

भारतीय प्रौद्योगिकी संस्थान (भारतीय खनि विद्यापीठ), धनबाद
धनबाद, झारखण्ड, भारत, पिन – 826004
INDIAN INSTITUTE OF TECHNOLOGY
(INDIAN SCHOOL OF MINES), DHANBAD
DHANBAD, JHARKHAND, INDIA, PIN - 826004
(An Autonomous Institute under Ministry of HRD, Govt. of India)

Certificate

This is to certify that **Himanshu Jain**, Admission No **20MC0039**, has completed his dissertation work entitled "**Reinforcement learning based resolution improvement of geophysical data**" in partial fulfillment of the requirement for the award of Master of Science and Technology in Applied Geophysics under the guidance of **Prof. Swarandeep Sahoo** in Indian Institute of Technology (Indian School of Mines), Dhanbad. This work has not been submitted elsewhere for any other degree or distinction.

FORWARDED BY:	**Swarandeep Sahoo**
Head of the Department	Assistant Professor
Department of Applied Geophysics	Department of Applied Geophysics
Indian Institute of Technology	Indian Institute of Technology
(Indian School of Mines), Dhanbad	(Indian School of Mines), Dhanbad

Declaration

The Dissertation titled "**Reinforcement learning based resolution improvement of geophysical data"** is a presentation of my original research work and is not copied or reproduced or imitated from any other person's published or unpublished work. Wherever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions, as may be applicable. Every effort is made to give proper citation to the published/unpublished work of others, if it is referred to in the Dissertation.

To eliminate the scope of academic misconduct and plagiarism, I declare that I have read and understood the UGC (Promotion of Academic Integrity and Prevention of Plagiarism in Higher Educational Institutions) Regulations, 2018. These Regulations have been notified in the Official Gazette of India on 31st July, 2018.

I confirm that this Dissertation has been checked with the online plagiarism detector tool Turnitin (http:///www.turnitin.com) provided by IIT (ISM) Dhanbad and a copy of the summary report/report, showing Similarities in content and its potential source (if any), generated online through Turnitin is enclosed at the end of the Dissertation. I hereby declare that the Dissertation shows less than equal to 10% similarity as per the report generated by Turnitin and meets the standards as per MHRD/UGC Regulations and rules of the Institute regarding plagiarism. I further state that no part of the Dissertation and its data will be publishedwithout the consent of my guide. I also confirm that this Dissertation work, carried out under the guidance of **Prof. Swarandeep Sahoo**, Assistant Professor, Department of Applied Geophysics, has not been previously submitted for assessment for the purpose of the award ofa degree either at IIT (ISM) Dhanbad or elsewhere to the best of my knowledge and belief.

हिमांशु जैन

(Forwarded by)

Prof. Swarandeep Sahoo

3-Year M.Sc. Tech, Applied Geophysics

Department of Applied Geophysics

20MC0039

IIT (ISM), Dhanbad

Certificate for Classified Data

This is to certify that the thesis entitled "**Reinforcement learning based resolution improvement of geophysical data**" being submitted to the Indian Institute of Technology (Indian Schoolof Mines), Dhanbad by **Himanshu Jain**, Admission No. 20MC0039 for the award of 3-year M.Sc. Tech. in Applied Geophysics does not contain any classified information. This work is original and has not yet been submitted to any institution or university for the award of any degree.

हिमांशु जैन

HIMANSHU JAIN

M.Sc. Tech (Applied Geophysics)

Department of Applied Geophysics

20MC0039

Acknowledgement

I would like to sincerely thank **Prof. Swarandeep Sahoo**, Department of Applied Geophysics, Indian Institute of Technology (Indian School of Mines), Dhanbad, for his diligent guidelines, essential guidance, and constant encouragement of this work, as well as for supplying me properinstruments and guidance that helped me with completion of my dissertation satisfactorily.

I would also want to specially thank **Prof. S.K Pal**, Head, Department of Applied Geophysics, Indian Institute of Technology (Indian School of Mines), Dhanbad, for permittingme to use the resources there to finish my dissertation.

It is my outmost pleasure to express sincere gratitude to my friend **Shraddha P Jain** and my **Family**, without whom this work would not have been possible.

I would want to use this opportunity to thank **Mr. Priyabrata Mukherjee**, **Mr. Abhay Shukla**,**Mr.Dheeraj Sharma and Mr. Om Prakash Gupta** for their unwavering support and encouragement throughout my dissertation research.

I am grateful for having wonderful and supportive parents. Their support and companionship are always pushed me to push hard regardless any situation.

हिमांशु जैन

HIMANSHU JAIN

20MC0039

M.Sc. Tech (Applied Geophysics)

Department of Applied Geophysics

ABSTRACT

Majority of magnetic field of the Earth originate from outer core region. Models are required to observe the variation and study the origin of magnetic field all over the globe. Also, magnetic field changes abruptly through time and location on surface. Prior to advancement of satellite measurement, we have limited observed data. Our aim to develop a model which has ability to predict field value in the past and also to have a better resolution. In order to develop model, evolution is required to have more accuracy. But due to lack of data it is hard to made accurate model for field data prediction.

Hence our approach is to use the Reinforce learning along with physics behind its nature of variation. We require some paleomagnetic criteria to further evaluate our model. In this work we have tried to find an approach to use reinforcement learning technique which might be helpful to design a model that can predict earth's magnetic field in past events with high resolution. Also, we have tried to analyze the paleomagnetic criteria and measure the fit level with the current globally accepted model IGRF-13.

List of Figures

List of Tables

Table of Contents

CHAPTER-I

INTRODUCTION:

The Earth acts as large spherical magnet, and its field changes with time and location. Its magnetic field is so called geomagnetic field. The measured geomagnetic field on the surface of Earth is a combination of magnetic field generated by many sources, these fields are superimposed with each other. Field sources are generated from inside and outside of the Earth surface but the majority of field sources are generated from the inside of the surface i.e., field generated in the outer core and crustal field. External sources are caused by solar wind. Solar wind interacts with the Earth's atmosphere especially in magnetopause and produces magnetic field.

The field associated with outer core act as the main field of the Earth's magnetism.

Measurement of Geomagnetic data and Collection:

The Magnetic field of the Earth is a vector quantity i.e., it has both magnitude and direction. The magnitude or field intensity is measured in the units of *tesla (T) or nanotesla (nT).* The Magnetic field of the Earth varies all over the surface of the Earth. The field intensity ranges from *25000nT to 65000nT*, where it reaches maximum at the magnetic poles. The instruments which take measurement are known as magnetometers.

Measurements of Magnetic field of the Earth are taken with various ground observatories situated all over the globe, low-Earth orbiting satellite and paleomagnetic data. Ground observatories are continuously measuring the field and rapid change of magnetic field helps to monitor temporal variation from hour to decades or century.

Exploration of Magnetic field of the Earth began with the Sputnic3 satellite and further globally data are collected by various satellite such as POGO, MAGSAT, Ørsted and, CHAMP etc. In present time, a low-orbit satellite mission – SWARM constellation is using to study the dynamics of Magnetic field of the Earth.

It consists of three identical satellite named as *Alpha, Bravo and Charlie.* Each of the three Swarm satellites measures high resolution data with direction and variation of the magnetic field.

Geomagnetic Variation:

As we have discussed earlier, intensity of geomagnetic field changes through time termed as temporal variation. Different magnetic field source causes different temporal variation like outer core fields changes with time very slowly, while crustal field remains constant over geological scales. These variations are classified further as

1.3.1 Diurnal Variation:

Variation of field observe over the period of day time is termed as diurnal variation (DV).

It is associated with the field changes responsible by the external sources causes by solar wind.

1.3.2 Secular Variation:

Variation of field observe over prolonged period such as timescales of years, decade or centuries termed as secular variation (SV). It is associate with the Earth's internal field. Secular Variation is not constant with time and it varies from place to place.

Geomagnetic elements:

Magnetic field is a vector field, so 3 components are necessary to represent the field. The field **(F)** elements are called geomagnetic elements. It can be expressed as Cartesian coordinates as well as Spherical polar coordinates. I and

In Cartesian coordinate system (X, Y, Z),

X = component parallel to geographic north direction

Y = component parallel to east direction

Z = component in Vertical downward direction

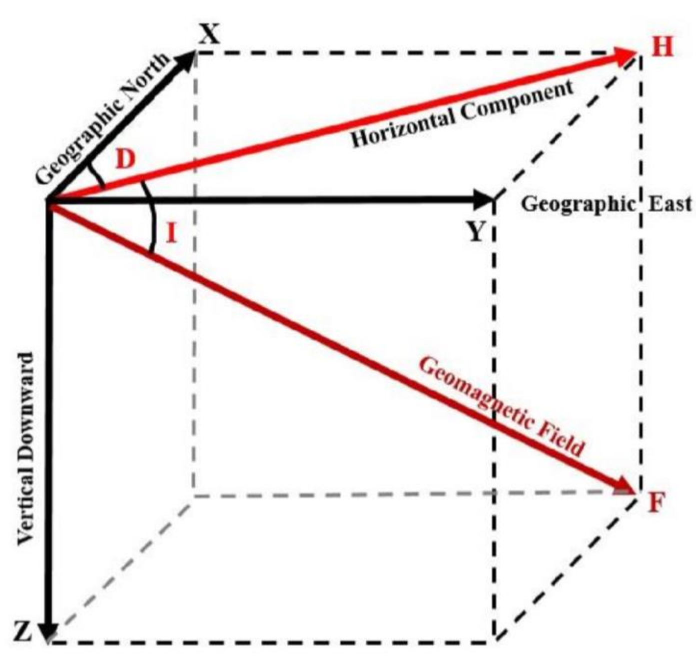


Fig. 1-1: Geomagnetic Elements represented into Cartesian coordinate.

These elements are generally expressed in terms of nanotesla ($10^{-9} Tesla$ / $10^{-5} Gauss$). Principal equations relating different elements are:

$$F = (X^2 + Y^2 + Z^2)^{1/2}$$

$$F = (H^2 + Z^2)^{1/2}$$

$$H = F * \cos(I)$$

$$Z = F * \sin(I)$$

$$X = H * \cos(D)$$

$$Y = H * \sin(D)$$

Here, I referred as Inclination, D referred as Declination and H referred as Horizontal component of magnetic field.

CHAPTER-II

IGRF-13TH GENERATION:

The International Geomagnetic Reference Field is group of spherical harmonics coefficient IGRF is used as input in model to find Earth's internal magnetic field between 1900 AD to present. Currently we are using 13th generation of IGRF model which is derived from the observation of satellites, base stations situated globally on surface of the Earth etc.

IGRF model is used to investigate the internal origin of magnetic field of the Earth, local magnetic anomalies in the lithosphere, space weather etc.

Magnetic field associated with internal origin i.e., outer core changes unpredictably with time-scale varies some months to Millions of months. That is why IGRF must regularly revised in every five years. The year in which coefficient are determined are known as epoch. IGRF models are non-definitive in nature that means it must revise in future.

2.1.1 Mathematical Analysis of IGRF model:

Majority of main field is related to field generated in the Earth's core. Let us assume that it is associated by current free region, i.e., the current density J = 0. Therefore, curl of the vector magnetic field B is zero.

$$\mu oJ = \nabla \times B = 0 \qquad (2.1)$$

Here B is referred as magnetic field of Earth. It can be further expressed as gradient of a scalar potential V,

$$B = -\nabla V \qquad (2.2)\ and \quad \nabla \cdot B = 0 \qquad (2.3)$$

Where '– 'sign is matter of convention. The scalar potential V must be the solution of Laplace's equation, therefore,

$$\nabla \cdot (\nabla V) = \nabla^2 V = 0 \qquad (2.4)$$

Above equation is called the Laplace's equation. It is practically easy to study the Earth in spherical coordinates rather than in Cartesian coordinate.

