FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Initial Condition Estimation in Flux Tube Simulations using Machine Learning

Ana Filipa Sousa Barros

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: André Monteiro de Oliveira Restivo

Co-Supervisor: João José de Faria Graça Afonso Lima

Co-Supervisor: Rui Nelson Ribeiro Ferreira Pinto

July 15, 2021

Abstract

Space weather has become an essential field of study as solar flares, coronal mass ejections, and other phenomena can severely impact Earth's life as we know it.

The solar wind is threaded by magnetic flux tubes that extend from the solar atmosphere to distances beyond the solar system boundary. As those flux tubes cross the Earth's orbit, it is essential to understand and predict solar phenomena' effects at 1 au, but the physical parameters linked to the solar wind formation and acceleration processes are not directly observable.

Some existing models, such as MULTI-VP, try to fill this gap by predicting the background solar wind's dynamical and thermal properties from chosen magnetograms and using a coronal field reconstruction method. However, this kind of simulation takes a long time to stabilize, although they converge faster with better initial guesses.

As such our main research goal consisted in understanding how to decrease *MULTI-VP*'s simulation time by performing good initial guesses.

The State of the Art shows that there is an increasing interest in using machine learning techniques to solve solar weather forecasting and classification problems whilst having an increase in the interest in applying deep learning techniques to such problems. It also showcases that there is pertinence in applying neural networks to the MULTI-VP simulation in order to improve simulation times.

To address this problem then, we started by using a sampled line of each of the 12k plus simulation generated files and started by performing preprocessing on the data and executing an exploratory data analysis. We showcased that some of the features possessed a considerable range of values and that some had a lot of outliers. We were also able to find some correlations between data concluding that the further away from the Sun, the bigger the diameter of the flux tube and the larger the wind velocity.

We then proceeded by performing data normalization and justified chosen configurations through loss measures and by the use of k-fold validation and tuners. Different models were tested and explained and compared with median and random based models outperforming these. Our results were then further analysed in comparison with the expected outputs proving not to be similar enough to the expected predictions.

Given the issues faced, we then developed and tested a new model by using 6000 simulation generated files instead of the lines as inputs and were faced with similar results.

We then questioned if the complexity of the developed network was high enough and decided to test the use of a higher level of layers on the network. With the final described model we achieved results that were more similar to those expected and that could be used in the *MULTI-VP* simulation.

As such we provided the simulation with an initial guess of all the outputs' lines in a profile so that the simulation could convert faster to a final result.

Afterward, we predicted 15 files with our model and used such files on the MULTI-VP simulation so as to compare their performance to the one achieved by standard files. We were then able to conclude that in the majority of cases, there is a small but existing improvement in the speedup of convergence times. Finally, we validated our result's statistical significance with the student's t-test achieving statistical significance at $p < 0.05$.

We further identified remaining open challenges to be tackled in the future.

Keywords: Initial Conditions Estimation, Machine Learning, Neural Networks, Solar Wind

Acknowledgements

A lot of people have made this dissertation and the completion of my Master's degree possible and I could not not thank them.

Firstly I would like to thank my supervisor André Restivo from whom I have never faced rejection when inviting him to all kinds of projects and who has given me an immeasurable amount of support at all hours with this work. I would also like to thank my co supervisors João Lima and Rui Pinto for all the insights on a topic that was so foreign though exciting to me and for providing insightful help with all aspects of this dissertation.

I would like to thank my friends who provided me some of the best memories of my life and supported me emotionally through many losses and whose presence I shall never question. I would also like to give a special thanking note to my friends Mafalda Falcão and Hugo Ferreira who besides emotional support gave me invaluable help during late night hours with the understanding of machine learning principles and who put up with my questions over and over again.

I would like to thank all the teachers that have taught me throughout my life. Your lessons and experiences are more important to student's lives than you will ever know.

A thank you to my mother who has always encouraged me to follow my dreams and was always there for me. And to my father who has served as a guiding light in times of need.

To my partner JP for the immeasurable support in all aspects of life and without whom life would not be the same.

Filipa Barros

To my grandpa Belmiro Sousa,
The kindest human I've ever known and whose promise I've now redeemed.

Contents

List of Figures

List of Tables

Abbreviations

1LPT	First order Lagrangian Perturbation Theory
2LPT	Second order Lagrangian Perturbation Theory
AI	Artificial Intelligence
au	Astronomical Unit
ANN	Artificial Neural Network(s)
AR	Active Region(s)
API	Application Program Interface
CAT-PUMA	CME Arrival Time Prediction Using Machine learning Algorithms
CME	Coronal Mass Ejection(s)
CFD	Computational Fluid Dynamics
CPU	Computer Processing Unit
DT	Decision Tree(s)
d	Index of Agreement, distance
D	Direction
EAC-DC	Evidence Accumulating Clustering with Dual Rooted Prim tree Cuts
FFNN	Forward-Fed Artificial Neural Network
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
GUI	Graphical User Interface
HSS	High Speed Stream(s)
KGE	Kling-Gupta Efficiency
KNN	K-Nearest Neighbor
LSTM	Long Short-Term Memory
MHD	Magneto-Hydrodynamics
ML	Machine Learning
MLPs	Multilayer Perceptrons
MSE	Mean Squared Error
NARX	Nonlinear AutoRegressive Moving Average with eXogenous inputs
NN	Neural Network
NOAA	US National Oceanic and Atmospheric Administration
NSE	Nash-Sutcliffe Efficiency
P	Precipitation
PET	Precipitation Temperature and Evapotranspiration
Pi	Persistence index
POD	Probability of Detection
POFD	Probability of False Detection(POFD)
RF	Random Forest(s)

RMS	Root Mean Square
RMSE	Rooted Mean Squared Error
RSPs	Radio Solar bursts
SC	Solar Cycle
SEP	Solar Energetic Particles
SIR	Stream-interaction Region(s)
SPRINTS	Space Radiation Intelligence System
SSN	Sunspot Number
STEREO	Spacecraft Away from the Sun-Earth Line
SVM	Support Vector Machine(s)
T	Temperature
TN	True Negative
TP	True Positive
TSS	True Skill Statistics
TT	Transit Times
VE	Volumetric efficiency
WDC	World Data Center for Geomagnetism
XRA	X-Ray Flare(s)
ZA	Zel'dovich approximation

Chapter 1

Introduction

In this chapter, an overview of the context, motivation, problem definition, goals of this dissertation, and document structure are given in the following sections.

1.1 Context

Space Weather Science is defined as the understanding of the chain of causality between physical phenomena that cover a wide range of regimes, from Sun to Earth (as well as other positions in space). Solar events are called geo-effective when their intrinsic properties, propagation path, and configuration regarding Earth's magnetosphere can produce atmospheric or ground-level disturbances; such disturbances can be of different types. Explosive phenomena called coronal mass ejections (CME) produce the strongest and less predictable impacts on Earth (radio-communication failures, ionospheric disturbances that lead to failure in global positioning systems, high-energy radiation on aircraft crews, and passengers). Fluctuations intrinsic to the background solar wind (*i.e.*, the CME-free plasma flow that fills the whole interplanetary medium) come in the form of waves, high-speed streams (HSS), or stream-interaction regions (SIR) and can produce similar effects with smaller amplitudes but with much higher occurrence rates.

1.2 Motivation

Early detection and forecast of CMEs remains highly challenging to this day, as impulsive phenomena trigger them on the Sun. However, some specific patterns visible in ultra-violet images of the Sun seem to be related to the early phases of CME evolution (a straightforward verification of CME precursors is still work-in-progress). The background solar wind follows a more deterministic phenomenology, but many of the physical parameters that determine its structure are not directly observable. This information gap is being progressively filled with the solar wind's physical modeling, which has benefited enormously over the last years. Physical models, nevertheless, always need to rely on heuristics to some extent, and more so in a context where real-time modeling is required. Accurate forecasting of the solar wind background is a significant endeavor to estimate its intrinsic disturbances and better and immediately determine the trajectories of CMEs after they are triggered.

Current knowledge, methods, and tools have been developed historically by different scientific communities. The underlying reason for this separation is intrinsically related to the availability of data: remote observations of the solar surface and geophysical observations. It has become evident over the last two decades that the Sun-to-Earth connections had to be addressed directly. A new community started developing at the interface between geo and solar physics, building the ground base for Space Weather Science and operational and industrial applications. A new generation of ground-based and space-borne observatories geared at probing the physical processes at play between the perturbation source (the Sun) and the target (Earth). Examples of these are space probes that scan intermediate orbital altitudes. (Parker Solar Probe[1] Solar Orbiter[2], BepiColombo[3]), radio probes, spacecraft away from the Sun-Earth line (STEREO, the future Lagrange), and remote monitoring of coronal solar magnetic fields. Despite the enormous growth in variety and volume of data, the physical coverage of the vast Sun-Earth domain remains very low (in comparison, *e.g.*, to the ocean and atmospheric monitoring for Earth meteorology). Extensive physical modeling of the Sun-Earth system is thus necessary, especially techniques that integrate data and models seamlessly. Space Weather forecasting furthermore requires a quick, reliable, and systematic analysis of increasingly more extensive and more diverse datasets and calls for the implementation of machine learning and deep learning methods (Artificial Intelligence (AI), generically). The solar and space physics community has been actively seeking event and feature classification schemes, early detection methods, intelligent data rejection/validation criteria, heuristic flagging of model results, among other techniques. With incidents such as *The Carrington event* [2] being a possibility, being able to predict and understand these is of vital importance to humankind given the repercussions these might have on most of the used technology as mentioned before.

[1] NASA's Parker Solar Probe is the first-ever mission to touch the atmosphere of the Sun, equaling a distance of 4 million miles from its surface. The Probe was launched on August 12, 2018. link

[2] Solar Orbiter's mission goals are to take the closest-ever images of the Sun, the first-ever close-up images of the Sun's polar regions, measuring the composition of the solar wind and linking it to its area of origin on the Sun's surface. It was launched in February 2020. link

[3] BepiColombo is Europe's first mission to Mercury which was launched on October 20, 2018. The mission shall last seven years. link

1.3 Problem Definition

The solar wind is threaded by magnetic flux tubes that extend from the solar atmosphere to distances beyond the solar system boundary. At 1 astronomical unit[4] (au), there are observable phenomena that depend on the physical properties of the solar corona (hot and dynamic layer of the Sun's atmosphere) and events taking place there. However, despite the need to determine and predict what is measured at 1 au, there is no direct access to the physical parameters causally linked to the solar wind formation and acceleration processes, which are not directly observable. Some existing simulation models try to fill this gap by extrapolating the magnetic field measured at the Sun's surface to the whole corona. These physical models allow the calculation of wind properties measured at 1 au. However, it is observed that their performance (result-wise) increases when a good initial guess is made regarding the initial conditions for the simulation. It should also be noted that these models also take a long time to process the information and develop a solution. The problem we are trying to tackle is obtaining good initial guesses using machine learning to decrease the time taken to come to a feasible solution.

1.4 Goals

We aim to apply machine learning/deep learning techniques to find good initial guesses regarding the initial conditions for a physical simulation model that predicts the solar wind. The goal is to minimize the computation time required to predict a feasible solution.

1.5 Document Structure

This chapter, chapter 1, served the purpose of contextualizing, motivating and describing the problems this dissertation intends to solve. This document is comprised of six more chapters, structured as follows:

- Chapter 2, Background, introduces the background information and explanation about concepts necessary to contextualize this dissertation.
- Chapter 3, State of the Art, describes the state of the art regarding this dissertation's scope, including the exploration of solution of Machine Learning applied to solar events forecasts and simulations of initial conditions using machine learning and for solar wind and magnetic field prediction.
- Chapter 4, Research Statement, presents the problem this dissertation aims to solve, as well as the approach taken to solve it.
- Chapter 5, Exploratory Data Analysis, explores the available data and studies correlation between features.

[4] An astronomical unit (that can be abbreviated as AU or au) is a unit of length equaling the average distance between Earth and the Sun, which has been measured at 149597870.7 km

- Chapter 6, Experiments and Results, presents the conducted experiments and results obtained while attempting to answer this thesis research questions. Implementation details are also given in this chapter as a way to guarantee this work's reproducibility.
- Chapter 7, Conclusions and Future Work, presents the final conclusions of the dissertation and work to be built upon them.

Chapter 2

Background

This chapter describes the necessary foundations regarding the context of Solar Magnetohydrodynamics. Section 2.1 describes the background of Space Weather. In subsection 2.1.1 a brief description of the Sun is provided, and in subsection 2.1.2 a more thorough look is taken at the solar magnetosphere, including definitions of important phenomena later used for this dissertation. Section 2.2 explains what Machine Learning is and introduces different algorithms and techniques as well as regression metrics and re-sampling techniques. Lastly in Section 2.3 summarizes this chapter.

2.1 Space Weather

Space weather is commonly defined as:

The term space weather generally refers to conditions on the sun, in the solar wind, and within Earth's magnetosphere[1]*, ionosphere*[2] *and thermosphere*[3] *that can influence the performance and reliability of space-borne and ground-based technological systems and endanger human life or health*[3].

In particular, activity occurring on the Sun, like solar flares and coronal mass ejections, as well as the solar wind, can trigger geomagnetic storms and other phenomena on Earth.

[1] A magnetosphere is the region around a planet or star dominated by the planet's or star's magnetic field.

[2] Earth's ionosphere overlaps the top of the atmosphere and the very beginning of space and has direct interference from the Sun.

[3] The thermosphere is directly above the mesosphere and below the exosphere extending from about 90 km to between 500 and 1,000 km above the Earth.

2.1.1 The Sun and its Magnetic Field

The Sun is a massive ball of gas held together by pure gravity force. Comprised mainly of Helium and Hydrogen, it provides fundamental conditions for life on Earth. However, several solar events such as *The Carrington Event* [2] have proven that the Sun's impact on Earth can be severe and, as such, the study of this stellar object has become more critical. These solar events, correlated with the Sun's magnetic field, are of different dimensions, intensities, and structures. This section intends to serve as a brief introduction to these topics. It is divided into overviews of the Sun's composition, Sun's Magnetic Field, major solar events, and finally, it contextualizes magnetic tubes, which will be studied in this work.

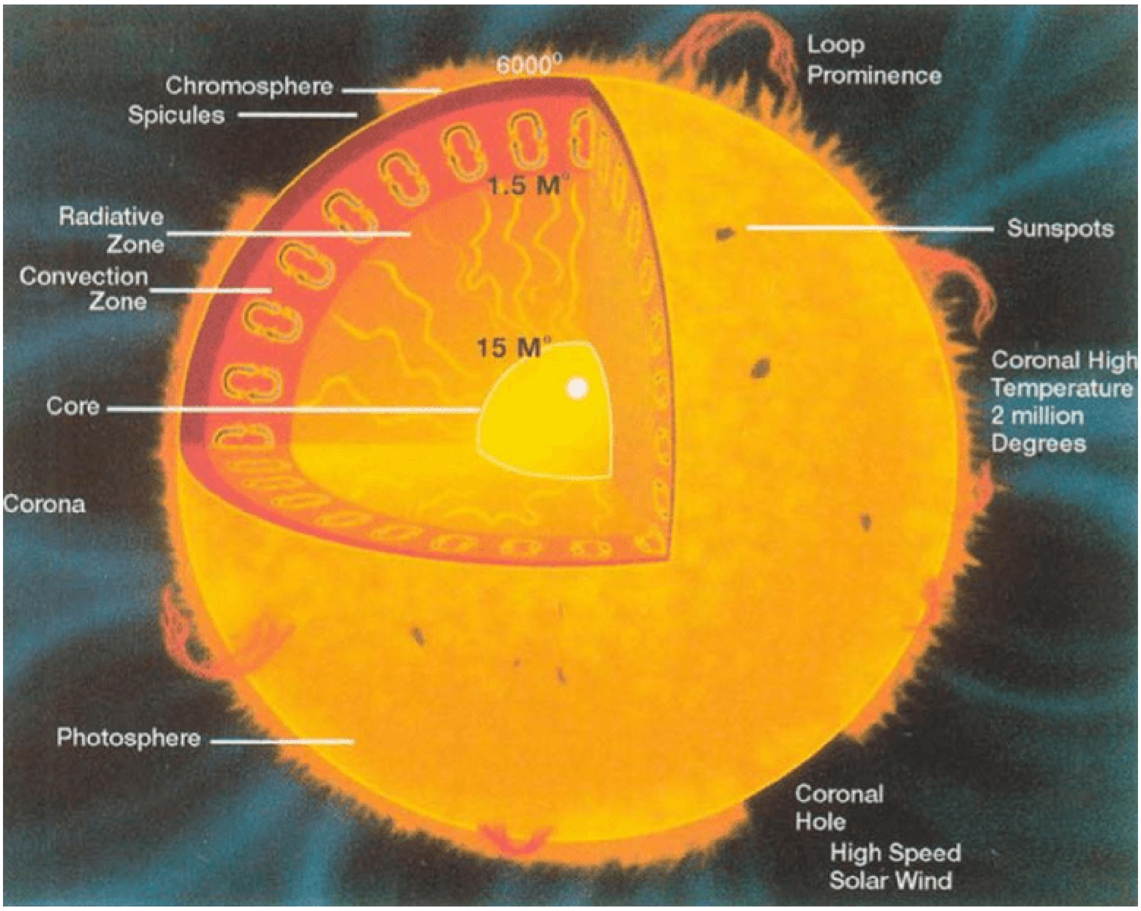


Figure 2.1: Structure of the sun. Temperatures in Kelvin and densities in kgm^{-3} (reprinted from [1]).