The solution of Laplace's equation in spherical co-ordinate will be

$$\nabla^2 V = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial V}{\partial r}\right) + \frac{1}{r^2 sin^2\theta}\frac{\partial^2 V}{\partial \lambda^2} + \frac{1}{r^2 sin\theta}\frac{\partial}{\partial \theta}\left(sin\theta\frac{\partial V}{\partial \theta}\right) = 0 \qquad (2.5)$$

Here, r is distance from the center, θ and λ are co-latitude and longitude respectively. But the Magnetic field of the Earth is rotationally non symmetric, so solution of Laplace's equation becomes,

$$V = \sum_{n=0}^{\infty}\left(A_n r^n + \frac{B_n}{r^{n+1}}\right)\sum_{m=0}^{n}(a_n^m cosm\varphi + b_n^m sinn\varphi)\, P_n^m(cos\theta) \qquad (2.6)$$

Where, $P_n^m(cos\theta)$ is a Legendre Polynomial associated with it having degree n and order m.

Above equation can be written as

$$V = \sum_{n=0}^{\infty}(A_n r^n + \frac{B_n}{r^{n+1}})\sum_{m=0}^{n} Y_n^m(\theta,\varphi) \qquad (2.7)$$

The function $Y_n^m(\theta,\varphi)$ is known as *Spherical Harmonics Function*. It describes the variation of potential V on spherical surface.

$$Y_n^m(\theta,\varphi) = (-1)^m\sqrt{\frac{(2n+1)}{4\pi}\frac{(n-m)!}{(n+m)!}}P_n^m(cos\theta)e^{im\varphi},\ for\ \begin{cases} n = 0,1,2,3,\dots\dots\dots\dots\dots\dots.., \\ m = -n,-n+1,\dots,n-1,l \end{cases} \qquad (2.8)$$

Here, P_n^m are Legendre function of Schmidt where m and n are order and degree respectively. Therefore, two solutions from radially dependency are possible, 1) V^{int} – describing internal sources and, 2) V^{ext} – describing external sources.

$$V^{int}(r,\theta,\lambda,t) = R\sum_{n=1}^{N_{int}}\sum_{m=0}^{n}[g_n^m(t)cosm\lambda + h_n^m(t)sinm\lambda]\left(\frac{a}{r}\right)^{n+1} P_n^m(cos\theta) \qquad (2.9)$$

Here, R = 6371 km is radius in general we take, (r, θ, λ) are the coordinate geographically, and

$[g_n^m, h_n^m]$ Gauss coefficients. Parameter N^{int} is maximum spherical harmonic degree. Initially it was chosen to be 10 up to epoch 1995 and increased to 13 from epoch 2000. The Gauss Coefficients $[g_n^m, h_n^m]$ changes through time.

2.1.2 Final Model Selection:

The IGRF model is produced and maintained under the IAGA. Association of Geomagnetism and Agronomy is associated with analysis of the Earth's electrical and magnetic character. This group invites different institution, research labs etc. to submit candidate model for next generation model. For selection of 13th generation, median of the coefficients of gauss for all the candidate models are accepted as final model.

Geocentric Axial Dipole (GAD) Moment:

More than 90% of the Magnetic field of the Earth is only from dipole field of earth, which is mainly at earth center and align with its rotation axis, which is known as geocentric axial magnetic dipole. Dipole field is associated by Gauss coefficients (n=1), g_1^0, g_1^1 and h_1^1. Here g_1^0 is the strongest component of the field and it determines the axial dipole moment.

$$m_{ax} = \frac{4\pi}{\mu_0}R^3 g_1^0 \qquad (2.10)$$

The next strongest term is g_1^1 andh_1^1, which describe the contribution from additional dipoles having axes in equatorial plane. The total dipole moment of the Earth is computed by vector sum of all three components.

$$m_T = \frac{4\pi}{\mu_0}R^3\sqrt{(g_1^0)^2 + (g_1^1)^2 + (h_1^1)^2} \qquad (2.11)$$

All three-dipole component describe geocentric dipole. In present time it is inclined at 11.2○ ̊to the Earth's rotation axis. By subtracting field associated with geocentric dipole from total field, is called non-dipole field. It is obtained by using all Gauss coefficient of degree $n \geq 2$.

2.2.1 Geomagnetic Poles:

Points on the surface of the Earth, where axis of geocentric dipole cuts are called north and south geomagnetic poles.

2.2.2 Magnetic Poles:

Points on the earth's surface, where inclination angle from normal to surface are called south or north magnetic pole.

CHAPTER-III

LITERATURE REVIEW:

C.J.Sprain et al, (2019), observed the change in paleomagnetic records and presented a set of five criteria which was termed as Quality of Paleomagnetic Modeling criteria Q_{PM}. In the paper they proposed the quantify degree of agreement of geodynamo simulation with spatial as well as temporal variation of long term paleomagnetic field. They have assessed 46 simulations with $\mathbf{Q}_{PM}$ criteria but no simulation has fully satisfied it. The paper is focused to reproduce computational parameter which may help to build Earth-like simulation.

3.1.1 Quality of Paleomagnetic Modeling Criteria (Q_{PM}):

Q_{PM} criteria helps to quantify Earth's paleomagnetic behavior. There is total five criteria in Q_{PM}, each criterion has score 1 and maximum score of five.

3.1.1.1 Inclination Anomaly:

Anomaly of inclination is calculated by the difference in between predicted (I_{obs}) and approximated calculated by geocentric axial dipole (I_{GAD}).

$$\Delta I = I_{obs} - I_{GAD} \qquad (4.1)$$

It was seen that I_{GAD} shows latitudinal change as following given axial dipole equation

$$\tan I_{GAD} = 2tan\lambda \qquad (4.2)$$

3.1.1.2 Virtual Geomagnetic Pole Angular Dispersion:

VGP angular dispersion (S) helps to quantify variation of paleosecular in time scale of field of paleomagnetism.

$$S = \left[\frac{1}{n-1}\sum_{i=1}^{n}\Delta_i^2\right]^{\frac{1}{2}} \qquad (4.3)$$

Δ_i is the distance from the pole to *i*th VGP.

3.1.1.3 VGP Dispersion Variation along Latitude:

Virtual Geomagnetic Pole dispersion also varies with latitude such as

$$S^2 = a^2 + (b\lambda)^2 \qquad (4.4)$$

Here, a and b are variation in the symmetric and antisymmetric spherical harmonic decomposition along equator of the field.

3.1.1.4 Virtual Dipole Moment:

Like the inclination, magnetic field intensity also depends on latitude. Virtual Dipole Moment is the strength of geocentric dipole that produces observed field intensity at given paleolatitude.

$$VDM = \frac{4\pi R^3 F}{\mu_0}(3\cos^2\theta_m + 1)^{-\frac{1}{2}} \qquad (4.5)$$

Where R, θ_m and μ_0 are radius of the Earth's surface, magnetic colatitude and permeability of free space respectively.

Variability of distribution of virtual dipole moment (VDMvar) estimates the temporal variation in magnetic intensity.

$$\%V = \frac{VDM_{max} - VDM_{min}}{VDM_{med}} \qquad (4.6)$$

3.1.1.5 Dipole Field Reversal:

The reversal criteria fulfils if model reverses in an Earth-like manner.

CHAPTER-IV

ARTIFICIAL INTELLIGENCE:

The difference between human and other species is the ability to learn, think, understand, and make it apply in livelihood. It is called intelligence. Today we are in the age of technology and automation. We require machinery to ease our work and but machine do not think as we human think. Getting machine to think and work as human do is called Artificial Intelligence. So, by the definition – *the science of getting machines to mimic the human's behavior.*

Machine learning is designing a system and making it learn and make prediction by experience with data. There are three types of machine learning such as Supervised, Unsupervised and Reinforcement learning. Main difference in between supervised and unsupervised learning is labelling of dataset which is used to train machine. In supervised learning, data is labelled but in unsupervised learning input data is unlabeled. Most of prediction and classification problem are solved by help of these two learning.

Another type of machine learning is Reinforcement Learning. Reinforce learning is all about making sequence of decision based on its previous action. Reinforcement learning algorithm interact with two elements – *Agent* and *Environment.* It includes different *state.* A learning agent acts in particular state, if it makes correct decision, it gets rewards point and if decision is wrong, it gets penalty. Based on all the rewards point collected in each state, optimum decision has been decided by algorithm.

REINFORECEMENT LEARNING

4.2.1 INTRODUCTION:

Reinforcement learning is another type of Machine learning in which the goal is achieved by reward and penalty basis. In this learning objective is to achieve high reward points. It is different from supervised learning. Since supervised learning knows the outcomes but in reinforcement learning model will give result based on reward by performing some tasks in the given situation. We can also understand that it is a feedback-based technique in which models will be learned by the feedback received by the environment.

In reinforcement learning, models must perform some action based on their situation. The model has no idea about in which way the action must be taken but it is known which action will result in the most award by trying them.

4.2.2 Terms Used in Reinforcement Learning:

A few terminologies that are used in reinforcement learning are-

Agent: It is a model or learner who performs some decision-making action which may receive reward.

Environment: It is a situation in which the agent is present and it will interact with it.

Action (A): Action are some moves taken by agent in environment.

State (S): State is a situation observed from the environment after an action performed by the agent.

Reward (R): A feedback after every action counter with environment by agent.

Policy (π): It is a function which decides the next action based on the current state.

Now a days using machine learning and Artificial intelligence tools is exceedingly popular in various fields. It is used in the field of geoscience such as lithology identification, earthquake prediction, fault detection, remote sensing application and making various models which can solve geophysical problems. In the geophysics community Deep Learning (a type of artificial intelligence technique) is immensely popular. The benefit of deep learning is that it has potential to predict complex systems accurately provided having a much big data. If data is more enough than deep learning is the best tool we have.

Our aim is to predict the intensity of earth's magnetic field of past events. We have chosen a mathematical model IGRF13 to get the prediction. The IGRF model uses some time dependent constants which are known as gauss coefficients. Changes are required in gauss coefficients in a 5-year interval because the Earth's magnetic field changes through time.

As we know, to predict earth's magnetic field in past events it is required to have known gauss coefficient of that time. So, the task is to make an AI model that can predict gauss coefficient at any time span. For construction of such a model, deep learning model is a very good idea to proceed with but it requires much big data. The problem of construction geomagnetic field model is that we have limited data set of gauss coefficient say from 2000 to 2020 in the interval of 5 year. To deal with this kind of problem there is a machine learning technique called reinforcement learning.

In reinforcement learning technique, model will not just predict gauss coefficient but it will decide whether it should choose coefficient so that it can be our desired model. I have tried to approach a few of the algorithms which can be helpful to construction of such model.

4.2.3 Geomagnetic Field Model in View of Reinforcement Learning:

It is incredibly difficult to get a model that can predict gauss coefficients accurately and precisely. To deal with such problems there is a way to perform reinforcement learning. Here in reinforcement learning our mathematical model should choose some specific gauss coefficient to satisfy paleomagnetic criteria which have been mentioned above. Our approach will be to assume mathematical model as agent and paleomagnetic criteria will be its environment. It will have a different combination of predictable gauss coefficients, considered as state. The freedom of choosing and denying will be its action and after each action, either reward or penalty will be given. Our goal is to decide the gauss coefficient based on maximum reward received. Maximum reward points means that the associated gauss coefficient will satisfy close to the all paleomagnetic criteria.

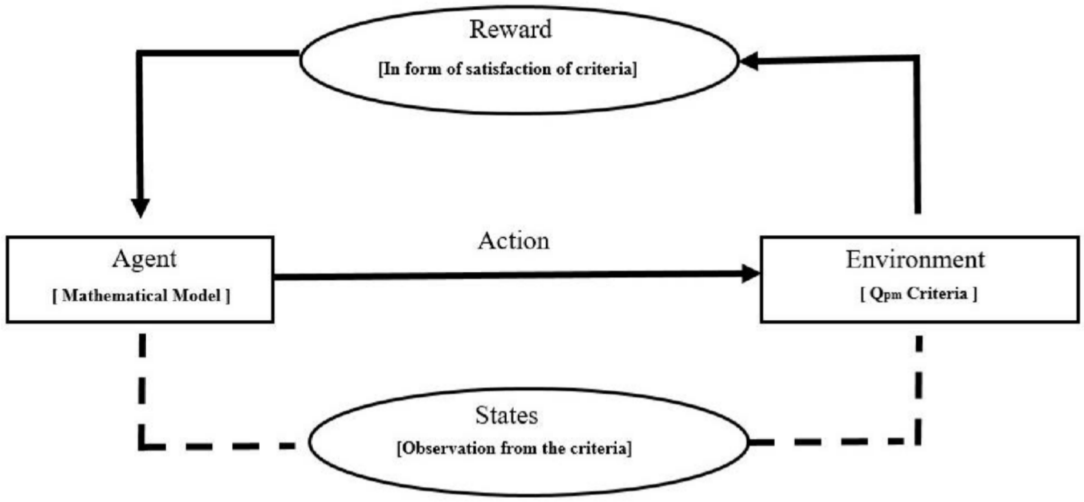


Fig. 4-1: The Agent-Environment interaction in Reinforcement learning.

4.2.4 Markov Decision Process:

It is a mathematical framework where it can be used for model the decision-making problems. In MDP problem the outcome are partial controllable and partial random. In MDP the agent constantly interacts with environment and perform actions. After each action a new state is generated by the environment.

The Markov process is formalized sequence of the action of agent such as –

i. The agent receives observation of the environment state,

$$S_t \in S$$

ii. The agent takes action,

$$A_t \in A$$

iii. Each combination of actions in each creates the state-action pair (S_t, A_t)

iv. From this step S_t, agent will increment to the next step

$$S_{t+1} \in S$$

v. After reaching next time step, the agent receives calculated reward for previous decision made,

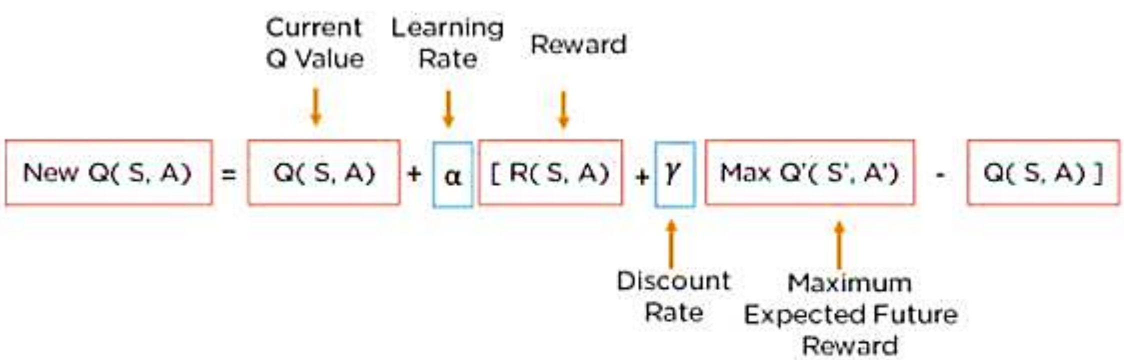
$$R_{t+1} \in R$$

4.2.5 Q – Learning:

In Q-learning 'Q' stands for quality. Quality in this algorithm represent how useful a given action is to get the highest reward. Q-learning is an off-policy reinforcement learning algorithm that triers to get the best action in each state. This is off policy because it chooses actions randomly. So, the policy does no required. Q- Learning is easy to understand and implement. Q-learning seeks a policy which gives the maximum possible total reward for each state. It takes only two values as an input which is state and action.

When it comes to apply Q-learning algorithm first thing is to create a Q-table. Q-table is a matrix of shape [states, actions]. In the start the values will be zero. Then we update and store values after each episode (when an agent reach to the goal or can't take new actions). The Bellman Equation is used to determine the value of particular state. The higher the value state will be the optimum state. The equation will be given as:

The discount factor (γ) in Bellman Equation determines how much the agent cares about the distant future rewards relative to immediate future rewards. If$\gamma = 0$, the agent will be completely learning about actions that produce an immediate reward. If $\gamma = 1$ the agent evaluates each of its action based on the sum of total of its future rewards. The learning rate α helps us to determine how much information of the previously computed Q-value for a given state action pair we retain over the newly computed Q-value calculated for the same state-action pair at a later time step. The higher the learning rate, the more likely the agent will adopt the newly



computed Q-value. Therefore, we need to take care of the tradeoff between new and old Q-value using the appropriate learning rate.

CHAPTER-V

METHODOLOGY:

To construct our synthetic model, some important things are needed, such as data, type of learning algorithm and few criteria to evaluate our model. Since it will require the use of Reinforce learning and our goal is to find the best suited gauss coefficients, and our model must satisfy paleomagnetic criteria. I have used python programming language for computational and data analysis purposes. Steps for computing paleomagnetic behavior are summarized below.

Using 13th generation IGRF model we get Inclination (I) (degrees), Declination (D) (degrees), Horizontal intensity of magnetic field (H) (nT), Total intensity of magnetic field (F) (nT), North component of magnetic field (X) (nT), East component of magnetic field (Y) (nT), Vertical component of magnetic field (Z) (nT), Secular variation of X (nT/yr), Secular variation of Y (nT/yr), Secular variation of Z (nT/yr), Secular variation of I (min/yr), Secular variation of D (min/yr), Secular variation of H (nT/yr), Secular variation of F (nT/yr). Out of this, we have some more constraints to choose what best fits our model. Constraints can be derived or can be used directly. Like inclination anomaly derived from inclination and declination.

Flow chart of basic working of the environment:

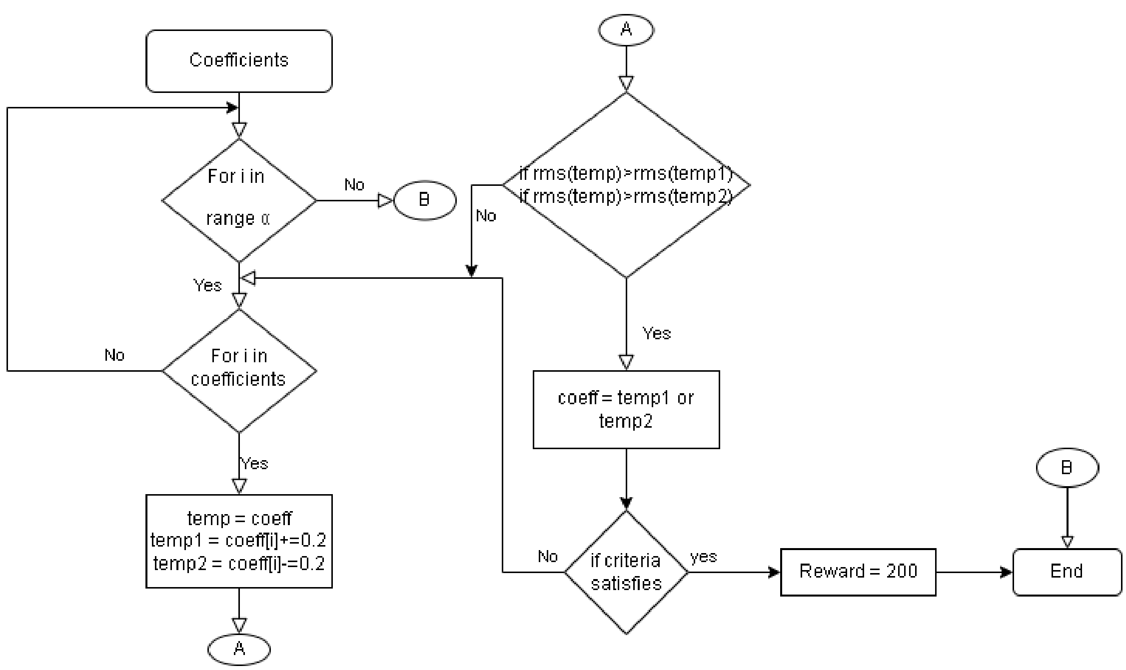


Fig. 5-1: Flow chart of the working of environment

Criteria for environmental constraints:

The environment of the model consists of some basic set of criteria that the agents should follow. But it is important to choose the best suitable criterion for our model so that model should converge and give the optimum result. It should use all the coefficients (available and predicted) to get the value. The constraint should follow in a single use. Also, the result from using more than one constraint should be in global minima.

5.3.1 Inclination Anomaly:

The inclination anomaly is the derived quantity using the inclination and declination. In paleomagnetic studies, for the long period, the field is best approximated by geocentric axial dipole (GAD), where the inclination is predicted to vary with latitude (λ) via the axial dipole equation

$$\tan I_{GAD} = 2 \tan \lambda$$

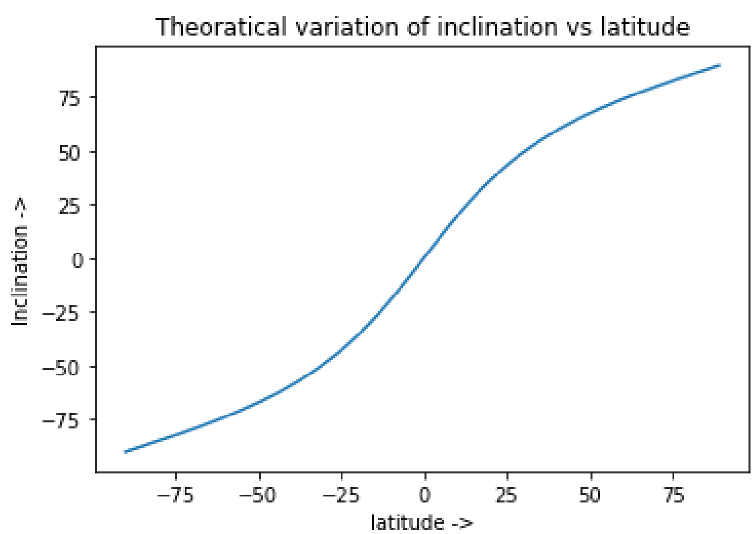


Fig. 5-2: Variation of inclination vs. latitude

However, paleomagnetic records show a small deviation from GAD.

$$\Delta I = \bar{I} - I_{GAD}$$

Here, $\bar{I}$ is the calculated fisher mean inclination values.

Now to make criteria, gauss coefficients of degree 10 up to epoch 1995 and degree 13 from 2000 have been taken. Geomagnetic elements such as Inclination and Declination are calculated using spherical harmonics expansion given in equation (2.9).

Inclination, $$I = tan^{-1}\left(\frac{-B_r}{(B_\theta^2+B_\varphi^2)^{\frac{1}{2}}}\right) \quad (5.1)$$

Declination, $$D = tan^{-1}\left(\frac{B_\varphi}{B_\theta}\right) \quad (5.2)$$

Here, B_r, $B\theta\ and\ B_\varphi$ are magnetic field components along spherical coordinates such as radial, colatitude, and longitude.

To calculation of Fisher mean contains following steps:

To calculate Fisher mean, the ith paleomagnetic direction is $\hat{x}_i$ for a given set of paleomagnetic data, is defined by an inclination-declination pair(I_i, D_i). Cartesian components for the pair will be:

$$x_i = \cos I_i \cos D_i\ ; \quad y = \cos I_i \sin D_i\ ; \quad z_i = \sin I\_i$$

The mean unit direction $\bar{x}$ given by N data is:

$$\bar{x} = \frac{1}{R}\sum_{i=1}^{N} x_i\ ; \quad \bar{y} = \frac{1}{R}\sum_{i=1}^{N} y_i\ ; \quad \bar{z} = \frac{1}{R}\sum_{i=1}^{N} z_i$$

Where,

$$R^2 = \left(\sum_{i=1}^{N} x_i\right) + \left(\sum_{i=1}^{N} y_i\right) + \left(\sum_{i=1}^{N} z_i\right)$$

The off axis angle θ_i of each datum, where

$$cos\theta_i = \bar{x}.\hat{x}_i$$

θ_i Is treated as the Fisher mean of inclination anomaly.