The Sun, as can be seen in Figure 2.1 is comprised of a core, where temperatures can reach billions of degrees, the radiative zone, in which energy is mainly transported toward the exterior employing radiative diffusion and thermal conduction, a convection zone, a photosphere, a chromosphere, and a corona.

The magnetic field of the Sun acts in different ways at each of the layers, and the aggregation of these behaviors eventually leads to the observable solar events referred to before.

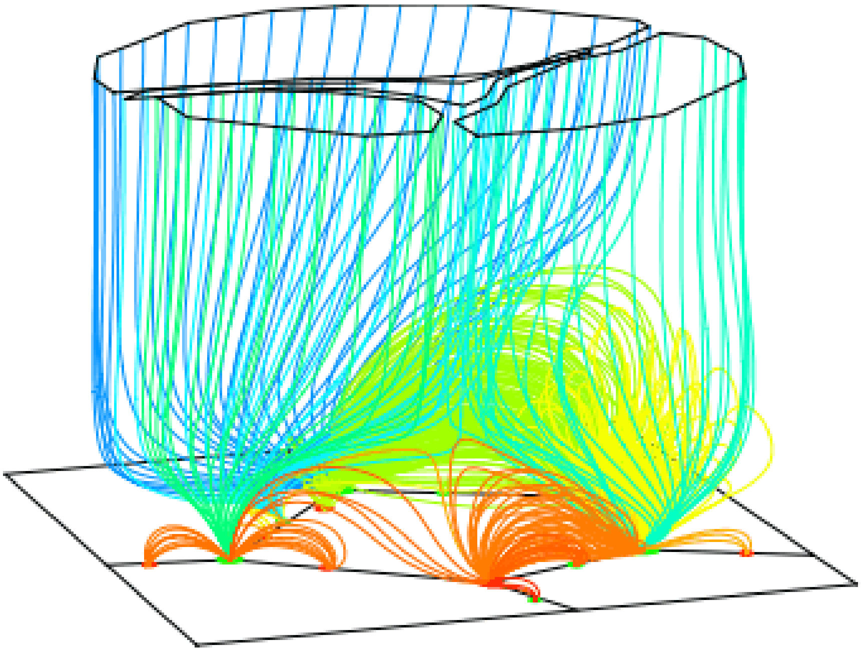


Figure 2.2: Magnetic field above the chromosphere (reprinted from [4]).

The core of the Sun is a so-called nuclear reactor generating an extraordinary amount of energy. This energy leaks continuously outwards, across, and through the radiative zone by radiative diffusion and thermal conduction. One should note that this is a considerably slow process due to this duality in emission and absorption, taking the photons a considerable amount of time to cross the radiative zone.

In the convection zone, as the name implies, the energy is primarily transported by convection. Such happens due to the abrupt fall in the temperature gradient from the interior to the outermost parts of the Sun. As such, the hot plasma rises and partitions before cooling, falling, and rising again. There is a sheer layer at the lower part of this zone where the Sun's magnetic field is believed to be generated by a dynamo.

The photosphere is the lowest part of what is considered the Sun's atmosphere and emits most of the solar radiation. Its optical thickness (τ), in the near-ultraviolet is defined by the equation $I = I_0\varepsilon^{-\tau}$ where I_0 is the radiation intensity at the source, and I is the observed intensity after a given path. Two types of granulation dominate convection in this layer. The granulation comprises individual granules of the order of 1 000 km, while the super-granulation cells are over 20 000 km. The atmosphere experiences magnetic flux emergence that can lead to solar flares, which can be intense (later explained in this section).

The chromosphere is the second of the three main layers in the Sun's atmosphere, and its density is four orders of magnitude lower than that of the photosphere. The magnetic flux that emerges from the photosphere is highly concentrated in small flux elements.

The magnetic flux which emerges from the photosphere is not distributed smoothly over the solar surface but highly concentrated in small flux elements. The upper photosphere and chromosphere form a relatively cool layer up to a height of several thousands of kilometers (km) above the visible surface where the plasma density and pressure decrease rapidly, the flux elements widen their horizontal cross-section, and their total pressure balances the one of the surrounding plasma. Eventually, individual flux tubes are bound to merge with a flux of equal polarity or bend into magnetic arches to connect to flux elements of opposite polarity, as can be seen in Figure 2.2. This phenomenon creates magnetic flux loops, which will be later discussed.

Finally, the corona stands as the outermost part of the Sun's atmosphere. It is dominated by the magnetic field and is most easily seen during a total solar eclipse. The coronal magnetic fields vary between a few G[4] to many hundreds of G. It is comprised of three main parts being these dark coronal holes (in these plasma escapes outwards and results in fast solar winds), bright coronal loops, which are magnetically closed and connect photospheric regions of opposite polarity; and finally, small intense features called X-ray bright points consisting of tiny loops [1, 4]. Modeling of the corona is usually based on magneto-hydrodynamics (MHD), which describes a strongly ionized and magnetized plasma behavior such as that found in the corona.

2.1.2 Solar Events

Solar events such as sunspots (2.1.2.1), coronal mass ejections (2.1.2.2), solar flares (2.1.2.3) and solar wind (2.1.2.4) are phenomena that occur in the Sun's magnetic atmosphere. In this chapter these events are explained.

2.1.2.1 Sunspots

Sunspots are dark cold spots in the sun's surface caused by an extreme concentration of magnetic flux that inhibits convection.

This concentration is caused by intense flux tubes of up to 2 kG that due to convective compression, flux expulsion and evacuation become more vertical and compressed thus achieving higher magnetic intensity.

The sunspots are composed of two main parts: the umbra and the penumbra. In the umbra (darkest part) the magnetic field is almost perpendicular to the surface of the Sun. In the penumbra the magnetic field is not so inclined.

The life-cycle of a sunspot is comprised of two stages. In the first stage, the so called formation, a high intensity magnetic flux tube rises buoyantly, being strong enough to inhibit convection leading to cooling and falling of the plasma in such tube. As such, the magnetic field in the spot increases up until it becomes unbalanced with the external gas pressure. In the second stage, the stage of decay, magnetic pressure removes field concentrations, leading to the dispersion of sunspots. Sunspot cycles tend to accompany the solar eleven-year cycle [1, 4].

[4] The gauss, is a unit of measurement of magnetic induction, measuring the magnetic flux density.

2.1.2.2 Coronal Mass Ejections

Coronal Mass Ejections, also known as CMEs, are magnetic plasma releases of up to 1022 Mx and up to 4052kg, expanding from the Sun's surface, having a long range of reach. This range goes from near corona projections to farther into the planetary system. They often appear in active regions where one can find sunspots and are closely related to solar flares. Their occurrence rate varies from 0.2 per day while at solar minimum to 6 a day at solar maximum.

CMEs are believed to be caused by magnetic reconnection, a phenomenon in which magnetic energy is converted into kinetic and thermal energy. It corresponds to an impulsive release of energy accumulated progressively in the magnetic field.

Most ejections originate from active regions on the Sun's surface, such as groupings of sunspots and are frequently associated with solar flares [1, 4].

2.1.2.3 Solar Flares

Solar flares are bright eruptions of light in the surface of the Sun and just like CMEs, are usually observed near a sunspot. These bright phenomena can be categorized from A-C, M and X classes according to the X-ray flux felt near Earth. Each class corresponds to a peak flux 10 times higher than that of the previous one, and can be further subdivided from 1 to 9.

Just like CMEs, there is some evidence that solar flares can be caused by magnetic reconnection. As such the plasma is increasingly heated and its particles accelerated leading to the release of radiation across the electromagnetic spectrum.

One of the possible impacts of solar flares lies in the possibility of it affecting Earth's ionosphere causing possible disruption in radio communications, radars and other devices [1, 4].

2.1.2.4 Solar Wind

Solar wind, a flow of charged particles, originates at the outermost layer of the Sun, the corona. In here, there is a continuous outward expansion carried forward with the solar wind.

Table 2.1: Comparison of fast and slow winds (based on [1]).

Type of wind	Speed	Electron Density	Mass Loss	Ram Pressure
Slow	$400kms^{-1}$	$7x10^6m^{-3}$	$1.5x10^9kgs^{-1}$	$2.1X10^{-9}Pa$
Fast	$750kms^{-1}$	$2.5x10^6m^{-3}$	10^9kgs^{-1}	$2.1X10^{-9}Pa$

Most of the solar wind plasma comes from thin intense flux tubes described in 2.1.2.2 which appear at supergranule as well as granule boundaries. It should also be noted that most of coronal holes' loss of energy is in fact due to the winds.

There are two fundamental states of solar wind being these fast and slow. The main differences at 1 AU can be seen in Table 2.1.

Solar winds can cause disturbances in Earths' magnetic field in the form of geomagnetic storms [1, 4].

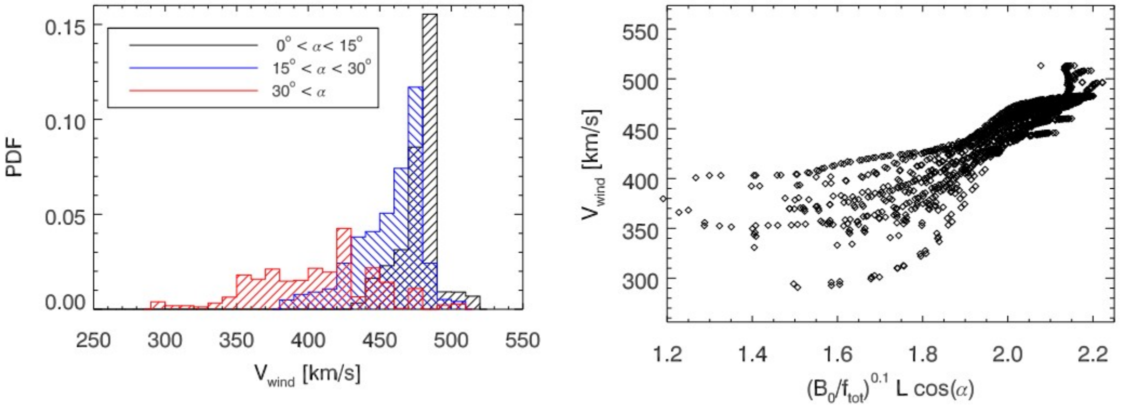


Figure 2.3: Left: histograms of the terminal wind speed for different intervals of maximum field-line inclination. Right: Terminal wind speed vs a function of the absolute magnetic field amplitude and the total expansion factor (reprinted from [5]).

Changes in the magnetic field during the solar activity cycle cause variations in the solar wind speeds at 1 au. Evident changes are seen in the solar wind at solar minima versus solar maxima, making it clear that the spatial distribution of both slow and fast wind is directly impacted by the coronal magnetic field's cyclic variations. These cyclic variations are themselves closely linked to the 11-year cycle of the Sun's activity. Thus, solar wind speeds appear to be determined by the magnetic flux-tubes' geometrical properties through which it flows. The total expansion factors f_{tot} and absolute magnetic field amplitudes B_0 are given by $f_{tot} = \frac{B_0}{B_1}\left(\frac{r_0}{r_1}\right)^2$ where B_0 and B_1 are evaluated respectively at the surface ($r_0 -> r = 0$) and at the outer boundary of the domain ($r_1 -> r = 15$ solar radius). These variables are proven to have a direct impact on wind speed along with the flux-tube inclination.

These relations can be better observed in Figure 2.3 where L is the field-line length and α is the maximum inclination of each field line regarding vertical direction [5].

2.2 Machine Learning

Machine learning is often described as the art or science of making computers modify or adapt their actions to make them more accurate. This accuracy is measured by how well the actions chosen by a computer reflect the correct ones.

2.2.1 Machine Learning Algorithms

There are four main types of learning algorithms: supervised, unsupervised, reinforcement and evolutionary learning. The main difference between supervised and unsupervised learning consists of having a training set of examples with the correct answers on the first one but no on the second one. Reinforcement learning consists of telling the algorithm if the answer is wrong but not providing the correct ones, and finally, evolutionary learning bases itself in biological organisms to learn [6].

There exist several machine learning algorithms to be considered when dealing with any problem resolution.

Some of the approaches considered are: Bayesian analysis, clustering, decision trees, deep learning, dimensionality reduction, regression, ensemble, instance-based, neural networks, regularization, and rule systems. Each approach is further explained in its corresponding subsection.

2.2.1.1 Statistical Learning Algorithms

Finding a predictive function based on data is the aim of Statistical learning theory, and it has a big abundance of applications in the realm of AI. The major aim of statistical learning algorithms is to provide a framework for examining the inference problem: obtaining knowledge, making predictions, and making decisions by creating a model from a data set [7].

Bayesian Bayesian inference is described as suiting a probability model to a data set and summing the result by a probability distribution on the model's parameters and unobserved quantities such as predictions for new observations. The three main steps of Bayesian data analysis consist in arranging a complete probability model, conditioning on observed data, and finally assessing the fit of the model and the implications of the resulting posterior distribution. The main motivation for using Bayesian approaches is the facilitation of a common-sense interpretation of statistical conclusions [8].

Regression Regression is a method of obtaining mathematical relationships between variables. Such requires the opening premise that a specific type of association, linear in unfamiliar parameters is in place. The undiscovered parameters are estimated following certain additional premises by using available data, and finally, a suited equation is achieved [9].

2.2.1.2 Instance Based Learning

Instance-based learning can be described as algorithms that drag the generalization process until classification is accomplished. These algorithms demand limited computational time during the training period as opposed to other eager-learning algorithms but need longer computation time throughout the classification process [10].

Clustering One of the most critical data analysis ventures is to classify or group data toward a collection of classes or clusters. Data targets that are grouped ought to exhibit alike features based upon some guidelines. Clustering is an unsupervised classification technique since no labelled data are available and, as such, has the goal of separating a measurable, unlabeled data set into a measurable and discrete set of masked data structures in lieu of providing an accurate characterisation of unobserved samples generated from the same probability distribution [11].

2.2.1.3 Logic based Algorithms

Logic-based algorithms, such as decision trees make use of probabilistic reasoning and deductive reasoning solving probabilistic satisfiability based on the study of the logical properties of the problem to solve [12].

Decision Trees Sami K Solanki *et al.* . describe decision trees as a classifier represented as a recursive partition regarding the instance space [4]. The decision tree comprises nodes that create a rooted tree, a directed tree with a rooted node with no incoming edges. Any other nodes have precisely one incoming edge [13]. There are two main stages concerning the DT induction method: the growth and the pruning stage. The growth phase entails a recursive partitioning regarding the training data occurring in a DT such that either each leaf node is associated with a single class or additional partitioning of the given leaf would occur in at least its child nodes being under a stipulated threshold. The pruning phase intends to generalise the DT made in the growth phase by creating a sub-tree that bypasses over-fitting the training data. In each iteration, the algorithm considers the partition of the training set utilising a discrete function of the input properties. The election of the most suitable function is performed according to some splitting criteria. When picking a good split, each node farther divides the training set into more diminutive subsets, until no division obtains an adequate splitting measure or a stopping rule is satisfied [14].

2.2.1.4 Deep Learning

Deep-learning techniques are representation-learning methods with various representation levels, achieved by making manageable but non-linear modules that convert the description at one level into a greater, somewhat more conceptual level. With the combination of fairly before-mentioned conversions, rather complex functions can be learned. Higher tiers of representation augment the important input features for differentiation and repress unnecessary variations for classification tasks [15].

Neural Networks Artificial neural networks are inspired by the structure and performance of the human's biological neural network. It is made up of neurons that consist of processing units which receive an input and the associated weights, a summing part, an activation value, an output function and finally an output signal. These neurons are interconnected according to some topology to achieve a pattern identification task. Each unit of an ANN accepts inputs from other associated units and an external origin in operation. A weighted sum of inputs is calculated at a given instant of time. Usually, the activation dynamics is done to recall a pattern saved in the network. There are several options available for both activation and synaptic dynamics, in an implementation [16].