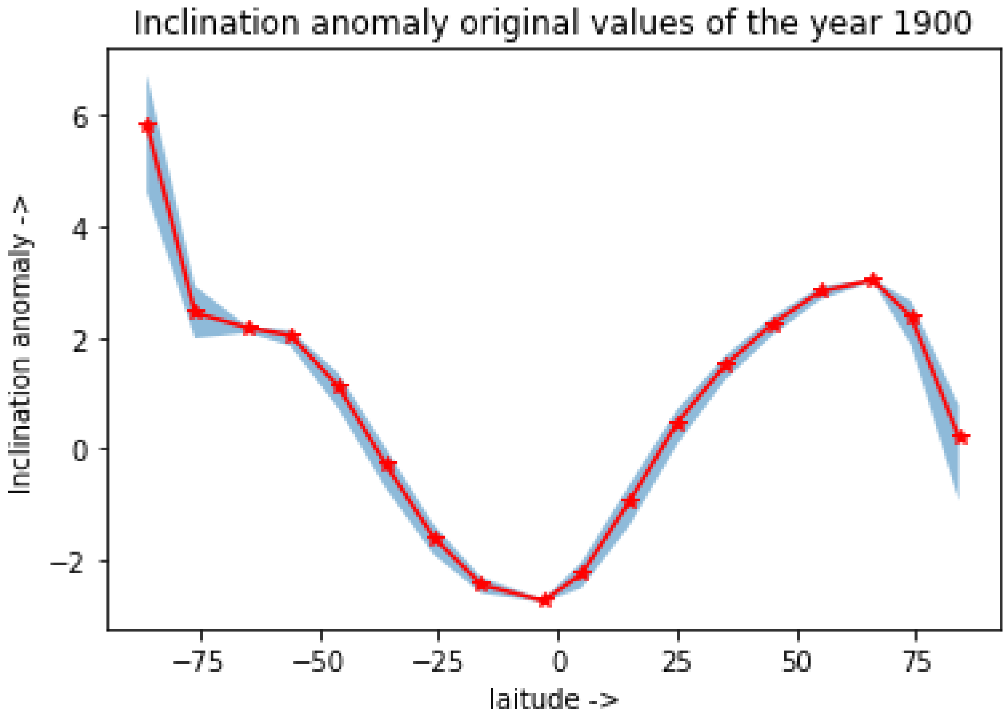


Fig. 5-3: Fisher mean of Inclination anomaly with 10' latitude bins and median (red) with upper and lower bound (blue area) for the year 1900.

The inclination anomaly criterion utilizes the maximum absolute median ΔI (Calculated from 10^{o}) and its 95% confidence intervals as a measure of total average field behavior.

5.3.2 Total field:

The total field is the direct used quantity. The constraints for this quantity are derived using the coefficients from 2000 to 2020. The total magnetic field is taken for all over the globe and analyzed. The total magnetic field intensity is given by:

$$F = \sqrt{B_r^2 + B_\theta^2 + B_\phi^2}$$

Here, B_r, $B\theta$ and B_φ are magnetic field components along spherical coordinates such as radial, colatitude, and longitude. The total magnetic field for the year 2020 is shown below.

Tools:

The whole analysis was executed in Python programming language, in which the following open-source library have been used as NumPy, Pandas, Matplotlib, Scikit-learn, Open Ai, Gym.

Data for Analysis:

The gauss coefficients of IGRF13 model are available at the website of NOAA National Centers for Environmental Information (NCEI). It consists of all the coefficients from epoch 1900 to 2020 and secular variation for 2020 – 2025 used as annually basis.

N^{int} is chosen for 13 for the latest IGRF models, so total Gauss coefficients are 195 for IGRF model from epoch 2000 and 120 coefficients up to epoch 1995. To define the dipole component of fields, coefficient having degree 1 are needed i.e., first three coefficients.

Epoch	g_{01}	g_{11}	h_{11}
2020	-29404.8	-1450.9	4652.5
2015	-29441.46	1501.77	4795.99
2010	-29496.57	1586.42	4944.26
2005	-29554.63	1669.05	5077.99
2000	-29619.4	-1728.2	5186.1
1995	-29692	-1784	5306
1990	-29775	-1848	5406
1985	-29873	-1905	5500
1980	-29992	-1956	5604
1975	-30100	-2013	5675
1970	-30220	-2068	5737
1965	-30334	-2119	5776
1960	-30421	-2169	5791
1955	-30500	-2215	5820
1950	-30554	-2250	5815
1945	-30594	-2285	5810
1940	-30654	-2292	5821
1935	-30715	-2306	5812

1930	-30805	-2316	5808
1925	-30926	-2318	5817
1920	-31060	-2317	5845
1915	-31212	-2306	5875
1910	-31354	-2297	5898
1905	-31464	-2298	5909
1900	-31543	-2298	5922

Table 5-1: Gauss coefficients (g, h) of only 1st degree (n = 1, m = 0, 1) showing from 1900 to 2020 epochs.

CHAPTER-VI

RESULTS AND DISCUSSIONS:

From the Q_{PM}Criteria I have taken the Inclination anomaly. And the Total field is also used as one of my model constraints. In the following points output by inclination anomaly, output by total field and output by combined effect of total field and inclination anomaly is given.

6.1.1 Inclination Anomaly:

From the model, the inclination anomaly is not exceeding the criteria and the following figures shows the result, it is clearly seen that for the year 1900 the change in inclination anomaly contours become wiggly due to increase in resolution. For the

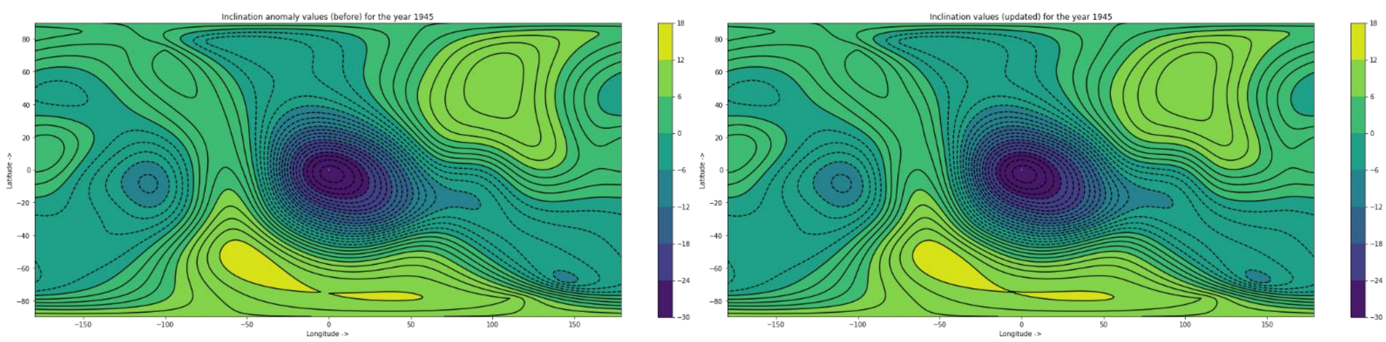


Fig. 6-1: Inclination anomaly (before) for the year 1945 (left). Inclination anomaly (after) for the year 1945 (right).

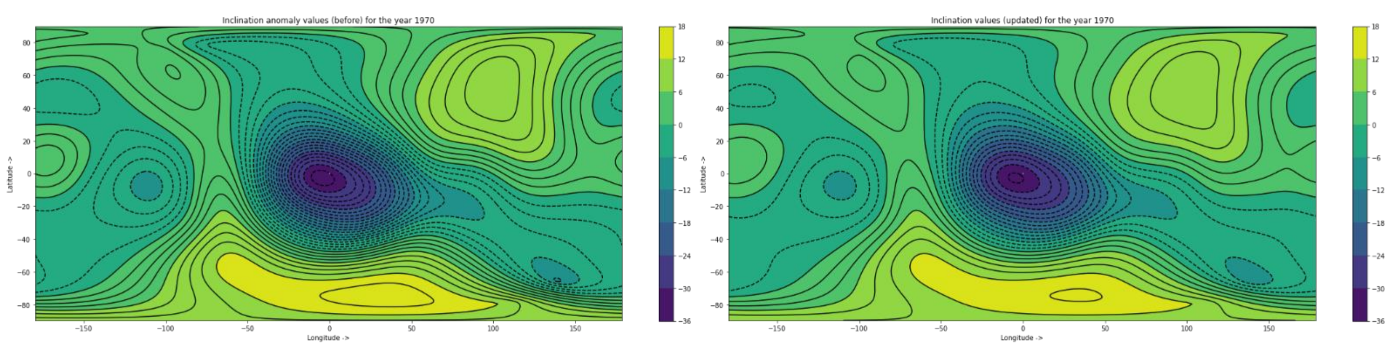


Fig. 6-2: Inclination anomaly (before) for the year 1970 (left). (b) Inclination anomaly (after) for the year 1970 (right).

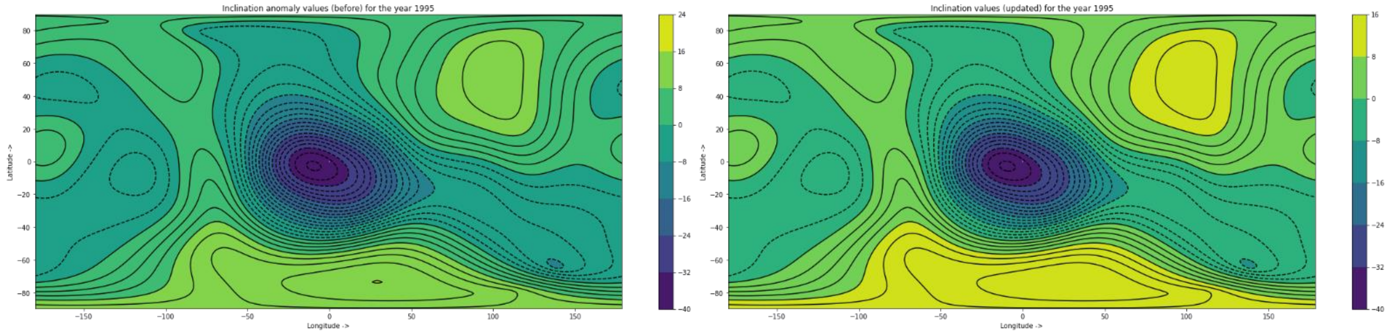


Fig. 6-3: Inclination anomaly (before) for the year 1995 (left). Inclination anomaly (after) for the year 1995 (right).

6.1.2 Total magnetic field:

The total magnetic field is of the range 70000nT. The change in total magnetic field by using degree n =10 and degree n = 13 is less than 1% of average total magnetic field. The following plots show the total magnetic field due to coefficients of degree n = 10 (available), total magnetic field due to coefficients of degree n = 13 (predicted) and the difference of both total magnetic fields.

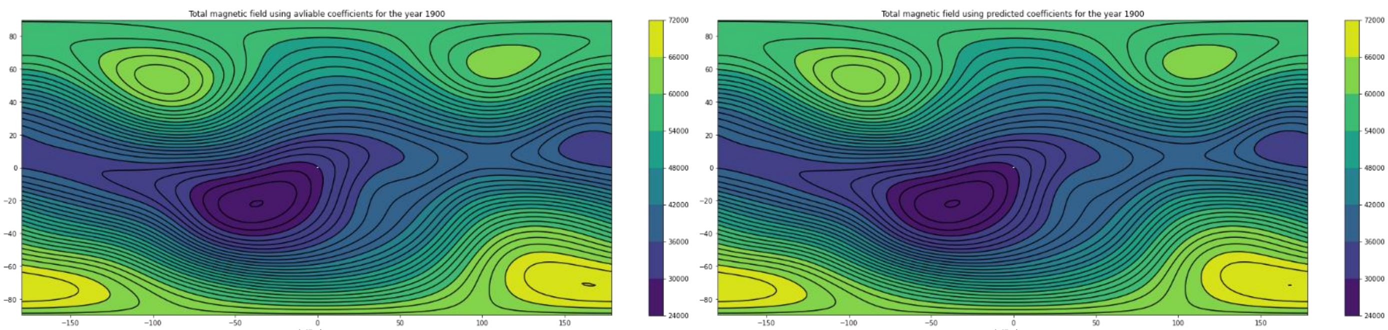


Fig. 6-4: Total magnetic field due from original values for the year 1900 (left). Total magnetic field from the predicted coefficients for the year 1900 (right).