2.2.1.5 Support Vector Machines

Support Vector Machines are the realisation of mapping $x \in \mathbb{R}^n$ into a high dimensional space and construct an optimal hyper-plane in this space. The mapping is performed by a Kernel function which defines an inner product in $\mathbb{H}$. The optimal hyperplane is then the one with the maximal distance in $\mathbb{H}$ space to the closest image from the training data [17].

2.2.2 Validation in Regression Machine Learning Models

In this section, some background concerning validation of results when using and developing regression machine learning models will be given. The section is split into regression Metrics for Machine Learning (2.2.2.1) and Cross Validation (2.2.2.2).

2.2.2.1 Regression Metrics For Machine Learning

To evaluate if the used/developed model is a good fit for our case and dataset, we must measure the quality of the fit. In other words, this means measuring the difference between the predicted value and the true value.

In a regression setting, the most commonly-used measure is the mean squared error (MSE), given by

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{f}\left(x_i\right)\right) \tag{2.1}$$

where $\hat{f}\left(x_i\right)$ is the prediction that $\hat{f}$ gives for the ith observation.

The smaller the MSE, the smaller the differences between the expected and predicted values.

In order to minimize the MSE described in equation 2.1 there are a series of techniques that can be successfully used to estimate this minimum. One of the most common to be used is cross validation which is discussed in section 2.2.2.2.

2.2.2.2 Re-sampling Techniques

Single hold-out random subsampling When the dataset is considered to be representative of the whole possible population, one of the simplest methods to use as a resampling technique is to simply randomly split the data into a training and test data split. Usually the test set varies between 10% and 30% of the cases and the rest goes to the training data fed to the model. However, seldomly does the obtained data is considered to be fully representative and as such more complex methods are utilized.

k-fold random subsampling In this method, the method described above is repeated k times. This means that k pairs of training and testing splits are generated. In the end, the used MSE is the average over all the k test sets.

k-fold cross-validation Cross-validation is a data re-sampling method used in machine learning to assess the generalization ability of predictive models and prevent over-fitting [18]. As such one can use an limited sample in order to estimate how the model is expected to perform when predicting data not used in its training stage. Cross validation is very similar to the k-fold random subsampling method. However, the sampling is done in such a way that there are no overlaps among the tests.

Algorithm 1: k-fold cross-validation

```
Require: $K_n$, where $n$ is the number of folds and $K_n$ represents each fold
Require: $D$ dataset containing input features $x$ and output features $y$
Require: $P_{sets}$ , set of hyperparameters with different values
Require: $M$ a single model
foreach $i = 1$ to $K_n$ folds do
    Split $D$ into $D_i^{train}$ and $D_i^{test}$ for the $i'th$' split
    foreach $p$ in $P_{sets}$ do
        Train $M$ on $D_i^{train}$ with hyperparameter set p
        Compute test error $E_i^{test}$ for M with $D_i^{test}$
    end
end
```

The algorithm for the k-fold cross validation technique can be seen in Algorithm 1. The MSE computed and used for the procedure done is the mean of the models skill scores, that is, the MSE values of each. Another important aspect to take into consideration is the choice of k. K sets the difference in size between the training and testing sets. If k increases then the difference decreases and the overfitting odds of the algorithm become slimmer. In known bibliography, the choice of k is 5 or 10, with authors never setting for each choice. The value of 10 is commonly used in machine learning applications though and such choice is shown to yield test error rate estimates that suffer neither from excessively high bias nor from very high variance [19] [20].

2.3 Summary

This chapter introduced Space Weather, Machine Learning and gave some mathematical background which are all fundamental parts to the understanding of this dissertation. Section 2.1 introduced the concept of space weather explaining some properties of the Sun and its magnetic fields as well as details of important solar events. Section 2.2 introduced fundamental concepts of machine learning and common techniques whilst also presenting a few validation concepts such as the mean squared error.

Chapter 3

State of the Art

In this chapter, a review of state of the art is made throughout sections 3.1, 3.2 and 3.3. In section 3.1 an overview of ML techniques applied to solar weather are explored. In section 3.2 peer work on the area of improvement of initial conditions in physical simulations using ML is explored. Finally in section 3.3 a literature review of the impact initial conditions have in physical simulations is done.

3.1 Machine Learning Approaches to Solar Weather

In this section, the different applications to Solar weather of the approaches presented in Section 2.2 are described. To do so, an ad-hoc search in google scholar[1] was made throughout the use of combinations pertaining the words and expressions "machine learning", "solar weather", and words related to the different solar weather phenomena identified in 2.1.2 and the methods described in 2.2 were used. In the end, an analytical summary of the found literature is made, plotting the different used techniques per year and per problem.

3.1.1 Statistical Learning Algorithms

In this subsection we discuss the algorithms described in 2.2.1.1. As stated before, statistical learning theory has the main goal of finding a predictive function based on data. Bellow we discuss Bayesian and Regression methods applied to solar weather problems.

[1] https://scholar.google.com/

3.1.1.1 Bayesian

The use of a Bayesian method to predict solar flares is made in the work of MS Wheatland [21]. This prediction is made using the flaring history of an active region and phenomenological rules of flare statistics. Such is done to refine a primary prediction for a big flare during the succeeding time. To calculate ε, the probability of a big event other variables must be calculated first. Firstly the estimation of λ_1, the distribution of the rate of flares larger than S1, using a Poisson model, and γ (gamma is a constant that depends on the choice of the quantity S, but typically is found to be in the range 1.5-2)(using the estimation of a power-law index [22]) is made. Secondly, the probability distribution $P_2(\lambda_2)$ for the rate λ_2 of flares larger than S2 is made using a combination of $P_1(\lambda_1)$ and $P_\gamma(\gamma)$ (estimated as in T Bai's work [22])). Finally, $P_\varepsilon(\varepsilon)$ can be constructed using $P_2(\lambda_2)$ by performing a change of variable. Using the first approach, the method can provide a reasonable estimate of a big event's probability. Using the second approach, the authors conclude that a reasonable estimate for υ is obtained for a relatively small number of events.

Another example of a Bayesian approach is described in Neal *et al.* [23]. The authors use Bayesian inference techniques, and Markov Chain Monte Carlo sampling approaches to predict dose-time profiles for vigorous solar particle events. Dose and dose-rate measurements obtained earlier in the event are used as inputs. To find links amidst similar solar particle events, surrogate dose values are arranged in hierarchical models which consider nonlinear, sigmoidal growth for dose during an event. Later, Markov Chain Monte Carlo methods are used to sample from Bayesian posterior predictive distributions for dose and dose rate. It is concluded that the used Bayesian forecasting models presented provide reasonable predictions of dose and dose–rate time profiles with both for the used events of November 8, 2000, and August 12, 1989.

In Jonas *et al.* , a unified framework approach is used to visualise space weather event probability using a Bayesian model average and extreme historical events. This Bayesian model average (BMA) combines the three most used statistical methods in recent literature to describe the probability of extreme space weather events: Power Law Distribution, Extreme Values Theory, and the Lognormal Distribution. These three models fit the same data under a Bayesian paradigm while the events are considered independent. Later, a Poisson distribution is used to infer the probability of events greater than x_crit(the critical value of Dst considered) in a period of duration Δt. To determine the resulting probability across D_st (disturbance storm time) values for x_crit, a weighted average over the three models is calculated, weighing them by the goodness of fit. The framework ultimately allows the user to tailor the time range to whichever period is of interest helping to display different D_st values for different time frames and is shown to work according to the expected values in previous works [24].

In an attempt to generalise the Bayesian methodology previously used to predict single-event solar particle events, Neal *et al.* bind particle flux and fluence data with dose rate and dose calculations to develop a standard for defining an event as multiple. The model was proven to improve previous forecasts [25].

A Bayesian approach to solar flare prediction was generated in the work of MS Wheatland using solely the statistics of previously observed solar flare events. It is assumed that the size distribution of flares follows a power-law distribution and that, on short time scales, flares happen as a Poisson process in time. Assuming that the power-law index γ plus the current rate of small events λ_1 are both estimated, then ε is the probability that the authors are trying to find. Using Bayesian generalisation, and a series of M events with sizes s_x observed from $t = 0$ to $t = T$, the usage of the Bayesian blocks procedure is conducted to determine this time interval. Shortly, this method comprises Bayesian hypothesis experimentation, which compares an individual rate Poisson model with a dual-rate Poisson model for the data. If the dual-rate model is more suitable, the data section is segmented, and the two segments are themselves subject to the test. And so, the posterior distribution for ε can be found as:

$$P(\varepsilon) = C[-\ln(1-\varepsilon)]^{M'}(1-\varepsilon)^{(T'/\Delta T)(S_2/S_1)^{y^*-1}-1} \times \wedge[-\frac{\ln(1-\varepsilon)}{\Delta T}(\frac{S_2}{S_1})^{y^*-1}]$$

. The paper concludes with the skill scores for M and X (events with peak fluxes larger than 10^5 and $10^4 Wm^2$, respectively, in the 1–8 Å band witnessed by the satellites) event prediction of the model which are, respectively, 0.272 and 0.066. The event statistics method is seen to out-perform the NOAA (US National Oceanic and Atmospheric Administration) method in foretelling overall numbers concerning M-X and X event days [26].

3.1.1.2 Regression

A.S. Parnowski [27] proposes a regression modelling approach to space weather forecast. Such a model allows forecasting D_{st} index as far as 6 hours before with nearly 90% correlation. It may additionally be utilised for creating phenomenological models of synergy amid solar wind and magnetosphere. Two novel geoeffective parameters were obtained with its help: the solar wind's latitudinal and longitudinal flow angles. It was attested that D_{st} index retains its previous 2000 hours. The introduced approach is statistical yet holds some empirical principles based on regression analysis and mathematical statistics. In its framework the prognosticated D_{st} value is sought in the form $D_{st}(j+k) = \sum_i Ci.xi(j), (1)$ where j is the number of current steps (number of hours since January 1, 1963), k is the prediction length, C_i are the regression coefficients, and x_i are the regressors, which denote functions and combinations of input amounts, already measured at the time while the prediction is made. Values of C_i are defined by a least square method (LSM) across a broad sample of solar wind and geomagnetic data, with corresponding statistical weights of all points. The irrelevant parameters are then discarded, and the method is iterated till the regression carries only meaningful regressors. The authors used the OMNI2 (2009) database together with provisional D_{st} data, taken from Kyoto WDC (World Data Center for Geomagnetism). Hence a consecutive 44-year D_{st} time series was collected being fitting parameters not used. The suggested regression approach seemed to be fit concerning space weather forecasting problems. For the forecasting alone, its significant benefits are quite good correlation (about 90% for 6 hours forecast), adaptability to any samples, and swift forecasting code.

N. Srivastava [28] executes a logistic regression model for forecasting appearance of intense/super-intense geomagnetic storms. A twofold dependent variable, showing the existence of intense/super-intense geomagnetic storms, is regressed facing a set of independent model variables that describe several solar plus interplanetary features of geoeffective CMEs. The regression coefficients are estimated from a training data set obtained from 64 geoeffective CMEs seen between 1996 and 2002. The trained model is verified by forecasting geomagnetic storms from a validation dataset, similarly obtained of the same data set but not used for training the model. The model predicts 78% of the geomagnetic storms of the validation data set. Besides, it predicts 85% of these geomagnetic disturbances from the training data set. Those outcomes show that logistic regression models can be efficiently adopted for predicting the appearance of intense geomagnetic storms from a set of solar and interplanetary circumstances. The logistic regression model is a generalised linear model that extends the linear regression model by inking the range of real numbers to the 0–1 range. To implement logistic regression, the authors used the same dataset used by other authors. The logistic regression was trained on the training dataset using XLSTAT software. Such training included estimating the regression coefficient applying an iterative maximum likelihood method. The outcomes reveal that the model accurately classifies 62.5% of the training super-intense geomagnetic storms and 97% of the training intense geomagnetic storms. The model accurately classifies 50% of the super-intense geomagnetic storms and 100% of the intense geomagnetic storms among the validation events.

A dynamic multiple regression model aiming to forecast the diurnal peak of unusual energy electron fluxes at geosynchronous orbit from input data is deduced in Wei *et al.* [29]. The model takes as input variables the upstream solar wind speed v, the solar wind dynamic pressure P_{dyn}, the half-wave rectifier function vB_s, the asymmetric disturbance index in the horizontal direction $AsyH$ plus the symmetric disturbance index in the horizontal direction $SymH$. The model's output variable is the logarithm of relativistic electron flux maxima $(> 2MeV)$. To deduce the prediction model, the NARX model (Nonlinear AutoRegressive Moving Average with eXogenous inputs) [30, 31, 32]. Applying the NARX model's input variables, two multiple linear regression models, one with lags $m = 1$, and other with lags $m = 3$ were estimated. All of the concluding predicted models hold 30 significant model terms chosen one by one in order of value. Following [33], a measure designated the prediction efficiency defined as $PE = 1 - MSE(error)/var(output)$ was employed to measure the model performance. The outcomes display that this model execution of the identified multivariate and multi-rate dynamical regression models is better than the one produced by the non-parametric models as presented in [33].

3.1.2 Instance Based Learning

In this subsection we discuss the algorithms described in subsubsection 2.2.1.2. As stated in the reffed subsection, instance-based learning can be described as algorithms that drag the generalization process until classification is accomplished. In this subsection we discuss peer work on clustering applied to space weather problems.

3.1.2.1 Clustering

Unsupervised clustering methods and learning vector quantity (LVQ) are applied in Rong Li *et al.* to predict solar flares [34]. As measurements, three magnetic parameters, being these the neutral line's length, the number of singular points and the maximum horizontal gradient extracted from SOHO/MDI longitudinal magnetograms, are applied. As a way of forming sequential data, the sliding-window technique is employed. The method used in this work has two steps: the first is the K-means clustering method that converts the non-balanced training set in balanced ones; the following is the learning vector quantity(LVQ) method. The number of groups (k clusters) is fixed to the corresponding number of the flaring samples during the K-means approach. The outputs of the clustering algorithm are clustering centres, and every one of these clustering sections. The clustering cores samples are selected to construct a well-balanced training set by adding the flaring sample part. Then, the balanced training set is fed toward a training vector quantity network. The learning vector quantity network output is the group label of a trial sample and is applied to forecast the flares' level inside 48 hours. Five measurements are utilised to evaluate the performances of the proposed approach: True Positive (TP) rate, True Negative (TN) rate, False Positive (FP) rate, False Negative (FN) rate and accuracy. Preliminary outcomes show that this introduced model's performance with sequential data is enhanced than if solely using the LVQ method.

To separate active regions (ARs) that are quiet from possibly eruptive ones, the authors' in Moon *et al.* [35] use an innovative clustering of ARs through matrix factorisation. A new clustering of moving regions based on the local geometry observed in Line of Sight magnetogram and continuum images is introduced. Using $m \times n$ patches $Z \approx AH$ where Z is the $2m^2 \times n$ data matrix with n data points being regarded. As such, matrix factorisation of image patches is an encouraging novel means of defining active regions. Some recommendations for metrics, matrix factorisation techniques, and regions of interest to study active regions are provided. It is found that these metrics produce organic clusterings of active regions. The clusterings are associated with grand scale descriptors of an active region, such as its size, local magnetic field configuration, and complexity as measured with the Mount Wilson classification system. It was further observed that including data centred on an active region's neutral line can increase correspondence among the clustering outcomes and separate active region descriptors such as the Mount Wilson classifications and the R-value. Finally, the used clustering method is the Evidence Accumulating Clustering with Dual Rooted Prim tree Cuts (EAC-DC) method which groups the data by setting a metric based on the increase of two minimum spanning trees grown sequentially of a couple of points. The Hellinger metric is utilised to grow the spanning trees by feeding it into a spectral clustering algorithm that gathers related inputs. The conclusions that are withdrawn from the paper are the following. When analysing and clustering the ARs built on the local properties' global statistics, there are similarities to the classification based on the large scale characteristics. For example, when clustering using the Hellinger distance, one cluster contained most of the complex ARs. As before-mentioned, matrix factorisation of image patches is an assuring novel approach regarding characterising active regions.

3.1.3 Logic-Based Algorithms

In this subsection we discuss the algorithms described in subsubsection 2.2.1.3. As stated before, logic-based algorithms, make use of probabilistic reasoning and deductive reasoning solving probabilistic satisfiability based on the study of the logical properties of the problem to solve. In this subsection we resume peer work made in decision trees applied to solar weather problems.