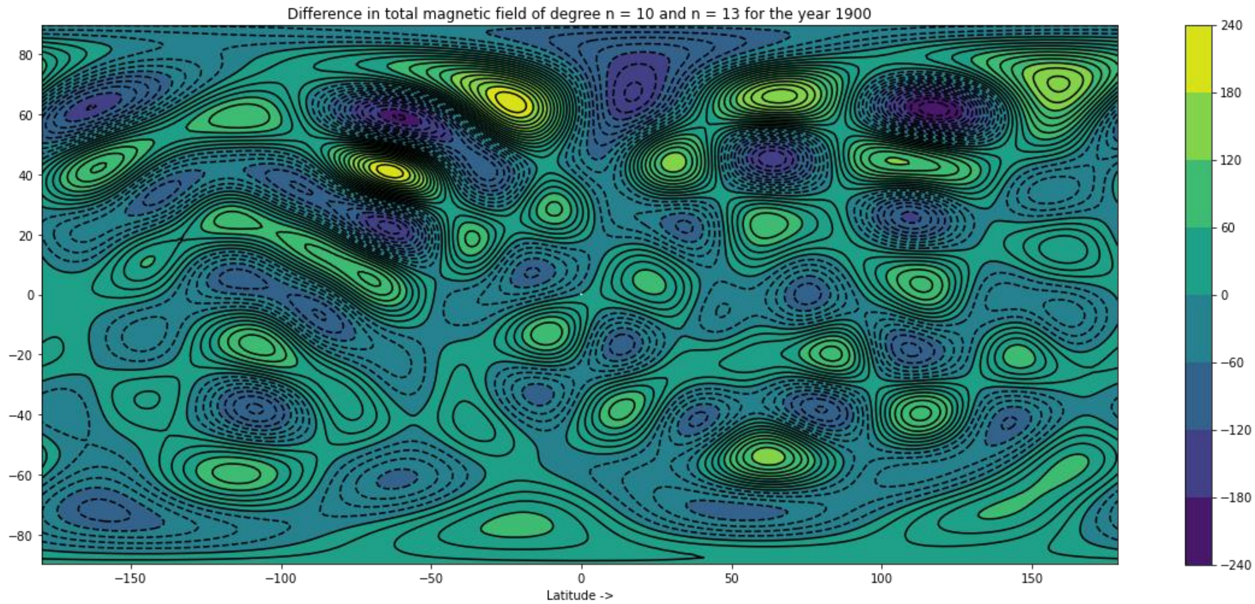


Fig. 6-5: Difference in original field and predicted total magnetic field for the year 1900.

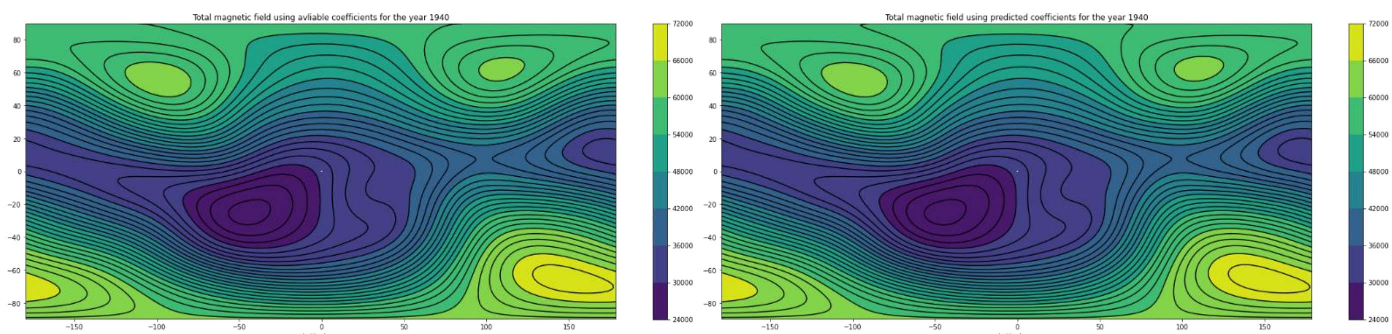


Fig. 6-6: Total magnetic field due from original values for the year 1940 (left). Total magnetic field from the predicted coefficients for the year 1940 (right).

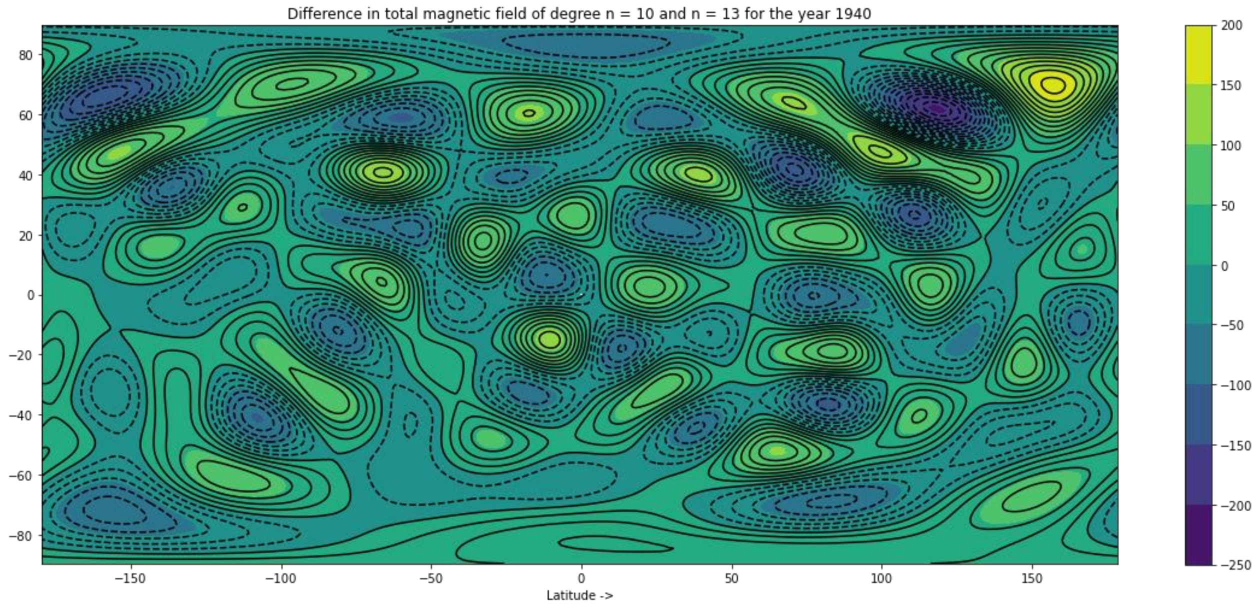


Fig. 6-7: Difference in original field and predicted total magnetic field for the year 1940.

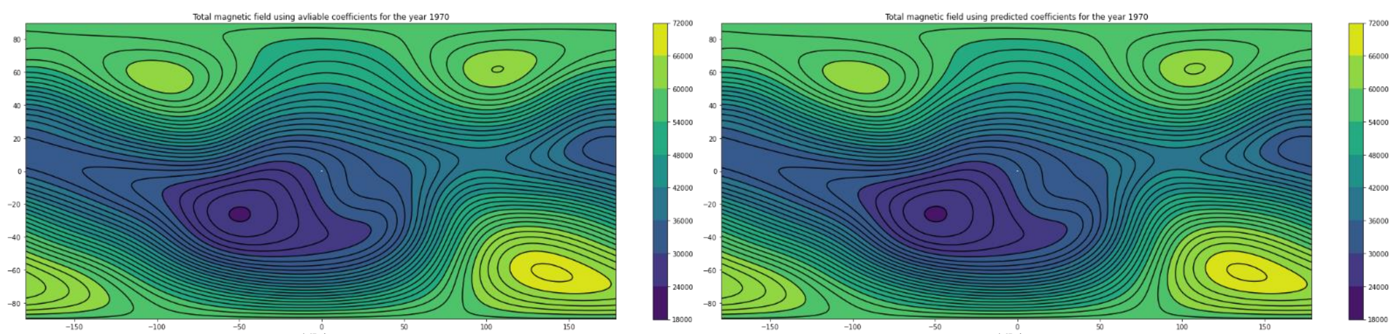


Fig. 6-8: Total magnetic field due from original values for the year 1970 (left). Total magnetic field from the predicted coefficients for the year 1970 (right).

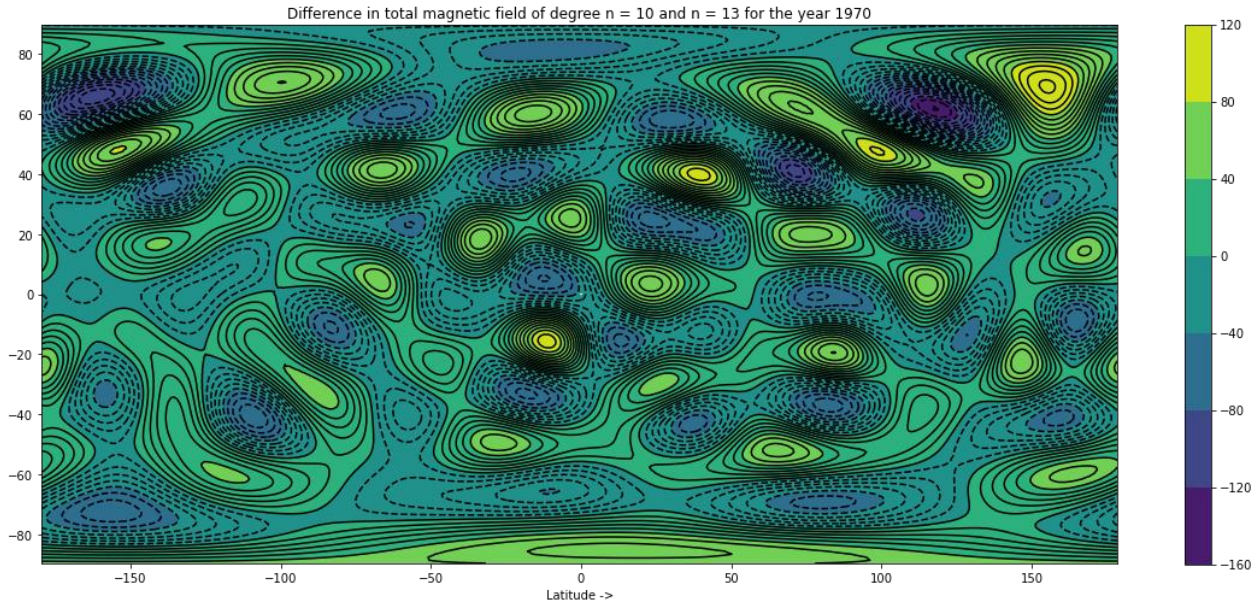


Fig. 6-9: Difference in original field and predicted total magnetic field for the year 1970.

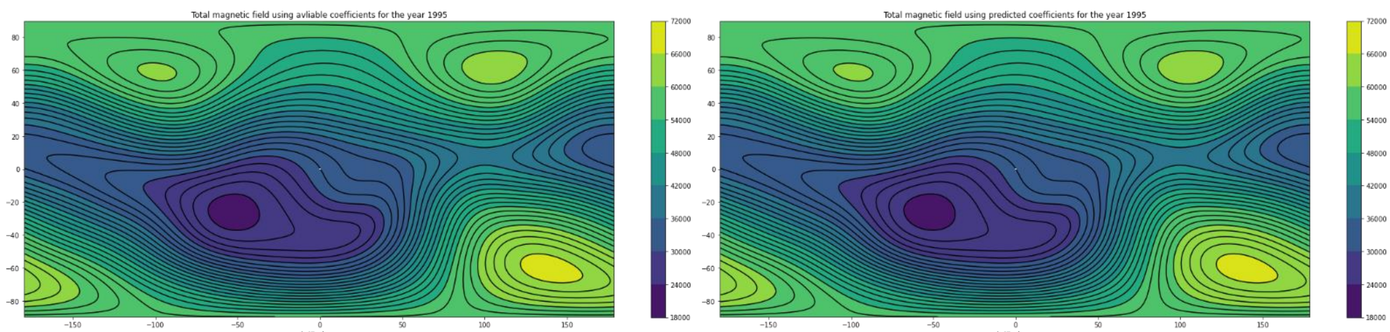


Fig. 6-10: Total magnetic field due from original values for the year 1995 (left). Total magnetic field from the predicted coefficients for the year 1995(right).