3.1.3.1 Decision Trees

In Engall *et al.* , the authors use decision trees to predict onset and time profiles of solar-driven events, including solar X-ray flares; solar energetic particles (SEP); coronal mass ejections; and high-speed stream. To do so, the Space Radiation Intelligence System (SPRINTS) is presented. The systems' input consists of public forecasts, physics-based models, measurements made of the local plasma environment, and two-dimensional event meta-data (magnetograms and automatic detection). These inputs are then fed into a SPRINTS, and then the output is divided by different deployment models. SPRINTS leverages machine-learning techniques to build and explore these more advanced multivariate forecasts capabilities automatically. With an emphasis on human interpretability and knowledge discovery concerning these models, SPRINTS has currently designed two main machine-learning applications:1. Decision Trees2. K-Nearest Neighbor (KNN) Evaluator. These partitions create a tree-like hierarchy using the most relevant attribute at each split point that best separates the given events by their label. The final bins (leaves) of the tree represent the model's predictions and ideally have high levels of purity, whereby a majority of the events in each bin share the same label. Beyond the benefit of simplicity and understandability, decision trees implicitly produce a ranking of parameter (and specific value) importance which are later grouped into random forests. The parameters from automatic detections used are: flare magnitude, flare-integrated X-ray flux, flare decay phase, flare heliolongitude, flare heliolatitude. The used separation is yes/no forecast. SPRINTS has been demonstrated to provide comparable and novel SEP forecast modelling capabilities relative to others in the industry and academia [36].

3.1.4 Deep Learning

Deep-learning techniques are representation-learning methods with various representation levels,achieved by making manageable but non-linear modules that convert the description at one level into a greater more conceptual level. In this subsection we discuss deep learning techniques such as, but not limited to, neural networks applied to solar weather problems.

In Yi *et al.* , the application of deep learning approaches to the prediction of crucial solar X-ray flare flux profiles is presented. The used data encompasses the Geostationary Operational Environmental Satellite 10 X-ray flux data from 1998 to 2006 [37]. The 10-fold cross-validation plus the RMS error (RMSE) based on flux profiles and RMSE based on its maximum flux are applied to evaluate the models. For judgment, the authors consider two simplistic deep learning models and four traditional regression models. The proposed models make use of LSTM [38] recurrent neural networks. The first proposed model consists of two LSTM layers and one fully connected layer for the encoder and a decoder. The LSTM layer results in the decoder are sent to a fully connected layer without an activation function. The second proposed model has an identical composition without attention. The models used for comparison are the Auto-Regressive Integrated Moving Average [39], the K-Nearest Neighbor Regression [40], the Support Vector Machine Regression [41], and the Random Forest Regression [42]. The main conclusions are taken from the paper are as follows. The suggested models better the other models, the models achieve better performance for projecting X-ray flux characterisations with low-peak fluxes than those with high-peak fluxes, and finally, the models are successful in forecasting flare duration with large correlations for both all events as well as events at peak times.

3.1.4.1 Neural Networks

In Valach *et al.* the authors use supervised Artificial Neural Networks, ANNs, as a way of quantifying the geomagnetic response of selective solar events to conclude if the success of the neural network prediction scheme based solely on the solar disc observations — X-ray flares (XRAs) accompanied by solar radio bursts(RSPs) — can be improved by additional information concerning the energetic solar particle (SEP) flux. For doing so, a three-layer fed forward ANN was used with N input neurons, one layer of hidden neurons and one output neuron to compute the geomagnetic responses using the standard backward propagation algorithm. It is shown that supplying such additional input data enhances the neural network forecasting scheme [43].

In Uwamahoro *et al.* the authors develop a supervised three-layer forward-fed artificial neural network (FFNN) to estimate the probability of occurring geomagnetic storms after halo coronal mass ejection, including related interplanetary events have taken place. NN optimisation consisted of repeatedly training the network by changing the number of iterations and methodically modifying the number of nodes in the hidden layer; the optimum network architecture consisted of five inputs and five hidden nodes. This implementation led to an accuracy of geomagnetic storm prediction of 100% for intense storms and up to 75% for moderate one. The model's estimate of the storm occurrence rate from halo coronal mass ejection is assessed at a probability of 86% [44].

In H Lundstedt's work, two artificial neural networks are used for forecasts of solar-terrestrial effects which encompass geomagnetic provoked currents. These ANNs make use of multi-layer back-propagation. As inputs, 20 items ranging from the number of X-ray flares, Coronal Mass Ejections, CMEs, number of times the LDEs last more than three hours, and disappearing solar filaments during the day, coronal holes as well as proton event flux and geomagnetic activity. All these items come together to form an input pattern for the ANN in which the values for the present day and one, two, three, four and 27 days before being used. The output patterns consisted of values for the quantities of, no storm, minor storm, major storm and severe storm for the day after the present day (one ANN) and the day after that one (another ANN). The succeeding rates were 73% for the first ANN and 68% for the other one [45].

Using solar magnetic field observations (such as highest horizontal gradient, the length of neutral line and the number of singular points), in Wang *et al.* a solar flare forecasting model is proposed backed by a supervised artificial neural network, using back-propagation training.

The developed model tends to overestimate some flares being that it predicted 69% of the flares correctly and 31% were considered incorrect forecasts [46].

In Sudar *et al.* the authors use a neural network in order to predict transit times (TT) of coronal mass ejections (CMEs) from their opening parameters (initial velocity of the CME and central-meridian distance of its associated flare). The used method was a feed-forward algorithm, and the network consisted of 2 inputs (described before), one output (TT) and one hidden layer with three nodes. The developed NN only predicts when particular CMEs reach 1AU from the Sun. The work concludes that the medium error of the NN prediction in contrast to observations is of approximately twelve hours and that the velocity at which acceleration by drag changes to deceleration is roughly 500 km/h [47].

The work of Vandegriff *et al.* is to predict the remaining time until the arrival of interplanetary (IP) shocks at Earth through the use of a recurrent ANN. This ANN uses ten input nodes, one output node, and two hidden layers with four nodes each. The inputs consisted of proton intensity of varying energy levels, anisotropy coefficient, spectral slope and its derivative, intensity at the midpoint and its derivative. The output consisted solely on the remaining time for IP shock to arrive. The results are as follows: for the 24 hours in advance prediction, the uncertainty was 8.9 hours whilst for the 12 hours in advance, it was 4.6 hours [48].

In Vallach *et al.* , a forecasting scheme of geomagnetic activity was developed using an ANN with a backward propagation algorithm. The scheme's main goal was to determine the probability of which solar flares will be followed by a geomagnetic response of a specific intensity. The NN inputs used were four, namely the heliographic latitude and heliographic longitude of the centre of the area on the solar disc, XRA(X-Rays) class and, finally RSP(radio bursts) type. The output quantity was the probability of the XRA event appearing in the given area being geoeffective. Then, a hidden-layer of five neurons is used and as an output one neuron containing the probability of the XRA event appearing in the given area being geoeffective was given. To assure that the results were stable, the authors trained nine independent networks and used the median of the results obtained as the final result. How successful the model was depended both on the solar flare class and on the combination of radio-burst types being that for RSP IV, the success rate was of 58%, and when only RSP II was observed, the forecast was successful only for flares of the X class with a success percentage of 67% [49].

ANNs were used by T. Colak *et al.* as a method for predicting solar flares. Their system consisted of two neural networks in which the first generates the probability that a sunspot region will produce a solar flare in the next 24 hours and the second one is activated When the first NN predicts that a flare is going to occur and determines whether the predicted flare is going to be C, M, and X class flare. Both neural networks take four inputs (sunspot area, modified Zurich class, most extensive spot and sunspot distribution) and one hidden layer whilst the first has one output (Flare: Yes or No) and the second has three outputs nodes being these the probability of a flare being of type C, M or X. The number of nodes in the hidden layer is calculated by training the Neural Networks changing the nodes from 1 to 20 and calculating the Mean Squared Error (MSE) for each outcome. The networks were optimised using ten nodes in the hidden layer for the first one and 12 for the second one [50].

Intending to predict geomagnetic storm using solar wind data, Wu *et al.* [51] used dynamic neural networks. Such ANNs consisted of Elman's recurrent model consisting of a two-layer back-propagation network with feedback connections from the hidden layer to its input (consisting of precise input units and context units) and an activation function of the hyperbolic tangent for the hidden layer and a linear one for the output layer. The authors were able to conclude that the coupling functions that can be considered suited for predicting geomagnetic storms are $P^{1/3}VB_s$, $P^{1/2}VB_s$, V^2B_s, VB_s, VB_z and V^3B_s in order from high to low prediction goodness being that the best PE can reach 78% accuracy and a correlation coefficient p of 0.89 for a prediction of 1 hour and for predictions of 6-8 hours the values go to 67% and 0.77 at best.

In Yang *et al.* [52] ANNs are used to predict solar wind speed at 1AU. These ANNs consist of 3 layers and make use of the Levenberg-Marquardt back-propagation algorithm. In doing so, they were able to achieve an overall correlation coefficient of 0.74 having 68 km/h as a root-mean-square error. Finally, the probability for identifying a high-speed case is 0.68, with a resolute forecasted value of 0.73 and a threat score of 0.55.

A Hybrid Regression-Neural Network (HR-NN) Method for Predicting Solar Activity is described in Okoh *et al.* [53]. This model fuses regression analysis and neural network learning to forecast the SSN (sunspot number). In testing the method, the current solar cycle(SCs) activity is predicted utilising previous solar cycles, and then the same method is used to predict the upcoming solar cycle 25. The Ap index, along with the SNN values, were used as inputs for the model. Since the everyday and monthly averages of the daily SNN are noisy, they must be smoothed, and such was done by using a time series as in Conway [54]. The 13-month running mean is a conventional smoothing centred on the month in the subject and using half-weights for the months at the beginning and end of the series. Using linear regression, it was revealed that there is a direct relationship between the mean-rise rate and the peak value of SSN for each SC 7 to 24. Also, there is a moderate inverse correlation between the rise duration and the fall duration, suggesting that sequences that take longer to rise will take less time to fall, and vice versa. Lastly, it is shown via regression there exists a positive correlation between the mean rise rate and the mean fall rate, implying broadly that cycles with higher rise rates also have higher fall rates, and vice versa. Despite the accomplished correlations found, the authors recognised that given the need for knowing the Ap index, the regression would not aid beyond the accomplished results in characterising an SNN. Hence, a neural network training method was developed. In this method, the time series indicator was used as input for the neural network and the peak of SSN for each cycle, the rise and fall duration for each cycle and an A normalised fractional value. Training of the neural networks was performed employing the Bayesian regularisation back-propagation algorithm [55]. The HR-NN model was used to predict SSN values for the remaining part of SC 24 at an RMSE value of 3.5, and to give indications of the expectations for SC 25. Forecasts by the model show that the total duration of SC 24 is 11.167 years, and the end of the cycle will be in March 2020. Using an estimated precursor Ap index of 5.6 nT for SC 25, the peak SSN, 122.1 of the cycle, occurs in January 2025 with a total duration of 11 years. Additional simulations of the SSNs by modifying the precursor Ap index between 4.6 and 6.6 nT showed that peak SSNs for SC 25 would change by about 11 units for every 1-nT change in the assumed precursor Ap index.

3.1.5 Support Vector Machines

As discussed in Section 2.2.1.5, support vector machines have many applications. In this subsection we discuss the application of such a technique to solar weather problems.

An application of Support Vector Machines(SVMs) can be found in Inceoglu *et al.* [56] with the purpose of forecasting if solar Flares will be connected to CMEs and SEPs. In this work, the authors use SVMs and Multilayer perceptrons (MLPs) to compare both results. Eighteen physical parameters from active regions (ARs) are used as input for the machine learning algorithms. The first tested algorithm is SVMs creating k individual binary classifiers for l number of classes. The mth binary SVM classifier is then trained using the mth class as positive (+1) whereas the remaining $l-1$ number of classes are regarded as negative (-1) examples. It then analyses the data using a hyper-plane separation with the most significant distance between the data classes. The second used algorithm are MLPs which consists of a feed-forward ANN that classifies multidimensional data into l different classes. This can be regarded as a multinomial logistic regression, in which the results from the Neural Network is the following probability that the input data pertains to a distinct class. The estimated posterior probability distribution of an arbitrary categorical variable depends not only on a data object from a chance feature but additionally on the neurons' weights, which are the basic processing units. As a feed-forward network, the MLP delivers the non-linear parameterized mapping from input I to an output that is a continuous function of the input and the weights. To compare the algorithms, the author's define True Skill Statistics (TSS, compares the probability of detection(POD) to the probability of false detection(POFD)) and the Heidke Skill Score (HSS, it is a method of measuring the fractional improvement of the forecast over the random forecast).

In Liu *et al.* [57] SVMs are used for partial/full halo CME Arrival Time Prediction Using Machine learning Algorithms (CAT-PUMA). Previously observed geoeffective partial-/full halo CMEs do such and the predictions made after applying CAT-PUMA to a test set show a mean absolute prediction error of 5.9 hr within the CME arrival time, with 54% of the predictions having absolute errors inferior to 5.9 hr. Compared with other models, the engine has a higher final prediction for 77% of the events investigated.

Jiao *et al.* [58] use SVMs to detect ionospheric scintillation and classify scintillation events based on training data in the frequency domain. The detector input is the signal intensity.

3.1.6 Analysis

In Table 3.1 a resume of the techniques used, mentioned in this review, by problems and year is made.

Table 3.1: List of papers reviewed concerning machine learning techniques applied to solar weather

Paper	Approaches	Technique	Goal	Problem	Year
Wheatland [21]	Statistical Learning Algorithms	Bayesian	Prediction	folar flares	2004
Neal *et al.* [23]	Statistical Learning Algorithms	Bayesian	Prediction	solar particle events	2001
Jonas *et al.* .[24]	Statistical Learning Algorithms	Bayesian	Prediction	solar particle events	2018
Neal *et al.* [25]	Statistical Learning Algorithms	Bayesian	Prediction	solar particle events	2005
Wheatland [26]	Statistical Learning Algorithms	Bayesian	Prediction	solar flares	2005
Parnowski [27]	Statistical Learning Algorithms	Regression	Prediction	geomagnetic storms	2009
Srivastava [28]	Statistical Learning Algorithms	Regression	Prediction	geomagnetic storms	2005
Wei *et al.* [29]	Statistical Learning Algorithms	Regression	Modeling	magnetic field	2011
Li *et al.* [34]	Instance Based Learning	Clustering	Classification	solar flares	2011
Li *et al.* [34]	Deep Learning	Learning vector quantity	Prediction	solar flares	2011
Moon *et al.* [35]	Instance Based Learning	Clustering	Classification	active regions	2016
Engell *et al.* [36]	Logic-based algorithms	Decision Trees	Prediction	coronal mass ejections; Solar Falres;	2017
Yi *et al.* [37]	Deep Learning	Deep learning	Prediction	solar flares	2020
Valach *et al.* [43]	Deep Learning	Neural Networks	Prediction	solar flares; solar radio Bursts	2009
Uwamahoro *et al.* [44]	Deep Learning	Neural Networks	Prediction	coronal mass ejections; Geomagnetic storms	2012
Lundstedt [45]	Deep Learning	Neural Networks	Prediction	geomagnetic storms	1992
Wang *et al.* [46]	Deep Learning	Neural Networks	Prediction	solar flares	2008
Sudar *et al.* [47]	Deep Learning	Neural Networks	Prediction	coronal mass ejections	2015
Vandegriff *et al.* [48]	Deep Learning	Neural Networks	Prediction	geomagnetic storms	2005
Valach *et al.* [49]	Deep Learning	Neural Networks	Modeling	magnetic field	2007
Colak *et al.* [50]	Deep Learning	Neural Networks	Prediction	Solar Flares	2009
Wu *et al.* [51]	Deep Learning	Neural Networks	Prediction	geomagnetic storms	1997
Okoh *et al.* [53]	Deep Learning	Neural Networks	Prediction	sunspots	2018
Yang *et al.* [52]	Deep Learning	Neural Networks	Prediction	solar wind	2018
Inceoglu *et al.* [56]	Support Vector Machines	Support Vector Machines	Prediction	solar flares; coronal mass ejections; sunspots	2018
Liu *et al.* [57]	Support Vector Machines	Support Vector Machines	Prediction	coronal mass ejections	2018
Jiao *et al.* [58]	Support Vector Machines	Support Vector Machines	classification	solar flares	2017

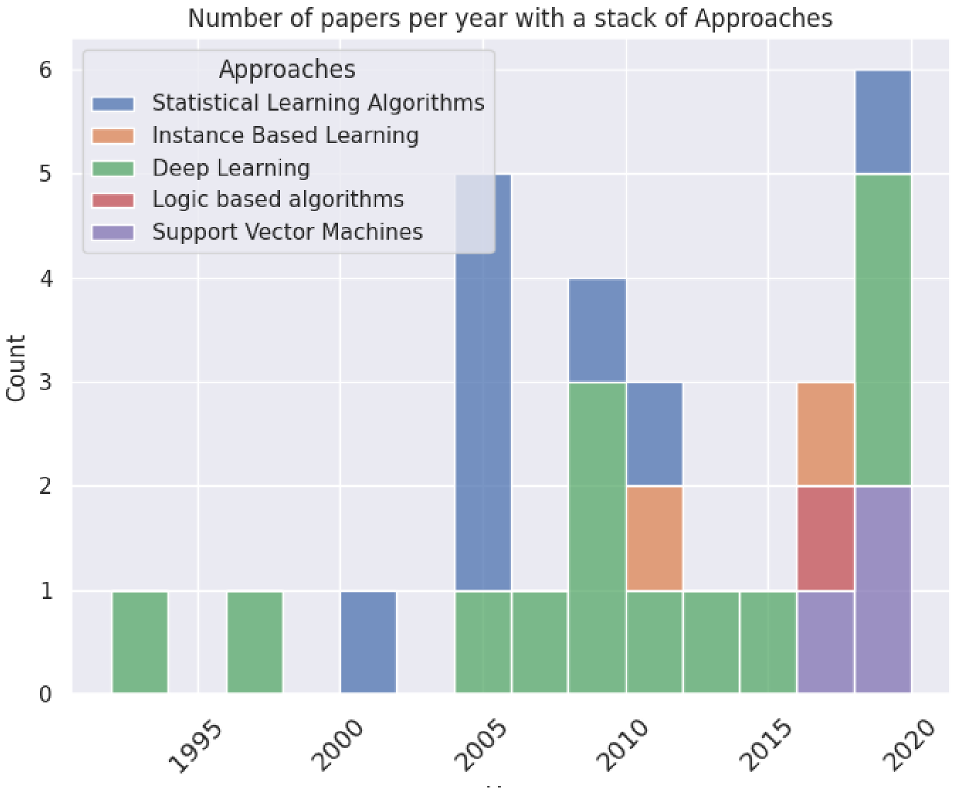


Figure 3.1: Literature Review analysis for Machine Learning techniques over the years stacking the approaches used. The graphic shows a clear increase in usage of these techniques as well as a preference for deep learning solutions

Figure 3.1 shows increasing interest in using machine learning techniques to solve solar weather forecasting problems. Another trend which can be depicted is that increase in the use of deep learning algorithms and a fall in statistical learning algorithms.