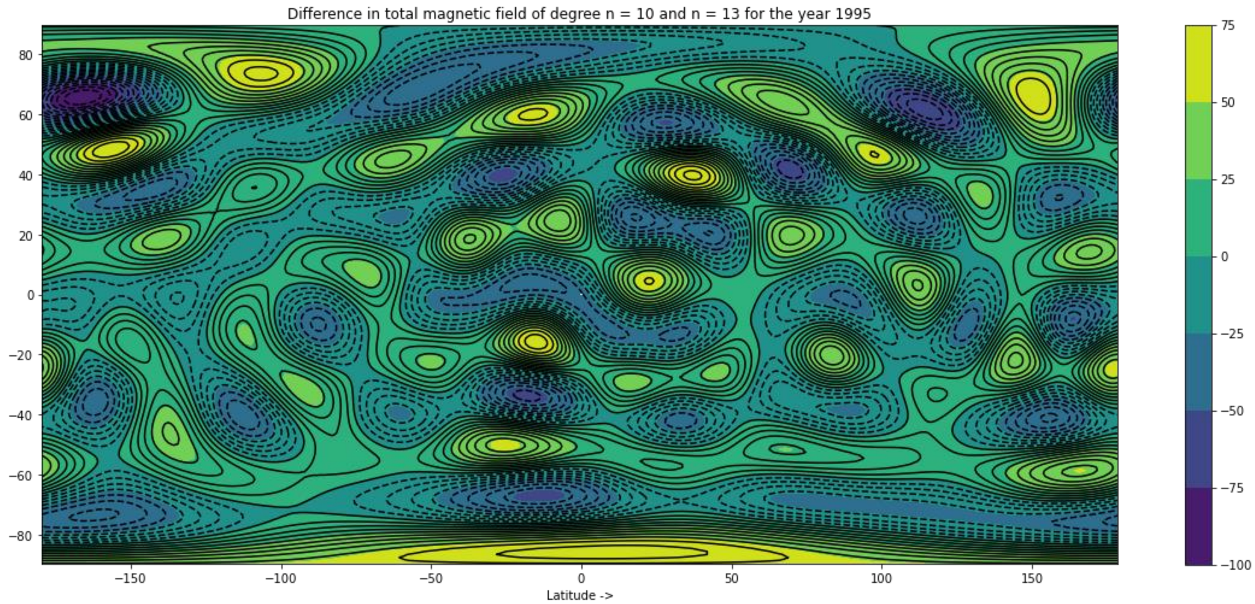


Fig. 6-11: Difference in original field and predicted total magnetic field for the year 1995.

All these results are essentially required to develop a predictive model. Since for modelling purposes large data set is required at corresponding time-stamp. Having large data helps to get our model of better accuracy. It involves training and testing phase in term of machine learning. In our case, goal is to make prediction model at different time scale in past events. But the challenge is availability of less data. To handle with this challenge our approach is to use reinforcement learning algorithm. This type of learning interacts with environment, so in our case we presented criteria-based environment. As we have seen IGRF satisfies model criteria. The model can be updated further for the better IGRF values.

CONCLUSION:

In the present study I have tried to point an approach to make a geomagnetic field model based on Machine learning. I have used reinforcement learning because to construct such model the biggest problem is the limitation of data. Reinforcement learning is suitable approach to deal such data limited problem. But alongside the limited data we have also some physics behind the earth's magnetic field. Some criteria is defined as inclination anomaly and total magnetic field, so that model can fulfill those in order to perform accurately and precisely. More criteria can be added to get the better values of IGRF coefficients.

Initially I have used current geomagnetic field model IGRF-13th generation to assessment the behavior against the criteria which can further play a role as environment in reinforcement learning. The IGRF model has satisfied perfectly those criteria.

In the conclusion I can say that where due to limitation of data, other learning is not performing well, reinforcement learning is a suitable approach to get the model because of its ability to work with the environment.

REFERENCES:

- Alken, P., Thébault, E., Beggan, C.D., Amit, H., Aubert, J., Baerenzung, J., Bondar, T.N., Brown, W.J., Califf, S., Chambodut, A. and Chulliat, A., 2021. International geomagnetic reference field: the thirteenth generation. *Earth, Planets and Space*, *73*(1), pp.1-25.
- https://www.ngdc.noaa.gov/
- Olsen, N., Friis-Christensen, E., Floberghagen, R., Alken, P., Beggan, C.D., Chulliat, A., Doornbos, E., Da Encarnação, J.T., Hamilton, B., Hulot, G. and van den IJssel, J., 2013. The Swarm satellite constellation application and research facility (SCARF) and Swarm data products. *Earth, Planets and Space*, *65*(11), pp.1189-1200.
- Sprain, C.J., Biggin, A.J., Davies, C.J., Bono, R.K. and Meduri, D.G., 2019. An assessment of long duration geodynamo simulations using new paleomagnetic modeling criteria (QPM). *Earth and Planetary Science Letters*, *526*, p.115758.
- Sutton, R.S. and Barto, A.G., 2018. *Reinforcement learning: An introduction*. MIT press.
- Yu, S. and Ma, J., 2021. Deep learning for geophysics: Current and future trends. *Reviews of Geophysics*, *59*(3), p.e2021RG000742.