Figure 3.2 shows a particular interest has been taken in predicting solar flares followed by geomagnetic storms and CMEs. Such can be explained given the possible impact such events can have on Earth.

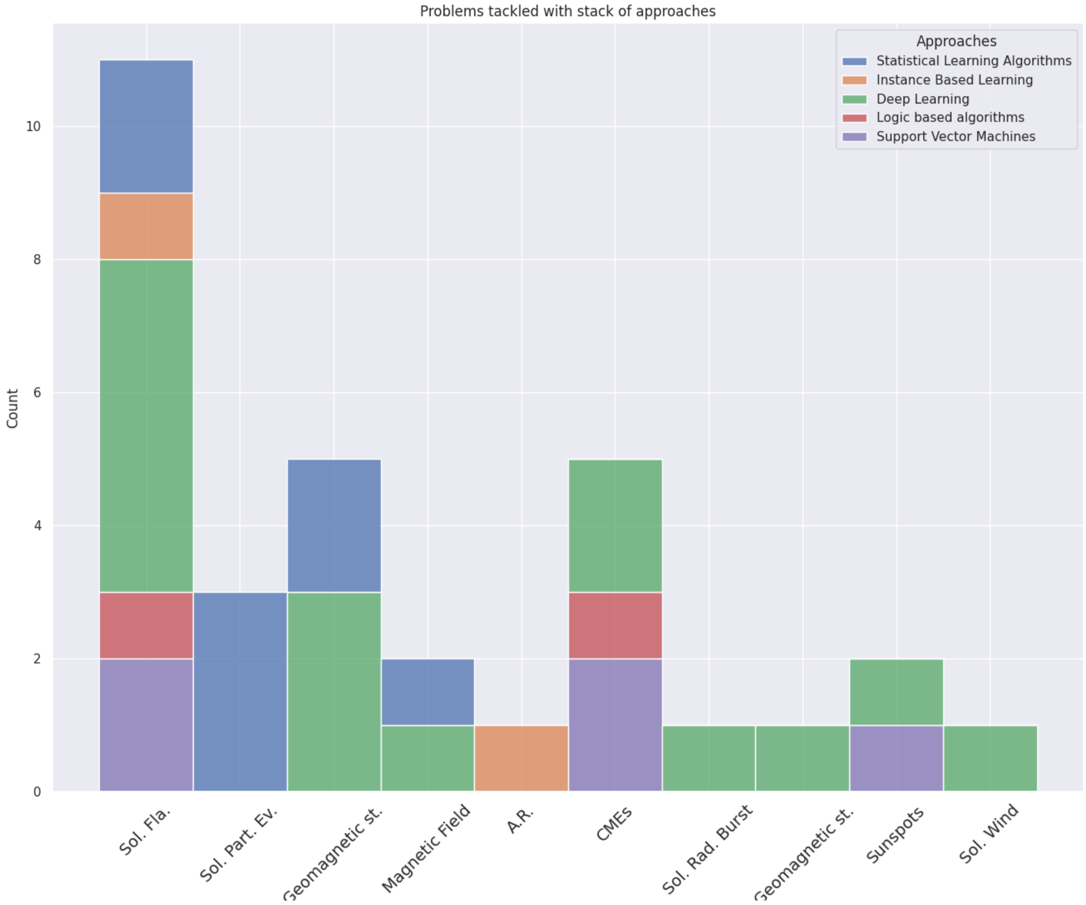


Figure 3.2: Literature Review analysis for Machine Learning techniques for solar weather problems. Solar flares are shown to have the preference of these applications followed by geomagnetic storms and CMEs

3.2 Machine Learning Approaches for Simulation Initial Data Improvement

This section intends to explore how machine learning has been used as a way of improving simulation's initial data/conditions. To do so, an ad-hoc search in google scholar[2] was made throughout the use of combinations pertaining the words and expressions "machine learning", "simulation", and "initial conditions" was made.

In Kochov *et al.* 's [59] work fluid dynamics simulations, also know as computational fluid dynamics (CFD), are accelerated and improved by the use of deep learning. Even-though fluids can be described using the Navier-Stokes equations, solving such is many times limited by the computational cost it encompasses. As such, the authors use deep learning as a way of improving approximations inside computational fluid dynamics for modeling two-dimensional turbulent flows. The methods used are learned solvers, learned interpolation and learned correction. The

[2] https://scholar.google.com/

model for each time step then consists of a convolutional neural network controlling learned approximations inside the convection calculation of a standard numerical solver of advected and advecting velocity components. Using supervised learning, the loss function of the model is given by the cumulative point-wise mean squared error between the predicted and true velocities. The authors then compare the performance of learned interpolation top alternative ML methods. The method described in the paper achieves the same accuracy as traditional finite volume methods whilst augmenting the resolution up to 10 times finer and performing the computation calculations up to 80 times faster.

In Lattimer *et al.* 's [60] work, the need to provide rapid high-fidelity predictions of fires is enhanced and as such, the author's give an overview of ML methods and techniques used to perform low computational cost predictions. The two main approaches discussed are dimensionality reduction and deep learning. It is stated that unsupervised dimensionality reduction using principal component analysis has been used for simple plumes without reactions and for wildfire spread models. In those works, it is shown that reduced order models were 2 to 3 orders of magnitude less than the CFD models regarding computational costs. Deep learning is moreover discussed to be an useful tool to not only predict estimates of a single quantity-point but also to create generative models. These approach has been used for CFD applications with the limitation of not having been explored to perform full-field predictions with changes in general boundary conditions. The paper concludes by showing how promising ML techniques can be for predicting both single points and full generative models regarding physical simulations.

In P Watson's [61] work, the author explores the use of ANNs in combination with physically derived models so as to predict the chaotic Lorenz '96 system. The idea behind this combination comes with the advantage of still understanding the physicality of the calculations whilst not needing to feed the ANNs with too much data. In this work, error-correcting ANNs are used so as to reduce the error in the used coarse-resolution model. The ANNs used had a multi-layer perceptron architecture and were trained to predict the difference between the true system 5tendency and the one predicted by the model. Different arrangements were tested being that more complex ANNs proved to be more effective than simpler ones given that for the largest tested ANN the maximum error reduction on the validation dataset was of 42%. The Rooted Mean Squared Error (RMSE) on the validation dataset is shown to be below 3% increase from the training one, which indicates over-fitting has been avoided.

In Tongal *et al.* 's [62] work, three machine learning models are tested for simulating and forecasting steam-flows. These models, Random Forests (RF), ANNs and SVRs, were used coupled with base flow separation. Separation of these base flows seems to be the key factor on the augmentation of performance of these models compared to previous work using these. As input data four rivers' steam-flow from the United states were used. Aside from the simulation scheme, a forecasting scheme was also employed by sing antecedent discharge values in addition to the values of precipitation (P), temperature (T) and evapotranspiration (PET) used in the simulation framework. The authors present several metrics for comparison, being these Root Mean Square Error), Nash-Sutcliffe efficiency (NSE), Kling-Gupta efficiency (KGE), Volumetric efficiency (VE), Index of Agreement (d) and Persistence index (Pi). ANN (best in 3 out of 4 flow-streams) and SVR (best in 1 out of 4 flow-streams) models performed better than RF model in the calibration period. There were improvements in model performances in the validation versus the calibration period. However, in this period RF and SVRs beat ANNs due ANNs' poor generalization capacity.

3.3 Initial Conditions in Physical Simulations and Models

This section intends to explore the impact initial conditions may have when used in simulations related to astronomy and cosmology as well as understand what techniques have been used to calculate the initial conditions.

To do so, an ad-hoc search in google scholar[3] was made throughout the use of combinations pertaining the words and expressions "physical simulation", "physical model" and "initial conditions" was made.

Hahn *et al.* [63] propose a new algorithm using an adaptive convolution of Gaussian white noise with a real-space transfer function kernel and a multi-grid poisson solver to generate displacements and velocities following first- (1LPT) or second-order Lagrangian perturbation theory (2LPT). The main purpose of the discussed work is to generate multi-scale initial conditions with multiple levels of refinements for cosmological 'zoom-in' simulations. Lagragian pertubation theory describes the evolution of density perturbations in the restframe of the fluid. 2nd order Lagrangian theory (perturbations) is used in order to solve solve accuracy issues related to 1st order perturbation theory. The authors develop a hybrid Poisson solver, using an adaptive multi-grid algorithm for inter-level gravity and a Fast Fourier Transform based Poisson solver for the finest grid. The tests to the proposed model were comprised of three parts. Firstly, the accuracy of the model is shown by finding RMS errors velocity and displacement fields order of 10^{-}4 showing an improvement of about two orders of magnitude over previous approaches. Secondly, using a re-simulations of a galaxy cluster halo the model is shown to be able to reproduce correlation functions, density profiles, key halo properties and sub-halo abundances with one hundred percent accuracy. Thirdly, generalizing the model for two-component baryon and dark-matter simulations, it is shown that the power spectrum evolution is in perfect agreement with linear perturbation theory.

In Crocce *et al.* [64] the authors study the impact of using such initial conditions in numerical simulations related precisely to cosmology. As such, the paper compares the use of Zel'dovich approximation (ZA) initial conditions and the use of more accurate initial conditions based on second-order Lagrangian perturbation theory (2LPT). Since Zel'dovich initial conditions are known to excite long-lived transients in the ts in the evolution of the statistical properties of density and velocity fields, it is shown, therefore, that 2LPT initial conditions reduce transients significantly and being are much more appropriate for numerical simulations concerning precision cosmology.

[3] https://scholar.google.com/

In the scope of the Horizon collaboration, in Prunet *et al.* [65] a software package used to produce initial conditions for large scale cosmological simulations is described. Firstly, the authors explain the use of grid-based initial conditions where having a 3 dimensional Gaussian random field $\delta(x)$ (representing the density), if we consider zero-mean fields and their power spectra $P(k)$ than it is proven that $\left\langle \left|\delta(k)^2\right| \right\rangle = P(k)$. When dealing with dark matter then particles Lagrangian coordinates are used to remap the particle velocities to grid cells. Secondly, the paper describes grid-based implementation since given the white noise is has already been chosen in on a grid of a given size, it possible to use them to generate initial density and velocity realizations with the desired cosmology and power spectrum. Such is done as described before but by normalizing the parameters and constraining the initial conditions to lower-frequency. Afterwards, low-pass filtering and re sampling are applied, being followed by power spectrum estimation, estimation of matter density on a grid from a set of particle positions. Finally the authors decompose the domain using the Peano-Hilbert domain space filling curve decomposition which provides a complete mapping between the 3D position of a grid point and a 1D coordinate on this curve. The codes were validated up to resolutions of 4096^3 and used to generate the initial conditions of large hydrodynamical and dark matter simulations proving to be more versatile than previous solutions.

In Brown *et al.* [66] the authors propose an approach of customization to the root grid zoom-in initial conditions utilized for simulations of galaxy formation. The work starts by cutting a minor region of interest from the white noise used to seed the structures of an existing initial condition. This cut is used to create a new root grid allowing for a lesser box volume. As to not disturb the zoom region, the original dark matter particles and gas cells are put within the new root grid, with solely with the addition of a bulk velocity offset. To test the approach, collision-less simulations are run comparing the original and new ICs, finding plausible agreement. It is found that e two bigger galaxies at the original ones within a range of 15% The times and masses of major mergers, the full dark matter accretion histories, the maximum circular velocity and the distance from the central galaxy are all reproduced in agreement as well. It should be noted that by dimensionality reduction of the problem, the authors are able to achieve e CPU run-time speed up of up two.

In Jasche *et al.* [67] a probabilistic physical model of non-linearity evolved density field is presented. Using Bayesian exploration, the authors reconstruct the present density and velocity fields from large-scale structure surveys, including a full propagation of the observational uncertainties of the initial conditions. This physical model gives accurate reconstructions of the underlying present-day density and velocity fields on scales larger than $6Mpch^{-}1$. The method accurately reconstructs non-linear features corresponding to three-point and higher order correlation functions. Tests of the reconstructed initial conditions show statistical consistency with the Gaussian simulation inputs. Such approach demonstrates that statistical approaches based on physical models of large-scale structure distribution are now becoming feasible for realistic current and future surveys.

3.4 Summary

In Section 3.1 we presented a study and analysis of peer work in machine learning techniques being applied to Space weather. We concluded that there is an increasing interest in using machine learning techniques to solve solar weather forecasting and classification problems whilst having an increase in the interest in applying deep learning techniques to such problems. The prediction of solar flares as well as CMEs and geomagnetic storms have the higher rate of attempted solutions given the possible impact of such events on Earth. In sections 3.2 and 3.3 we described peer work done in applying machine learning learning approaches for simulation initial data improvement and introduced initial conditions in physical simulations and models. We have then deduced that there is pertinence in applying neural networks to the MULTI-VP simulation in order to improve simulation times given the peer worked reviewed.

Chapter 4

Research Statement

In this chapter an overview of the problem this dissertation is trying to solve is given in section 1.3. An hypothesis is given in section 4.2 and research question to validate such an hypothesis are provided in section 4.3. Finally in section 4.4 experimental methods are explored in order to validate and evaluate our work.

4.1 Problem Definition

The solar wind, driven along magnetic flux tubes, spreads from the Solar atmosphere to distances beyond the solar system. The properties of solar wind *in-situ* in the vicinity of Earth have been measured for a few decades, but the physical conditions that determine the differentiation between different types of wind are those of the lower Solar atmosphere, which are not easily measured. This incapacity to measure has become one of the core difficulties of this subject.

At 1 au, the solar wind speed bestows cyclic fluctuations in latitude and time, matching the evolution of the global background magnetic field during the activity cycle. It is commonly accepted that the terminal wind speed in a given magnetic flux-tube is generally anti-correlated with its total expansion ratio, which motivated the definition of widely used semi-empirical scaling laws relating one to the other. In practice, such scaling laws demand ad hoc revisions and empirical fits to *in-situ* spacecraft data.

Studies using magnetohydrodynamical (MHD) simulations of the corona and wind coupled to a dynamo model to determine the coronal magnetic field's properties and the wind velocity during a whole 11-year activity cycle have been made. These MHD simulations present a large statistical ensemble of open flux-tubes that were analyzed conjointly to identify relations of dependence between the wind speed and geometrical parameters of the flux-tubes that are valid globally (for all latitudes and moments of the cycle). These MHD models work as follows.

Firstly access to observations and measurements of the magnetic field at the Sun's surface is obtained. Then, with some well-known techniques, these are extrapolated to the entire solar corona. Secondly, from the obtained extrapolation, one can deduce the elementary geometry of each stream of the solar wind, which is due to the physical regime one is in, conducted by the magnetic field. Thirdly, a solar wind acceleration model. This model uses this geometric information from the flow tubes. This stage is the least trivial part of the problem.

There has not yet been a reasonable a priori estimate of what the properties of each solar wind element will be, and even less what the solar conditions will generate in terms of time series at 1 AU - which would be directly comparable to the observable in-situ, and that results from going through several consecutive elementary streams of solar wind. Afterward, a physical model that allows the calculation of the properties of each element from the surface of the Sun to far away from it is used. The final target is the asymptotic solar wind, which can be measured at 1AU. First, however, one must consider all the acceleration processes that act progressively in a vast region of the solar atmosphere. Studies confirm that the solar wind's terminal (asymptotic) speed depends very strongly on the geometry of the open magnetic flux tubes through which it flows. As a result, the total flux-tube expansion is more clearly anti-correlated with the wind speed for fast rather than for slow wind flows and effectively controls the locations of these flows during solar minima. Overall, the actual asymptotic wind speeds attained are also strongly dependent on field-line inclination, and magnetic field amplitude at the foot-points [5]. Some improvements on MHDs have been made, and models such as MULTI-VP [68] have been recently developed and used to compute the three-dimensional structure of the solar wind and include the chromosphere, the transition region, and the corona and low heliosphere. The model computes the three-dimensional structure of the solar wind and includes the chromosphere, the transition region, and the corona and low heliosphere.

It calculates a large ensemble of wind profiles flowing along open magnetic field lines that sample the entire three-dimensional atmosphere or a given region of interest. The radial domain starts from the photosphere and typically extends to about 30 solar radius. The elementary unidimensional wind solutions are based on a mature numerical scheme adapted to accept any flux-tube geometry. The results support the hypothesis that the geometry of the magnetic flux tubes in the lower corona controls the distribution of slow and fast wind flows. Furthermore, the inverse correlation between density and speed far away from the Sun is a global effect resulting from minor readjustments of the flux-tube cross-sections in the high corona (necessary to achieve global pressure balance and a uniform open flux distribution). MULTI-VP performs much faster than other global MHD models and does not suffer from spurious cross-field diffusion effects. Furthermore, it is shown that MULTI-VP can correctly predict the dynamical and thermal properties of the background solar wind (wind speed, density, temperature, magnetic field amplitude, and other derived quantities) and approach real-time operation requirements. It is concluded that the quality and performance of this physical model depends on the quality of an initial guess, which would be the initial condition for the simulation. Another poignant observation is that these models take some computation time which can be reduced by starting the simulations with good initial guesses. As such, the problem addressed by this dissertation is to decrease the time taken to come to a feasible solution using the physical model, MULTI-VP, described in Pinto *et al.* by obtaining good initial guesses [68].

4.2 Hypothesis

Considering the *MULTI-VP* simulation and its use, the primary research question this work attempts to answer is the following:

> How could Neural Networks be used to improve *MULTI-VP*'s performance pertaining predictions and computation time?

Following the previous research question, we claim the following hypothesis:

> Neural networks can be used to shorten the computation time needed for solar wind flux-tubes simulations made by the simulator *MULTI-VP* by learning to provide good initial guesses from previous runs.

Henceforth, we attempt to validate such hypothesis considering that (1) a decrease in computation time corresponds to a decrease in comparison to the time the simulation currently takes to produce results; (2) good initial guesses to be those similar to the expected ones and that lower the computation time.

4.3 Research Questions

To validate our hypothesis and validate our work, we have defined the following research questions:

RQ1 *Can neural networks acquire skill in initial guess estimation of solar wind flux-tubes simulations?*

RQ2 *Do initial guess estimations from Neural Networks improve solar wind flux-tubes simulation times?*

Answering these questions will allow us to assert the possible falsifiability of our hypothesis.

4.4 Validation and Evaluation

Zelkowitz *et al.* [69] define experimental models for validating technology. In this work, two main experimentation methods referred in Zelkowitz's work will be used.

Engineering method Engineers develop and test a solution to a hypothesis. Based upon the results of the test, they improve the solution until it requires no further improvement.

Empirical method A statistical method is proposed as a means to validate a given hypothesis. Unlike the scientific method, there may not be a formal model or theory describing the hypothesis. Data is collected to verify the hypothesis.

The engineering method is used in the first phase of our work when a solution for predicting the simulation outputs is made. The empirical method is used in a later phase of this work when we gather the data provided by the simulation to test our hypothesis.

These processes are explained in chapters 5 and 6.

4.5 Summary

Section 4.1 started by contextualizing the scope of the work done by Pinto *et al.* [68] and stating the problems currently faced by the MULTI-VP simulation. Section 4.2 details the hypothesis of this dissertation whilst section 4.3 detailed the research question this dissertation aims answering. Finally in section 4.4 the validation and evaluation methods to be used in this dissertation were presented.

Chapter 5

Exploratory Data Analysis

In this chapter as a first exploration of the available data is made. In section 5.1 the data is explained in detail whilst in section 5.2 an exploratory analysis of the data is made. Finally a small summary of the developed chapter's work is made in section 5.3.

5.1 Data

The data used in the experiments consisted of a set solar wind profiles randomly selected from a pool of simulations. Each one of these correspond to a given date and contains 2596 distinct instances each. Each instance corresponds to one independent simulation of an elementary solar wind flux-tube that contains a list of different physical quantities given in 640 points (abscissas) ordered as a function of distance to the Sun (up to about 31 solar radii). These simulations require an initial guess provided by the user, that the numerical code will evolve towards a physical solution.

These files contained twelve columns being these $R[Rsun]$, $L[Rsun]$, $lon[Carr]$, $lat[Carr]$, $B[G]$, A/A_0, $alpha[deg]$, V/Cs, $propag\_dt[d]$, $n[cm^{-3}]$, $v[km/s]$ and $T[MK]$. R represents the radial coordinate radius whilst L corresponds to the distance measured throughout the line. lon and lat represent the longitude and latitude of the point relative to the Sun. B stands for the magnetic field whilst A/A_0 represents the expansion term with A being the flow diameter and A_0 the initial diameter. $alpha[deg]$ indicates the inclination of the flux tube, V/Cs represents wind over sound speed and $propag\_dt[d]$ is the propagation time of the plasma from the solar surface to the point where we are. $n[cm^{-3}]$ is the umber of particles per unit volume (number of ionized H protons), $v[km/s]$ is the speed oriented along the line and $T[MK]$ is the temperature at that point in space.

5.2 Data Preparation and Analysis

Before analysing the data, some preprocessing techniques were performed in order to obtain more accurate representations of the data. To do so, we started by renaming the columns of the file for consistency (stripping extra-space and capitalizing the first letter of the columns leaving the rest in lower case.) We then merged the data from all files sampling one random line from each, obtaining a file with 12941 lines.

Data representation of a MULTI-VP generated flow can be seen in A.1 where one of the files is represented.

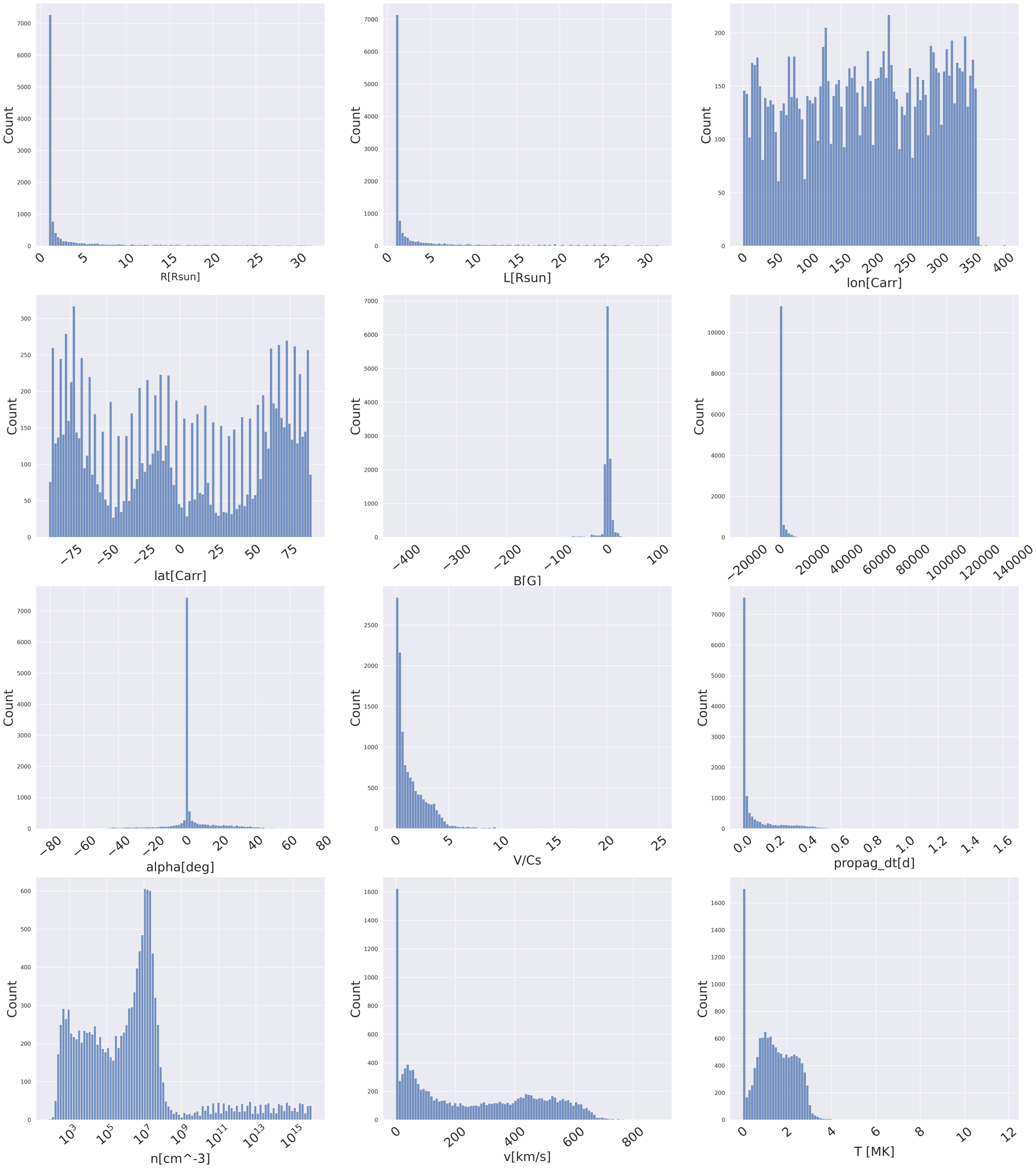


Figure 5.1: Uni-variate histograms representing the different features distributions.

Since we had a considerably large amount of data, we decided to drop lines where missing or null data was placed as well as ones where there were any mistypes. Therefore, from the expected 14760 lines, 1819 were discarded.

Table 5.1: Data Statistical Description with outliers. Count, mean, maximum, minimum, standard deviation and quartiles are represented for the different available variables.

	R[Rsun]	L[Rsun]	lon[Carr]	lat[Carr]	B[G]	A/A0
count	12940	12940	12940	12940	12940	12940
mean	4.7356	4.7908	185.9409	0.2045	-0.4920	1032.8481
std	7.1279	7.1700	104.2467	57.1295	13.8260	5249.1044
min	1.0000	1.0000	0.1292	-89.7902	-419.6305	-22152.6201
25%	1.0212	1.0212	97.5007	-55.7545	-1.6216	1.0598
50%	1.1579	1.1723	189.7790	-4.5143	0.0017	2.4265
75%	4.1951	4.3177	277.9799	58.0083	2.4711	163.5425
max	31.4920	31.5014	403.6049	89.9871	101.3155	135424.4260

	alpha[deg]	V/Cs	propag_dt[d]	n[cm^-3]	v[km/s]	T[MK]
count	12940	12940	12940	12940	12940	12940
mean	1.9863	1.5248	0.072813	8.3055e+13	256.1420	1.393164
std	14.4639	2.03380	0.133364	6.6338e+14	214.7340	0.899265
min	-80.5875	-3.4646e-07	0.000000	7.8516e+01	-0.0000	0.006000
25%	-0.0747	2.8999e-01	0.001693	1.6583e+04	51.2458	0.724575
50%	0.0000	8.2632e-01	0.008821	2.2975e+06	212.9088	1.345264
75%	1.2906	2.2009e	0.072140	2.0433e+07	449.8903	2.124096
max	72.8551	2.4942e+01	1.617492	1.0054e+16	886.0651	11.867441

The data distribution can be seen in Figure 5.1. A general statistical overview containing the mean, standard deviation, minimum, maximum and percentage quartiles of the data can be seen in Table 5.1.

In order to analyse the data a study of outliers, correlation and distributions was performed.

We started then by considering outliers detection using the definition provided by Hawkins[70] which defined an outlier as an observation that deviates so much from other observations that it incites suspicion that it was generated by a different mechanism and the definition of Johson *et al.* [71] which defines an outlier as an observation in a data set which appears to be inconsistent with the remainder of that set of data. Representing the data in Figure 5.2, we can spot some outliers in columns $B[G]$, A/A_0, $alpha[deg]$ and $T[MK]$.

$$\rho_{XY} = \frac{cov(XY)}{\sigma X \sigma Y} \tag{5.1}$$

Afterwards we proceeded with analysing the correlation between our different variables. The Pearson correlation, described in equation 5.1 can be defined as the measure of linear correlation between two sets of data [72].

As can be seen in Figure 5.3, one can find a strong correlation between the distance to the Sun $R[Rsun]$, $L[Rsun]$ and propagation $propag_dt[d]$ in days which was expected. Some degree

Table 5.2: Data Statistical Description with no outliers. Count, mean, maximum, minimum, standard deviation and quartiles are represented for the different available variables.

	R[Rsun]	L[Rsun]	lon[Carr]	lat[Carr]	B[G]	A/A0
count	12927	12927	12927	12927	12927	12927
mean	4.7342	4.7892	185.9532	0.1994	-0.4593	1035.8614
std	7.1285	7.1705	104.2510	57.1478	13.3321	5247.6020
min	1.0000	1.0000	0.1292	-89.7902	-132.8306	0.3483
25%	1.0212	1.0212	97.5007	-55.8247	-1.6266	1.0603
50%	1.1576	1.1723	189.7719	-4.5291	0.0017	2.4369
75%	4.1882	4.3024	278.0201	58.0238	2.4801	164.6877
max	31.4920	31.5014	403.6049	89.9871	101.3155	135424.4260

	alpha[deg]	V/Cs	propag_dt[d]	n[cm^-3]	v[km/s]	T [MK]
count	12927	12927	12927	12927	12927	12927
mean	1.9827	1.5248	0.0728	8.3138e+13	256.1073	1.3925
std	14.4621	2.0345	0.1334	6.6371e+14	214.7809	0.8949
min	-80.5875	-3.4646e-07	0.0000	7.8516e+01	-0.0000	0.0060
25%	-0.0749	2.8995e-01	0.0017	1.6920e+04	51.2289	0.7244
50%	0.0000	8.2565e-01	0.0088	2.2990e+06	212.8528	1.3455
75%	1.2959	2.2008e	0.0720	2.0463e+07	449.9966	2.1243
max	72.8551	2.4942e+01	1.6175	1.0054e+16	886.0651	4.2241

of correlation can be found between V/Cs and the distance to the sun as well as A/A_0 and, as expected, $v[km/s]$. This tells us that the further away from the Sun, the bigger the diameter and velocity of the wind flux tube.

A further analysis on the dependencies of the variables can be seen in Figure 5.4. In such Figure the comparison is made between the variables which we shall later in chapter 6 consider as inputs and outputs.

We then proceeded with removing the outliers and analysing data once more so as to better understand the differences. To do so the following conditions had to be met: $['B[G]'] >= -200$, $['A/A0'] >= 0$, $['T[MK]'] <= 6$. In Figure 5.5 we can observe the dependencies of the inputs vs outputs and are able to better analyse such dependencies with no outliers. We can also observe in Table 5.2 the difference in the general statistical overview.