ANNEXURE-1

Functions:

```
#!/usr/bin/env python
# coding: utf-8

# In[2]:


# program for calculation of the Inclination and decination for all the latitude longitude pairs

import numpy as np
from numpy import degrees, radians
from math import pi
import pandas as pd
import matplotlib.pyplot as plt
import pmagpy.pmag as pmg
import numpy as np
import statistics as s
import warnings
import matplotlib.tri as tri

def inclination(x,y,z):
    '''
    x,y and z are the main component of
    magnetic field in cartesian coordinates.
    return
    inc : Inclination anomaly
    dec : Declination anomaly
    t_field : Total field intensity

    '''
    hor = np.sqrt(x**2+y**2)
    inc = np.arctan(z/hor)
    dec = np.arctan(y/x)

    return inc, dec


def total_field(x,y,z):
    hor = np.sqrt(x**2+y**2)
    t_field = np.sqrt(hor+z*z)
    return t_field
```

```python
def synth_values(coeffs, radius, theta, phi,nmax, nmin, grid=None):

    # ensure ndarray inputs
    coeffs = np.array(coeffs, dtype=np.float)
    radius = np.array(radius, dtype=np.float) / 6371.2  # Earth's average radius
    theta = np.array(theta, dtype=np.float)
    phi = np.array(phi, dtype=np.float)

    if np.amin(theta) <= 0.0 or np.amax(theta) >= 180.0:
        if np.amin(theta) == 0.0 or np.amax(theta) == 180.0:
            warnings.warn('The geographic poles are included.')
        else:
            raise ValueError('Colatitude outside bounds [0, 180].')

    if nmin is None:
        nmin = 1
    else:
        assert nmin > 0, 'Only positive nmin allowed.'

    # handle optional argument: nmax
#    nmax_coeffs = int(np.sqrt(coeffs.shape[-1] + 1) - 1)  # degree
#    if nmax is None:
#        nmax = nmax_coeffs
#    else:
#        assert nmax > 0, 'Only positive nmax allowed.'

#    if nmax > nmax_coeffs:
#        warnings.warn('Supplied nmax = {0} and nmin = {1} is '
#                      'incompatible with number of model coefficients. '
#                      'Using nmax = {2} instead.'.format(
#                        nmax, nmin, nmax_coeffs))
#        nmax = nmax_coeffs

#    if nmax < nmin:
#        raise ValueError(f'Nothing to compute: nmax < nmin ({nmax} < {nmin}.)')

    # handle grid option
    grid = False if grid is None else grid

    # manually broadcast input grid on surface
    if grid:
        theta = theta[..., None]  # first dimension is theta
        phi = phi[None, ...]  # second dimension is phi

    # get shape of broadcasted result
    try:
        b = np.broadcast(radius, theta, phi, np.broadcast_to(0, coeffs.shape[:-1]))
```

```
except ValueError:
    print('Cannot broadcast grid shapes (excl. last dimension of coeffs):')
    print(f'radius: {radius.shape}')
    print(f'theta:  {theta.shape}')
    print(f'phi:    {phi.shape}')
    print(f'coeffs: {coeffs.shape[:-1]}')
    raise

grid_shape = b.shape

# initialize radial dependence given the source
r_n = radius**(-(nmin+2))

# compute associated Legendre polynomials as (n, m, theta-points)-array
Pnm = legendre_poly(nmax, theta)

# save sinth for fast access
sinth = Pnm[1, 1]

# calculate cos(m*phi) and sin(m*phi) as (m, phi-points)-array
phi = radians(phi)
cmp = np.cos(np.multiply.outer(np.arange(nmax+1), phi))
smp = np.sin(np.multiply.outer(np.arange(nmax+1), phi))

# allocate arrays in memory
B_radius = np.zeros(grid_shape)
B_theta = np.zeros(grid_shape)
B_phi = np.zeros(grid_shape)

num = nmin**2 - 1
for n in range(nmin, nmax+1):
    B_radius += (n+1) * Pnm[n, 0] * r_n * coeffs[..., num]

    B_theta += -Pnm[0, n+1] * r_n * coeffs[..., num]

    num += 1

    for m in range(1, n+1):
        B_radius += ((n+1) * Pnm[n, m] * r_n
                     * (coeffs[..., num] * cmp[m]
                        + coeffs[..., num+1] * smp[m]))

        B_theta += (-Pnm[m, n+1] * r_n
                    * (coeffs[..., num] * cmp[m]
                       + coeffs[..., num+1] * smp[m]))

        with np.errstate(divide='ignore', invalid='ignore'):
```

```
            # handle poles using L'Hopital's rule
            div_Pnm = np.where(theta == 0., Pnm[m, n+1], Pnm[n, m] / sinth)
            div_Pnm = np.where(theta == degrees(pi), -Pnm[m, n+1], div_Pnm)

        B_phi += (m * div_Pnm * r_n
                  * (coeffs[..., num] * smp[m]
                     - coeffs[..., num+1] * cmp[m]))

        num += 2

    r_n = r_n / radius  # equivalent to r_n = radius**(-(n+2))

  return B_radius, B_theta, B_phi


def legendre_poly(nmax, theta):

  costh = np.cos(radians(theta))
  sinth = np.sqrt(1-costh**2)

  Pnm = np.zeros((nmax+1, nmax+2) + costh.shape)
  Pnm[0, 0] = 1  # is copied into trailing dimenions
  Pnm[1, 1] = sinth  # write theta into trailing dimenions via broadcasting

  rootn = np.sqrt(np.arange(2 * nmax**2 + 1))

  # Recursion relations after Langel "The Main Field" (1987),
  # eq. (27) and Table 2 (p. 256)
  for m in range(nmax):
    Pnm_tmp = rootn[m+m+1] * Pnm[m, m]
    Pnm[m+1, m] = costh * Pnm_tmp

    if m > 0:
      Pnm[m+1, m+1] = sinth*Pnm_tmp / rootn[m+m+2]

    for n in np.arange(m+2, nmax+1):
      d = n * n - m * m
      e = n + n - 1
      Pnm[n, m] = ((e * costh * Pnm[n-1, m] - rootn[d-e] * Pnm[n-2, m])
                   / rootn[d])

  # dP(n,m) = Pnm(m,n+1) is the derivative of P(n,m) vrt. theta
  Pnm[0, 2] = -Pnm[1, 1]
  Pnm[1, 2] = Pnm[1, 0]
  for n in range(2, nmax+1):
    Pnm[0, n+1] = -np.sqrt((n*n + n) / 2) * Pnm[n, 1]
    Pnm[1, n+1] = ((np.sqrt(2 * (n*n + n)) * Pnm[n, 0]
```

```
                    - np.sqrt((n*n + n - 2)) * Pnm[n, 2]) / 2)

        for m in np.arange(2, n):
            Pnm[m, n+1] = (0.5*(np.sqrt((n + m) * (n - m + 1)) * Pnm[n, m-1]
                          - np.sqrt((n + m + 1) * (n - m)) * Pnm[n, m+1]))

        Pnm[n, n+1] = np.sqrt(2 * n) * Pnm[n, n-1] / 2

    return Pnm


def check_float(s):
    """Convert to float."""
    try:
        return float(s)
    except ValueError:
        raise ValueError(f'Could not convert {s} to float.')

def check_reward(pmed,lb,ub):
    '''
    check the result so that the reward or penelty will be given.

    '''
    check1 = sum(np.array(pmed)>np.array(ub))
    check2 = sum(np.array(pmed)<np.array(lb))

    if check1!=0:
        return -check1
    elif check2!=0:
        return -check2
    else:
        return 200


def find_inc(X,Y,Z):
    inc, dec = inclination(X,Y,Z)
    return inc, dec

def get_B(year,nmax,nmin):
    df = pd.read_csv("IGRF_coeff.csv")
    coeffs = np.array(df[str(year)])
    #try:
        #lats,late,lati = input("insert latitute start, end and interval-> ").split()
    #except:
    lats = -89.5
    late = 90
    lati = 1
```

```
    lons = check_float(-179.5)
    loni = check_float(1)
    lone = check_float(179.5)


    lats=check_float(lats)
    late=check_float(late)
    lati=check_float(lati)


    # Create a meshgrid to fill in the colat/lons

    colat, lon = np.meshgrid(90-np.arange(lats,late,lati),
                             np.arange(lons,lone,loni) )
    colat = 90-np.arange(lats,late,lati)

    #colat = colat.flatten()
    #lon = lon.flatten()
    lat = 90-colat

    # Arrange into a long vector for synth grid
    radius = 6371.2

    # find main components of magnetic field
    Br, Bt, Bp = synth_values(coeffs, radius,colat, lon,nmax,nmin)

    # Rearrange to X, Y, Z components
    X = -Bt; Y = Bp; Z = -Br

    return X, Y, Z



# find the confidance intervel
def confidence_interval(x):
    x.sort()
    N = len(x)
    Q = 0.5
    Z = 1.96
    lb = round(N*Q-Z*np.sqrt(N*Q*(1-Q)))
    ub = round(N*Q+Z*np.sqrt(N*Q*(1-Q)))
    return x[lb],x[ub-1]

# Program to get the inclination anomaly plot of the median of 10' latitude bins and 95% confidance interval
```

```
def get_inc(year,nmax):

    X,Y,Z = get_B(year,nmax,1)
    inc1,dec1 = find_inc(X,Y,Z)
    inc1 = np.array(inc1)*180/np.pi
    dec1 = np.array(dec1)*180/np.pi

    # final inc dec pair for each latitude to calculate the fisher mean
    final2 = []
    for j in range(len(inc1[0])):
        final1 = []
        for i in range(len(inc1)):
            final1.append([dec1[i][j],inc1[i][j]])
        final2.append(final1)

    # calculate fisher mean of all the longitudes and latitudes
    fminc = [0]*len(final2)
    for i in range(len(final2)):
        fminc[i] = pmg.fisher_mean(final2[i])['inc']

    # Calculate Igad
    lambd = np.arange(-90,90,1)
    lambd1 = lambd*np.pi/180
    tan = np.arctan(2*np.tan(lambd1))*180/np.pi

    # calculate anomaly
    anom = fminc-tan


    # plot fisher mean data
#    plt.figure()
#    plt.plot(lambd,fminc,"*")
#    plt.xlabel("laitude ->")
#    plt.ylabel("Fisher mean data ->")
#    plt.title("Inclination fisher mean vs latitude for the year %d"%year)

    # plot anomaly
#    plt.figure()
#    plt.plot(lambd,anom,"*")
#    plt.xlabel("laitude ->")
#    plt.ylabel("Inclination anomaly ->")
#    plt.title("Inclination anomaly vs latitude for the year %d"%year)
    # get list of 10' latitude bins
    i = 0
    lis = []
    while 1:
        lis.append(anom[i:i+10])
```

```
        i+=10
        if i>=len(anom):
            break

    # get the median for each 10' latitude bins
    med = []
    lambd2 = []
    for i in range(len(lis)):
        h = s.median_high(lis[i])
        med.append(h)
        lambd2.append(lambd[np.where(anom==h)])

    # create list for latitude
    lamb = []
    for i in range(len(lambd2)):
        lamb.append(lambd2[i].tolist()[0])

    # find the confidance interval
    ub = [0]*len(lis)
    lb = [0]*len(lis)
    for i in range(len(lis)):
        lb[i],ub[i]=confidence_interval(lis[i])
    # plot confidance interval
#     plt.figure()
#     plt.fill_between(lamb,lb,ub,alpha=0.5)
#     plt.plot(lambd2,med,"r-*")
#     plt.xlabel("laitude ->")
#     plt.ylabel("Inclination anomaly ->")
#     plt.title("Inclination anomaly with 10' latitude bins and 95% confidance interval for the year {}".format(year))

    return ub, lb, med, lamb # upper bound, lower bound, median,
                             # latitude wrt median and t_field



def get_B_pred(coeffs):
    '''
    Get inclination and declination values for the predicted coefficients.

    '''
    nmax=13
    nmin=1
    coeffs = np.array(coeffs)

    lats = -89.5
    late = 90
    lati = 1
```

```
    lons = check_float(-179.5)
    loni = check_float(1)
    lone = check_float(179.5)


    lats=check_float(lats)
    late=check_float(late)
    lati=check_float(lati)


    # Create a meshgrid to fill in the colat/lons

    colat, lon = np.meshgrid(90-np.arange(lats,late,lati),
                             np.arange(lons,lone,loni) )
    colat = 90-np.arange(lats,late,lati)

    #colat = colat.flatten()
    #lon = lon.flatten()
    lat = 90-colat

    # Arrange into a long vector for synth grid
    radius = 6371.2

    # find main components of magnetic field
    Br, Bt, Bp = synth_values(coeffs, radius,colat, lon,nmax,nmin)

    # Rearrange to X, Y, Z components
    X = -Bt; Y = Bp; Z = -Br

    return X,Y,Z

# Program to get the inclination anomaly plot of the median of 10' latitude bins and 95% confidance interval
def get_inc_pred(coeffs):
    '''
    This program will give only the upper bound, lower bound,
    median and latitude for the respected medians for the predicted
    coefficients.
    '''
    nmax = 13
    nmin=1
    X,Y,Z = get_B_pred(coeffs)
    inc1,dec1 = find_inc(X,Y,Z)
    inc1 = np.array(inc1)*180/np.pi
    dec1 = np.array(dec1)*180/np.pi

    # final inc dec pair for each latitude to calculate the fisher mean
```

```
final2 = []
for j in range(len(inc1[0])):
    final1 = []
    for i in range(len(inc1)):
        final1.append([dec1[i][j],inc1[i][j]])
    final2.append(final1)

# calculate fisher mean of all the longitudes and latitudes
fminc = [0]*len(final2)
for i in range(len(final2)):
    fminc[i] = pmg.fisher_mean(final2[i])['inc']

# Calculate Igad
lambd = np.arange(-90,90,1)
lambd1 = lambd*np.pi/180
tan = np.arctan(2*np.tan(lambd1))*180/np.pi

# calculate anomaly
anom = fminc-tan

# get list of 10' latitude bins
i = 0
lis = []
while 1:
    lis.append(anom[i:i+10])
    i+=10
    if i>=len(anom):
        break

# get the median for each 10' latitude bins
med = []
lambd2 = []
for i in range(len(lis)):
    h = s.median_high(lis[i])
    med.append(h)
    lambd2.append(lambd[np.where(anom==h)])

# create list for latitude
lamb = []
for i in range(len(lambd2)):
    lamb.append(lambd2[i].tolist()[0])

# find the confidance interval
ub = [0]*len(lis)
lb = [0]*len(lis)
for i in range(len(lis)):
    lb[i],ub[i]=confidence_interval(lis[i])
```

```
    return ub, lb, med, lamb # upper bound, lower bound, median, latitude wrt median and t_field




# used linear regression to find the first possible values of the coefficients
def start():
    df = pd.read_csv("IGRF_Coeff.csv")
    c = df.iloc[120:][("2000 2005 2010 2015 2020").split()]
    pco = []
    for i in range(len(c)):
        model = np.polyfit(list(c.columns.astype(int)),c.iloc[i],1)
        predict = np.poly1d(model)
        val = np.arange(1900,1996,5)
        pcoeff = predict(val)
        pco.append(pcoeff)
    return pco

def env(pred_coeffs,year):

    # nmax
    if year<=1995:
        nmax=10
    else:
        nmax=13

# For inclination anomaly
    update_coeff2 = list(range(-75,0,1))
    reward1=0
    #get original upperbound, lower bound, median and lambda
    ub,lb,med,lamb = get_inc(year,nmax)

# For Total field
    update_coeff1 = list(range(-75,0,1))
    t_field = get_total_field(year,nmax,nmin=1)
    reward2=0


    for i in range(10):
        print('\n')
        print("working on inclination anomaly and total field one by one")
        for count in update_coeff2:
            pub, plb, pmed,plamb = get_inc_pred(pred_coeffs)
            #temp = np.array(pred_coeffs)

            reward1 = check_reward(pmed,lb,ub)
```

```
    if reward1==200:
        break

    temp1 = get_inc_pred(update_coeffs1(pred_coeffs,count))[2]
    temp2 = get_inc_pred(update_coeffs2(pred_coeffs,count))[2]

    rms = rmserror(pmed,med)
    rms1 = rmserror(temp1,med)
    rms2 = rmserror(temp2,med)
    if rms1<rms:
        pred_coeffs = update_coeffs1(pred_coeffs,count)
    elif rms2<rms:
        pred_coeffs = update_coeffs2(pred_coeffs,count)
    else:
        update_coeff2.pop(update_coeff2.index(count))
    #print("Year : "+str(year)+"; for : "+str(i)+"; count : "+str(count),file=open('output.txt', 'a'))
    #print(rms,file=open('output.txt', 'a'))
    #print("Year : "+str(year)+"; for : "+str(i)+"; count : "+str(count))
    #print(rms)
print("Year : "+str(year)+"; for : "+str(i))
if reward1 == 200 or len(update_coeff2)<=0:
    break
#print(update_coeff2,file=open('output.txt', 'a'))
#print(update_coeff2)
incrms=rms

pred_coeffs = np.array(pred_coeffs)
for count in update_coeff1:
    tp_field = get_total_predicted_field(pred_coeffs)

    rms = rmserror(tp_field,t_field)
    if rms<=50:
        reward2=200
        break

    temp1 = get_total_predicted_field(update_coeffs1(pred_coeffs,count))
    temp2 = get_total_predicted_field(update_coeffs2(pred_coeffs,count))

    rms1 = rmserror(temp1,t_field)
    rms2 = rmserror(temp2,t_field)

    if rms1<rms:
        pred_coeffs = update_coeffs1(pred_coeffs,count)
    elif rms2<rms:
        pred_coeffs = update_coeffs2(pred_coeffs,count)
    else:
        update_coeff1.pop(update_coeff1.index(count))
```

```python
            #print("Year : "+str(year)+"; for : "+str(i)+"; count : "+str(count))
            #print(rms)
        print("Year : "+str(year)+"; for : "+str(i))
        if reward2 == 200 or len(update_coeff1)<=0:
            break
        #print(update_coeff1)
        tfrms = rms
    # plot incliation anomaly curve for predicted coefficients
#    plt.figure()
#    plt.fill_between(lamb,lb,ub,alpha=0.5)
#    plt.plot(plamb,pmed,"r-*")
#    plt.xlabel("laitude ->")
#    plt.ylabel("Inclination anomaly->")
#    plt.title("Inclination anomaly with predicted coefficients for the year {}".format(year))
    return pred_coeffs,reward1+reward2,incrms,tfrms

def doit(year):
    #year = int(input("year -> "))
    p = start()
    df1 = pd.DataFrame(p,columns = np.arange(1900,1996,5))
    df = pd.read_csv("IGRF_Coeff.csv")
    pred_coeff = np.array(df[str(year)].dropna().append(df1[year]))
    incerror = []
    tferror = []
    count=0
    rt=0
    try:
        while True:
            pred_coeff,rt,incrms1,tfrms1 = env(pred_coeff,year)
            count+=1
            incerror.append(incrms1)
            tferror.append(tfrms1)

            print('\n'+"inclination error")
            print(incerror)
            print("\n"+"total field error")
            print(tferror)
            print("reward : "+str(rt))

            if len(incerror)>5:
                if incerror[-5]<=incerror[-1]:
                    print("program terminated due to increase in rms error in inclination")
                    break
                if tferror[-5]<=tferror[-1]:
                    print("program terminated due to increase in rms error in total field error")
                    break
```

```
#     print("Doing again",file=open("output.txt",'a'))
#     print("Doing again")
#     final_coeffs = env(final_coeffs,year)
      #print(pred_coeff,file=open('output.txt', 'a'))
      #print("year = {}; Reward = {}".format(year,env(pred_coeff,year)))
          print('\n'+'the count reached to: ')
          print(count)
    except:
      print('\n'+"coefficients : ")
      print(pred_coeff)
      print('\n'+"reward : "+str(rt))
      print('\n'+"incerror : ")
      print(incerror)
      print('\n'+'tferror : ')
      print(tferror)

    return pred_coeff,rt,count


def rmserror(pmed,med):
    return np.sqrt(np.mean(np.square(np.array(pmed)-np.array(med))))

def update_coeffs1(pred_coeffs,count):
    pred_coeffs[count] = pred_coeffs[count]-0.2
    return pred_coeffs

def update_coeffs2(pred_coeffs,count):
    pred_coeffs[count] = pred_coeffs[count]+0.2
    return pred_coeffs

def get_total_field(year,nmax,nmin):
    X,Y,Z = get_B(year,nmax,nmin)
    t_field = total_field(X,Y,Z)
    return t_field

def get_total_predicted_field(pred_coeffs):
    X,Y,Z = get_B_pred(pred_coeffs)
    tp_field = total_field(X,Y,Z)
    return tp_field


def plot_contour_map1(coeff,year):
    field = get_total_predicted_field(coeff)
    field = field.flatten()
    lat = np.arange(-89.5,90,1)
    lon = np.arange(-179.5,179,1)
    # Create a meshgrid to fill in the colat/lons
```

```
    lat, lon = np.meshgrid(lat,lon)
    # Arrange into a long vector for synth grid
    lat = lat.flatten()
    lon = lon.flatten()

    min_radius = 0.5
    x = lon
    y = lat
    z = field
    triang = tri.Triangulation(x, y)
    triang.set_mask(np.hypot(x[triang.triangles].mean(axis = 1),y[triang.triangles].mean(axis = 1))< min_radius)

    fig1, ax1 = plt.subplots(figsize = (20,12))
    ax1.set_aspect('equal')
    tcf = ax1.tricontourf(triang, z)
    fig1.colorbar(tcf)
    ax1.tricontour(triang, z, colors ='k',levels=30)
    ax1.set_title('total field using the predicted coeffiencents for the year {0}'.format(year))
    ax1.set_xlabel('Longitude ->')
    ax1.set_xlabel('Latitude ->')
    plt.show()

def plot_contour_map2(field,year):
    field = field.flatten()
    lat = np.arange(-89.5,90,1)
    lon = np.arange(-179.5,179,1)
    # Create a meshgrid to fill in the colat/lons
    lat, lon = np.meshgrid(lat,lon)
    # Arrange into a long vector for synth grid
    lat = lat.flatten()
    lon = lon.flatten()

    min_radius = 0.5
    x = lon
    y = lat
    z = field
    triang = tri.Triangulation(x, y)
    triang.set_mask(np.hypot(x[triang.triangles].mean(axis = 1),y[triang.triangles].mean(axis = 1))< min_radius)

    fig1, ax1 = plt.subplots(figsize = (20,12))
    ax1.set_aspect('equal')
    tcf = ax1.tricontourf(triang, z)
    fig1.colorbar(tcf,shrink=0.687)
    ax1.tricontour(triang, z, colors ='k',levels=30)
#     ax1.set_title('total field using the predicted coeffiencents for the year {0}'.format(year))
    ax1.set_xlabel('Longitude ->')
    ax1.set_xlabel('Latitude ->')
```

Plots:

```
import matplotlib.pyplot as plt
import matplotlib.tri as tri
import numpy as np

year = int(input())

# Get the original values
upper_bound,lower_bound,median,res_lam,inc1 = get_inc(year,10)

# Get the fill the missing values using linear regression
def start():
    df = pd.read_csv("IGRF_Coeff.csv")
    c = df.iloc[120:][("2000 2005 2010 2015 2020").split()]
    pco = []
    for i in range(len(c)):
        model = np.polyfit(list(c.columns.astype(int)),c.iloc[i],1)
        predict = np.poly1d(model)
        val = np.arange(1900,1996,5)
        pcoeff = predict(val)
        pco.append(pcoeff)
    return pco

p = start()
df1 = pd.DataFrame(p,columns = np.arange(1900,1996,5))
df = pd.read_csv("IGRF_Coeff.csv")
pred_coeff = np.array(df[str(year)].dropna().append(df1[year]))
ub,lb,med,lamb,inc = get_inc_pred(pred_coeff)

# Get the final predicted values
df_inc = pd.read_csv('inc_pred_coeff11.csv')[str(year)]
pub,plb,pmed,plamb,inc2 = get_inc_pred(df_inc)


# Plot values found using linear regression
plt.figure()
plt.fill_between(res_lam,lower_bound,upper_bound,alpha=0.5)
plt.plot(lamb,med,"r-*")
plt.xlabel("laitude ->")
plt.ylabel("Inclination anomaly ->")
plt.title("predicted inclination anomaly (before) for the year {}".format(year))

# Plot final predicted values
plt.figure()
plt.fill_between(res_lam,lower_bound,upper_bound,alpha=0.5)
```

```
plt.plot(plamb,pmed,"r-*")
plt.xlabel("laitude ->")
plt.ylabel("Inclination anomaly ->")
plt.title("predicted inclination anomaly (updated) for the year {}".format(year))

# Plot Inclination for dipole for any year
lat = np.arange(-89.5,90,1)
lon = np.arange(-179.5,179,1)
# Create a meshgrid to fill in the colat/lons
lat, lon = np.meshgrid(lat,lon)
# Arrange into a long vector for synth grid
lat = lat.flatten()
lon = lon.flatten()

min_radius = 0.5
x = lon
y = lat
inc_dipole = get_inc(year,1)[4]
inc_dipole = inc_dipole.flatten()
z = inc_dipole
min_radius = 0.35
triang = tri.Triangulation(x, y)
triang.set_mask(np.hypot(x[triang.triangles].mean(axis = 1),
                         y[triang.triangles].mean(axis = 1))
                < min_radius)

fig1, ax1 = plt.subplots(figsize = (20,12))
ax1.set_aspect('equal')
tcf = ax1.tricontourf(triang, z)
fig1.colorbar(tcf,shrink=0.687)
ax1.tricontour(triang, z, colors ='k',levels=30)
ax1.set_title('Inclination values for dipole')
ax1.set_ylabel('Latitude ->')
ax1.set_xlabel('Longitude ->')
plt.show()


# Plot inclination anomaly for original values
inc1 = inc1.flatten()
z = inc1-inc_dipole
triang = tri.Triangulation(x, y)
triang.set_mask(np.hypot(x[triang.triangles].mean(axis = 1),
                         y[triang.triangles].mean(axis = 1))
                < min_radius)

fig1, ax1 = plt.subplots(figsize = (20,12))
ax1.set_aspect('equal')
```

```
tcf = ax1.tricontourf(triang, z)
fig1.colorbar(tcf,shrink=0.687)
ax1.tricontour(triang, z, colors ='k',levels=30)
ax1.set_title('Inclination anomaly values (before) for the year {}'.format(year))
ax1.set_ylabel('Latitude ->')
ax1.set_xlabel('Longitude ->')
plt.show()

# Plot inclination anomaly for final predicted values
inc2 = inc2.flatten()
z = inc2-inc_dipole
triang = tri.Triangulation(x, y)
triang.set_mask(np.hypot(x[triang.triangles].mean(axis = 1),
                         y[triang.triangles].mean(axis = 1))
                < min_radius)

fig1, ax1 = plt.subplots(figsize = (20,12))
ax1.set_aspect('equal')
tcf = ax1.tricontourf(triang, z)
fig1.colorbar(tcf,shrink=0.687)
ax1.tricontour(triang, z, colors ='k',levels=30)
ax1.set_title('Inclination values (updated) for the year {}'.format(year))
ax1.set_ylabel('Latitude ->')
ax1.set_xlabel('Longitude ->')
plt.show()

# Plot difference in inclination anomaly
inc2 = inc2.flatten()
z = inc2-inc1
triang = tri.Triangulation(x, y)
triang.set_mask(np.hypot(x[triang.triangles].mean(axis = 1),
                         y[triang.triangles].mean(axis = 1))
                < min_radius)

fig1, ax1 = plt.subplots(figsize = (20,12))
ax1.set_aspect('equal')
tcf = ax1.tricontourf(triang, z)
fig1.colorbar(tcf,shrink=0.687)
ax1.tricontour(triang, z, colors ='k',levels=30)
ax1.set_title('difference in inclination anomaly for the year {}'.format(year))
ax1.set_ylabel('Latitude ->')
ax1.set_xlabel('Longitude ->')
plt.show()


# Plot total field for original values
# import myfunctions as mf
```

```python
dfn0 = pd.read_csv("IGRF_Coeff.csv")[str(year)]
tf_org = get_total_field(year,10,1)
mf.plot_contour_map2(tf_org,year)
plt.title('Total magnetic field using avliable coefficients for the year {}'.format(year))

# Plot total field for predicted values
dfn0 = pd.read_csv("pred_total_field.csv")[str(year)]
tf_pred = get_total_predicted_field(dfn0,nmax=13,nmin=1)
mf.plot_contour_map2(tf_pred,year)
plt.title('Total magnetic field using predicted coefficients for the year {}'.format(year))

# Plot difference in total field
tf_diff1 = np.array(tf_org) - np.array(tf_pred)
mf.plot_contour_map2(tf_diff1,year)
plt.title('Difference in total magnetic field of degree n = 10 and n = 13 for the year {}'.format(year))

# Plot absolute difference in total field
tf_diff2 = abs(tf_org - tf_pred)
mf.plot_contour_map2(tf_diff2,year)
plt.title('Absolute difference in total magnetic field of degree n = 10 and n = 13 for the year {}'.format(year))
```

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Himanshu Jain
Assignment title:	Master's Thesis 2022-23
Submission title:	20MC0039_Himanshu
File name:	Himanshu_Jain.docx
File size:	6.82M
Page count:	35
Word count:	5,236
Character count:	27,295
Submission date:	15-Nov-2022 08:12PM (UTC+0530)
Submission ID:	1952300124

ABSTRACT

Majority of magnetic field of the Earth originate from outer core region. Models are required to observe the variation and study the origin of magnetic field all over the globe. Also, magnetic field changes abruptly through time and location on surface. Prior to advancement of satellite measurement, we have limited observed data. Our aim to develop a model which has ability to predict field value in the past and also to have a better resolution. In order to develop model, evolution is required to have more accuracy. But due to lack of data it is hard to made accurate model for field data prediction.

Hence our approach is to use the Reinforce learning along with physics behind its nature of variation. We require some paleomagnetic criteria to further evaluate our model. In this work we have tried to find an approach to use reinforcement learning technique which might be helpful to design a model that can predict earth's magnetic field in past events with high resolution. Also, we have tried to analyze the paleomagnetic criteria and measure the fit level with the current globally accepted model IGRF-13.

i

20MC0039_Himanshu

ORIGINALITY REPORT

7%	4%	3%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	livrepository.liverpool.ac.uk Internet Source	1%
2	mrstag.com Internet Source	1%
3	Submitted to Georgia Institute of Technology Main Campus Student Paper	1%
4	Basu, . "Z", Dictionary of Geophysics Astrophysics and Astronomy, 2001. Publication	1%
5	Submitted to Dhirubhai Ambani Institute of Information and Communication Student Paper	<1%
6	Submitted to The Rockhampton Grammar School Student Paper	<1%
7	upcommons.upc.edu Internet Source	<1%
8	"Spacecraft Attitude Determination and Control", Springer Science and Business	<1%

Media LLC, 1978
Publication

9	Submitted to Liverpool Institute for Performing Arts Student Paper	<1 %
10	etheses.whiterose.ac.uk Internet Source	<1 %
11	towardsdatascience.com Internet Source	<1 %
12	utpedia.utp.edu.my Internet Source	<1 %
13	"Automation Control Theory Perspectives in Intelligent Systems", Springer Science and Business Media LLC, 2016 Publication	<1 %
14	"The Earth’s Magnetic Field", The Earth’s Magnetism, 2006 Publication	<1 %

Exclude quotes	Off	Exclude matches	< 14 words
Exclude bibliography	Off		