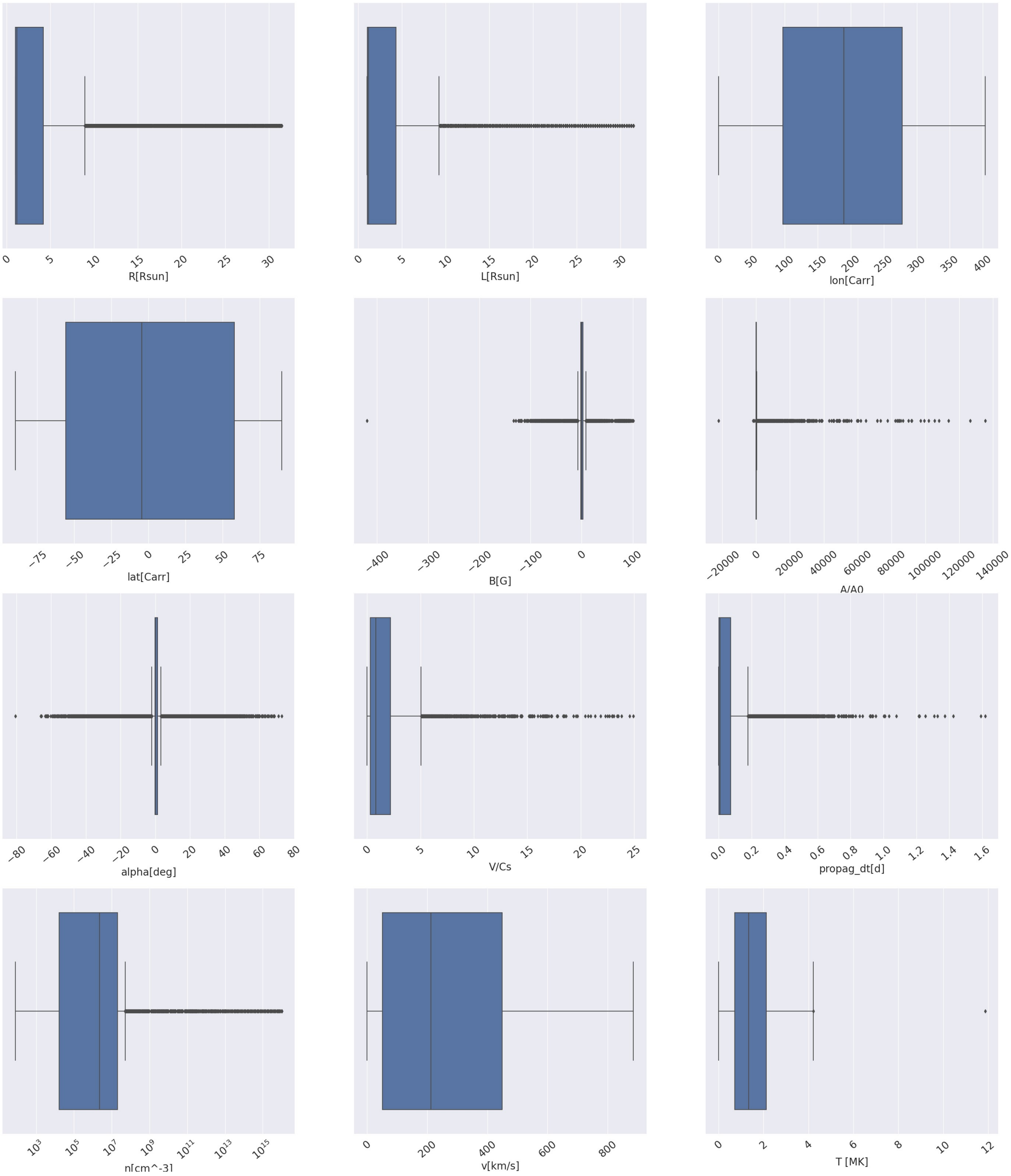


Figure 5.2: Box Plots for each variable so as to detect outliers.

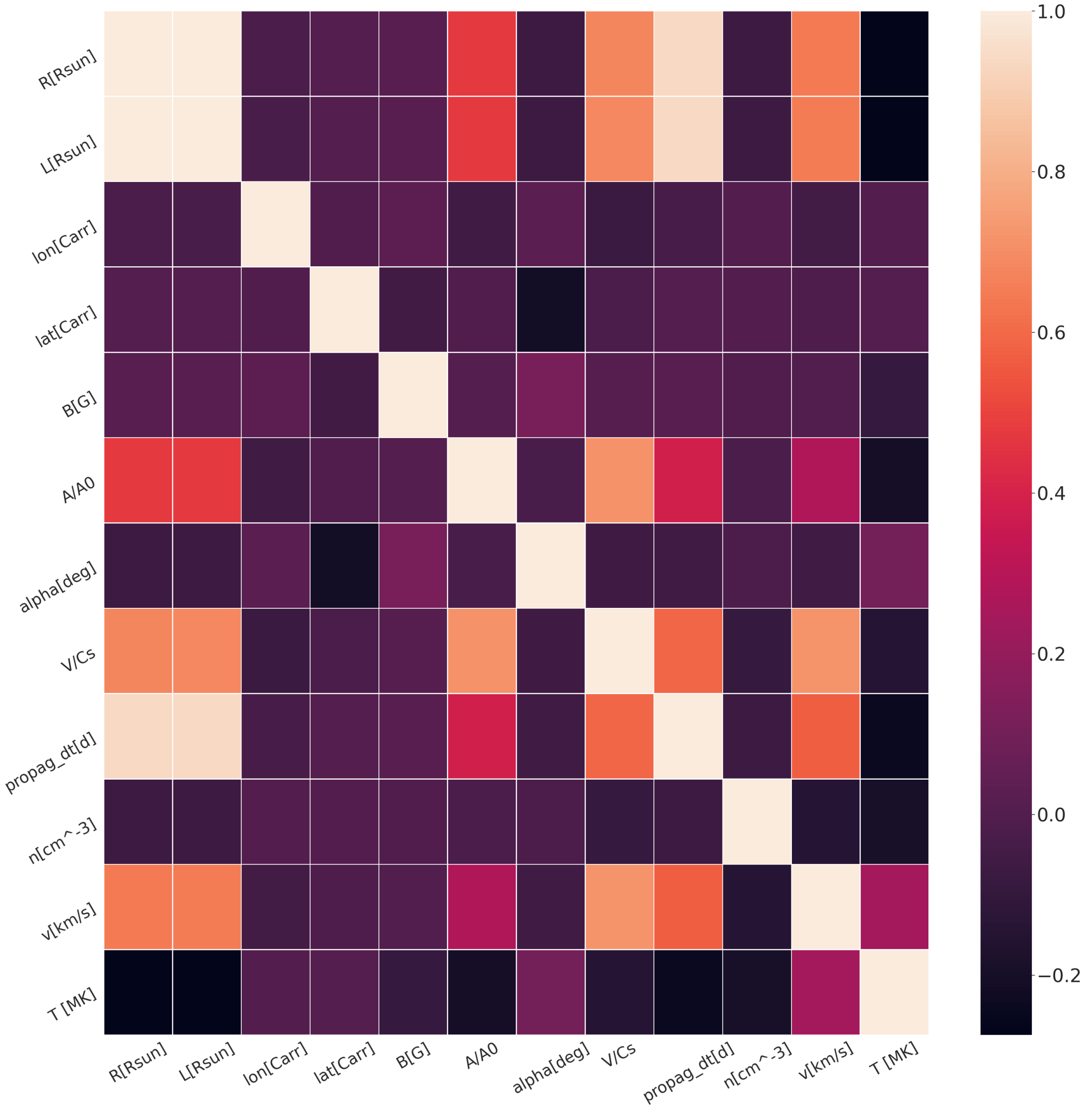


Figure 5.3: Correlation Heatmap representing the different correlations that can be found between all variables. The more intense the red or the blue in each square, the more directly or inversely correlated are two variables, respectively.

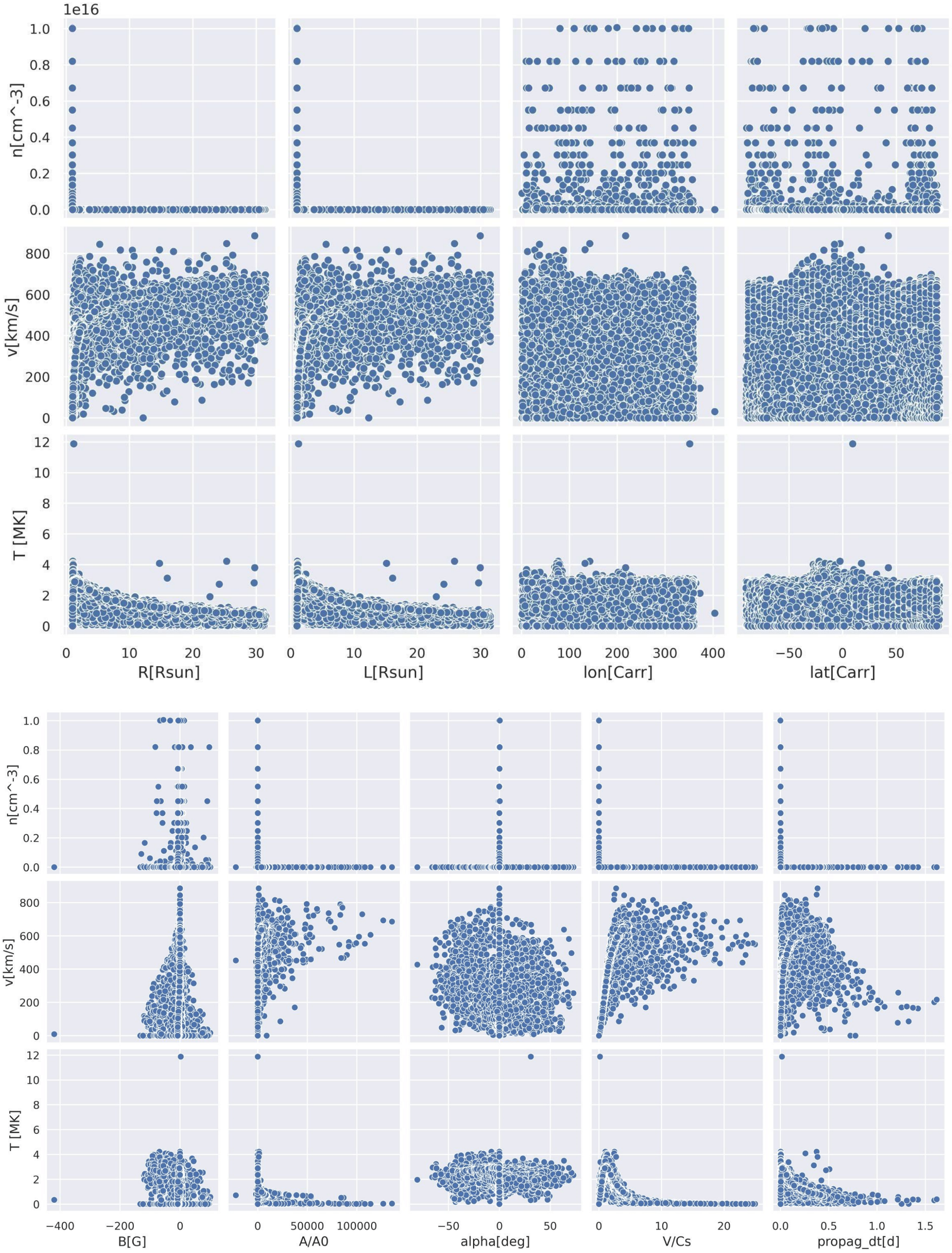


Figure 5.4: Multivariate pair plots with outliers. The graph aids in understanding the different relations between each pair of of variables concerning inputs vs outputs.

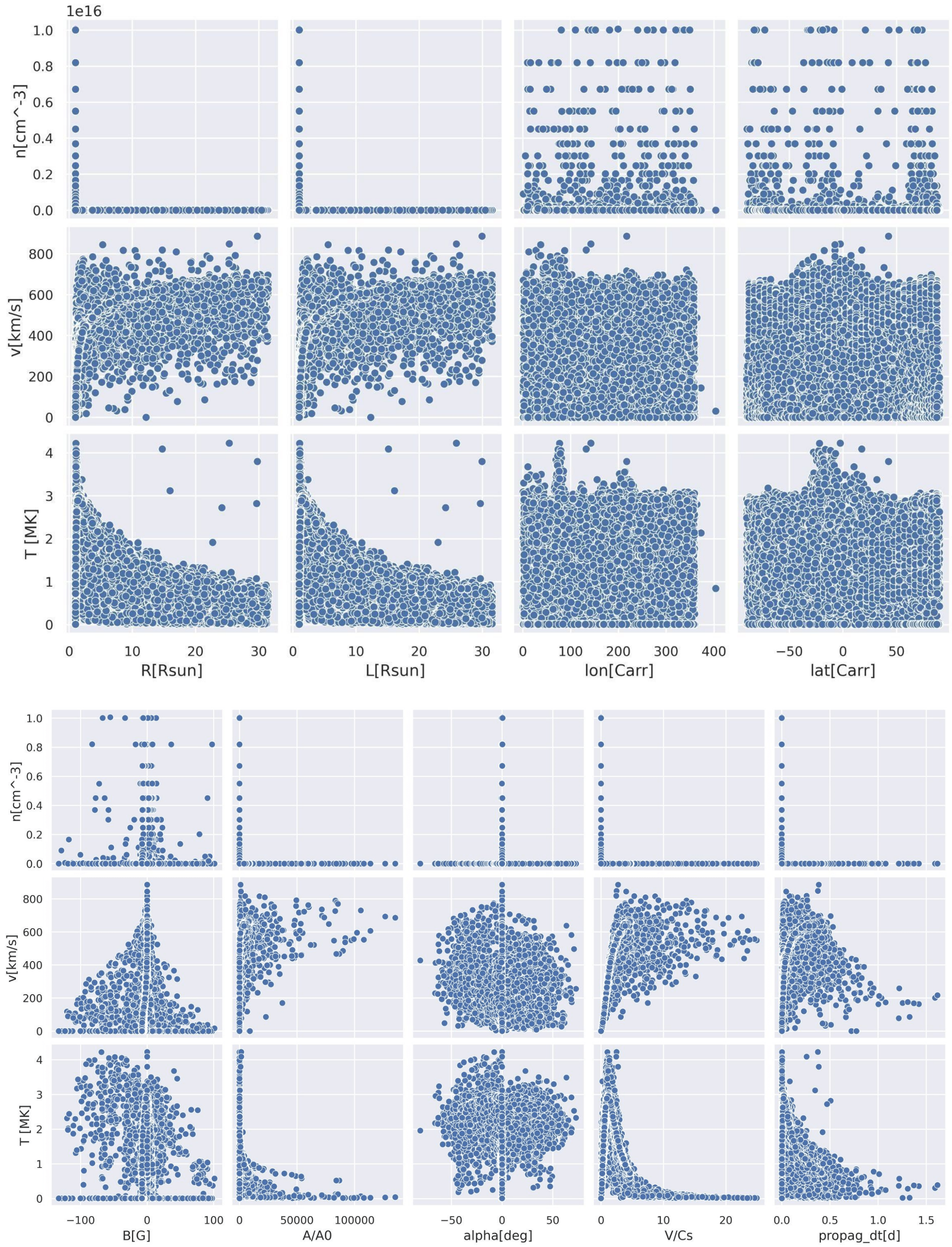


Figure 5.5: Multivariate pair plots with no outliers. The graph aids in understanding the different relations between each pair of variables concerning inputs vs ouputs.

5.3 Summary

In Section 5.1 we presented the data used in this work while in Section 5.2 we describes the preprocessing techniques that were applied the data and analysed it. We were able to verify that some of the features possess a large range of values and that one can detect a considerable number of outliers in some features. We also found some correlation between V/Cs and the distance to the sun as well as A/A_0 and, as expected, $v[km/s]$ were able to conclude that the further away from the Sun, the bigger the diameter of the flux tube and the larger the wind velocity.

Chapter 6

Experiments and Results

This chapter explores the different experiments and corresponding results made throughout this dissertation. We start by using a point-based approach in section 6.1. A flow-based approach is presented in section 6.2 and simulation runs and comparisons are made in section 6.3. Validation threats to our work are presented in section 6.4 and the hypothesis presented in 4.2 as well research questions presented in 4.3 are discussed in section 6.5. Implementation details are detailed in section 6.6. In the end, a short summary and brief discussion of our results is made in section 6.7.

6.1 Point-Based Approach

In order to answer the first research question we stated in chapter 4 we decided to start our approach with the use dense neural networks. We then tested these against random-based and median-based neural networks. Such neural networks are described further ahead.

The first step consisted of normalizing the data using the quantile scaler provided by the keras library since it has been found that input data normalization with certain criteria, prior to a training process, is crucial to obtain good results as well as to fasten significantly the calculations (especially in the training phase of the models) [73].

The used scaler transforms the features such that they follow a uniform or a normal distribution. As such, for a given feature, this transformation tends to spread out the most frequent values. In doing so, it also reduces the impact of outliers. The transformation is applied on each feature independently. First an estimate of the cumulative distribution function of a feature is used to map

Table 6.1: Best hidden layers configurations for N, V and T prediction. The mean squared error as well as its standard deviation are presented for each of the selected configurations.

Output	HL	HN	mean_mse	stdv
N	2	5	5.70×10^{-4}	1.12×10^{-4}
V	2	5	1.11×10^{-2}	2.92×10^{-3}
T	2	5	3.82×10^{-2}	1.22×10^{-2}

the original values to a uniform distribution. The obtained values are then mapped to the desired output distribution using the associated quantile function. Features values of new/unseen data that fall below or above the fitted range will be mapped to the bounds of the output distribution. As such this is not a linear transformation.

We started our experiments by dividing the prediction of each output: N, V and T into three different models. As such we avoided weighing on one of the outputs more than the others.

The sequential model provided by the keras library was used for providing us a straightforward approach for producing a plain stack of layers where each layer has exactly one input tensor and one output tensor [74].

A systematic experimentation with different configurations of hidden layers was made as recommended by Stathakis [75]. Since a single-layer neural network can only be used to represent linearly separable functions [76]. we started our experimentation with two hidden layers and five nodes on each hidden layer.

We started by randomizing the training data. We then divided the data into testing data and training using a percentage of 30% for testing and 70% for training. A first run of the initial configuration and its results for MSEs were stored. We started by analysing the number of hidden layers that would benefit our loss function (MSE) the most. To do so we started adding hidden layers to the model and comparing the results amongst such configurations. In order to make sure that our results were not random of biased we ran the tests using a 10-fold validation with 10 experiments for each configuration. While the average measured MSE was smaller than then the previous configuration we would keep adding layers until an optimal configuration was achieved.

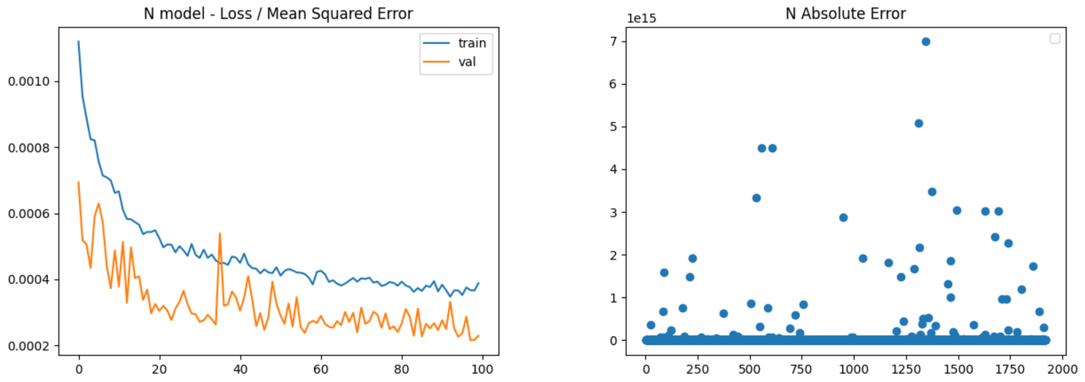
The process described above helped us getting the number of layers for each output which ended up being optimal with two layers for all of them. In Table 6.1, we can observe the winning configuration for each output as well as the mean of the 10-fold tests and its standard deviation.

Since getting the number nodes for each output would be quite exhaustive we decided to use the keras built-in tuner [77]. The Keras Tuner is a library that helped us pick the optimal set of hyperparameters for our TensorFlow program (also know as hyperparameter tuning). The tuners tried in this work were the RandomSearch and the Hyperband tuners. As the name suggests, the RandomSearch hyperparameter tuning method randomly tries a combination of hyperparameters from a given search space. Hyperband optimizes the random search method through adaptive resource allocation and early-stopping. It first runs random hyperparameter configurations then selects which configurations perform well, then continues tuning the best performers.

Before advancing with the training and predictions of the model, it was decided to add a dropout layer to the model. As explained in the work of Srivastava *et al.* [78], overfitting is a serious problem in neural networks. Dropout comes a solution for addressing this problem.It works by randomly dropping units, along with their connections from the neural network during training. In doing so, it prevents units from co-adapting too much and gives major improvements over other regularization methods. As such the dropout layer was added and the results for each output were as follows.

For the N output the best tuner was the Random Search which returned a configuration of two hidden layers with 52 nodes each and had an average MSE of 2.83×10^{-4}. For the V output the best tuner was the Random Search which returned a configuration of two hidden layers with 44 nodes each and had an average MSE of 2.32×10^{-3}. For the T output the best tuner was the Random Search which returned a configuration of two hidden layers with 48 nodes each and had an average MSE of 4.81×10^{-3}.

To use the tuner a hypermodel with two dense layers and a random number of nodes was generated. We proceeded then by training the models with 250 epochs and by splitting the data into training, testing and validation in a 70/15/15 ratio. The results are as follows. The model used for the N output had a training MSE of 1.33×10^{-4}, a validation MSE of 1.93×10^{-5} and a testing MSE of 1.58×10^{-4}. The model used for the V output had a training MSE of 4.34×10^{-4}, a validation MSE of 5.25×10^{-4} and a testing MSE of 5.99×10^{-4}. The model used for the T output had a training MSE of 1.15×10^{-3} , a validation MSE of 1.501×10^{-3} and a testing MSE of 1.391×10^{-3}.



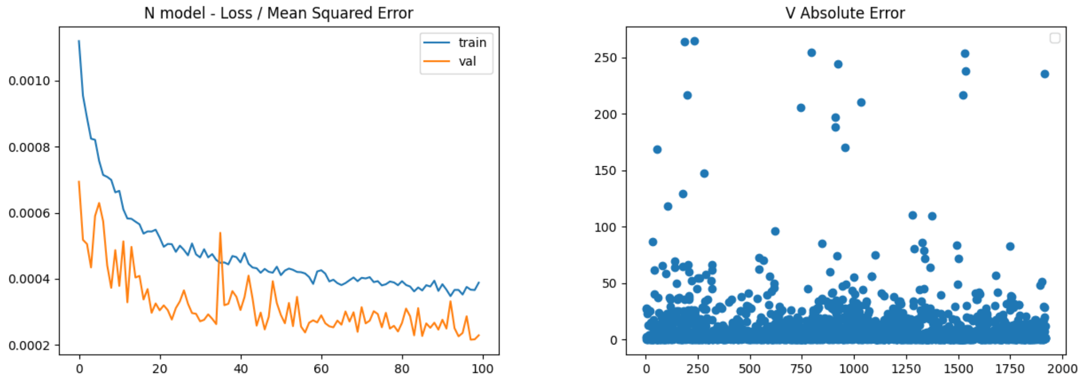
(a) Epochs' History

(b) Absolute Error

Figure 6.1: N model training history and absolute error in final predictions using a dropout layer. A diminishing MSE can be observed through the training epochs and the validation MSE stands below training MSE for most of the time.

In Figures 6.1, 6.2 and 6.3 we can see 100 of the 250 epochs of training of the *N*, *V* and *T* models whilst also observing the absolute error in the predictions of the training phase for each of the models.

Analysing both these results and the graphs one can see that the MSE for the training data tends to stabilize near the 80 epochs. We can also observe that for the N and V outputs the validation



(a) Epochs' History

(b) Absolute Error

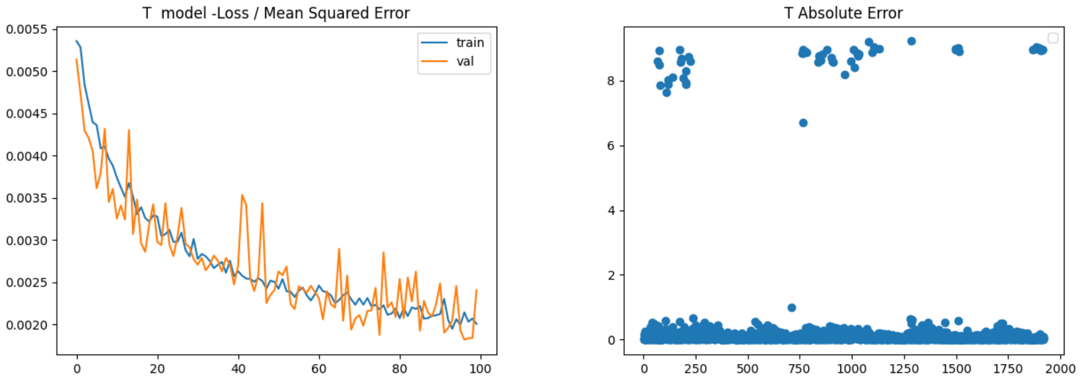
Figure 6.2: V model training history and absolute error in predictions using a dropout layer. A diminishing MSE can be observed through the training epochs and the validation MSE stands below training MSE for most of the time.

MSE tends to be lower than the testing MSE and that for the T output the training and validation MSEs are, for the most part, aligned with each-other. We can also observe that even though the MSE tends to be quite low, some absolute errors in the prediction stage are quite high. As such we decided to test the influence of both outliers and the dropout layer.

We decided then to start by comparing the results one might have obtained without using a dropout layer. The results for the tuner for each output were as follows: For the N output the best tuner was the Random search which returned a configuration of two hidden layers with 60 nodes each and had an MSE of 4.24×10^{-4}. For the V output the best tuner was the Hyperband Search which returned a configuration of two hidden layers with 48 nodes each and had an MSE of 1.954×10^{-3}. For the T output the best tuner was the Random Search which returned a configuration of two hidden layers with 48 nodes each and had an MSE of 6.358×10^{-3}. Just like before, we proceeded then by training the models with 250 epochs and by splitting the data into training, testing and validation in a 70/15/15 ratio. The results are as follows: The model used for the N output had a training MSE of 1.63×10^{-4}, a validation MSE of 2.34×10^{-4} and a testing MSE of 1.93×10^{-4}. The model used for the V output had a training MSE of 8.42×10^{-4}, a validation MSE of 1.017×10^{-4} and a testing MSE of 8.59×10^{-4}. The model used for the T output had a training MSE of 1.30×10^{-3} , a validation MSE of 1.46×10^{-3} and a testing MSE of 1.60×10^{-3}.

In Figures 6.4, 6.5 and 6.6 we can see 100 of the 250 epochs of training of the *N*, *V* and *T* models whilst also observing the absolute error in the predictions of the training phase for each of the models.

Analysing both the obtained results and the illustrative graphics one can note that the MSEs tend to be lower when using the dropout layer. We can also see that not using dropout achieves an optimal performance sooner, Even though the training phase is not as smooth using a dropout layer, because it avoids overfitting to the training data the validation becomes better than when we don't use the dropout technique.



(a) Epochs' History

(b) Absolute Error

Figure 6.3: T model training history and absolute error in predictions using a dropout layer. A diminishing MSE can be observed through the training epochs and the validation MSE stands in line with training MSE for most of the time.

On another note, we can also observe that there are some peaks potentially caused by outliers in our validation and training history in both models, especially when considering the V output.

We then proceeded by repeating the experiments on the datasets with no outliers so as to verify the impact these might have on the final results. We decided to use the dropout layer given the reasons mentioned before.

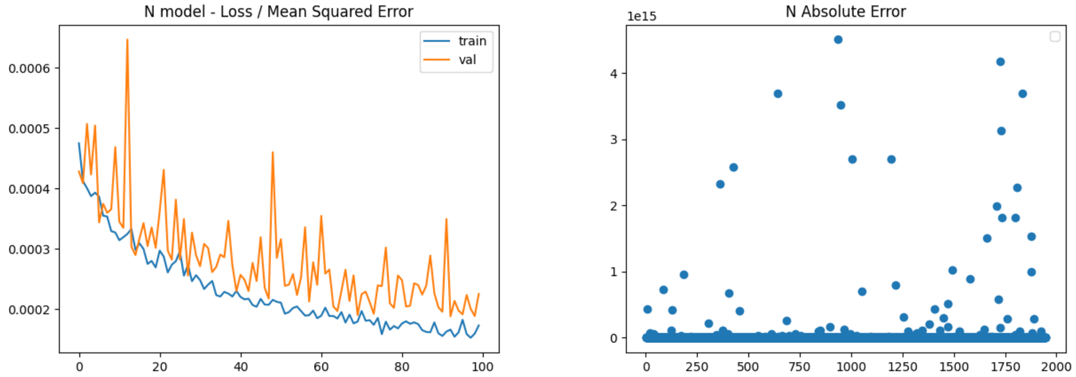
The results for the experiments with no outliers were as follows: The model used for the N output had a training MSE of 1.79×10^{-4}, a validation MSE of 1.90×10^{-4} and a testing MSE of 1.78×10^{-4}. The model used for the V output had a training MSE of 4.84×10^{-4}, a validation MSE of 6.04×10^{-4} and a testing MSE of 6.45×10^{-4}. The model used for the T output had a training MSE of 1.65×10^{-3} , a validation MSE of 1.99×10^{-3} and a testing MSE of 2.31×10^{-3}.

In Figures 6.7, 6.8 and 6.9 we can see 100 of the 250 epochs of training of the N, V and T models whilst also observing the absolute error in the predictions of the training phase for each of the models.

Analysing both the graphics and results mentioned above one can see that even though it was expected that the outliers' discarding might have a big impact in reducing the MSE obtained especially when considering output T, such is not the case. Not only has the MSE not reduced but it has even augmented. This tells us that the outliers are important for the physical problem at hand, happening more times with physical significance than what seems to be the case at first glance.

6.1.1 Random-Based and Median-Based Models

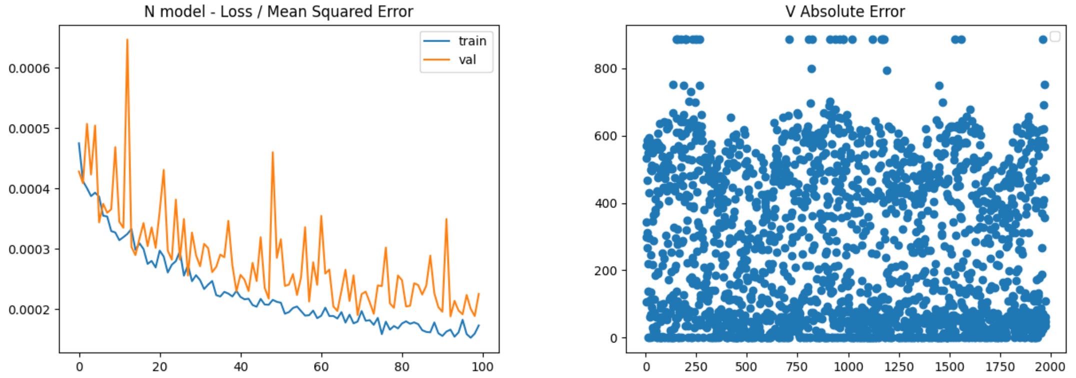
After this analysis we decided to make sure that our models outperformed the results given by a random-based and a median based model.



(a) Epochs' History

(b) Absolute Error

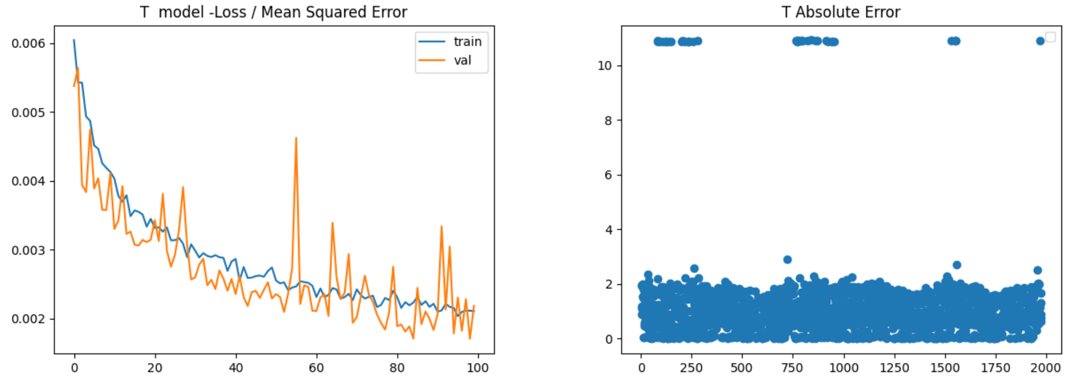
Figure 6.4: N model training history and absolute error in predictions not using a dropout layer. A diminishing MSE can be observed through the training epochs and the validation MSE stands above training MSE for most of the time.



(a) Epochs' History

(b) Absolute Error

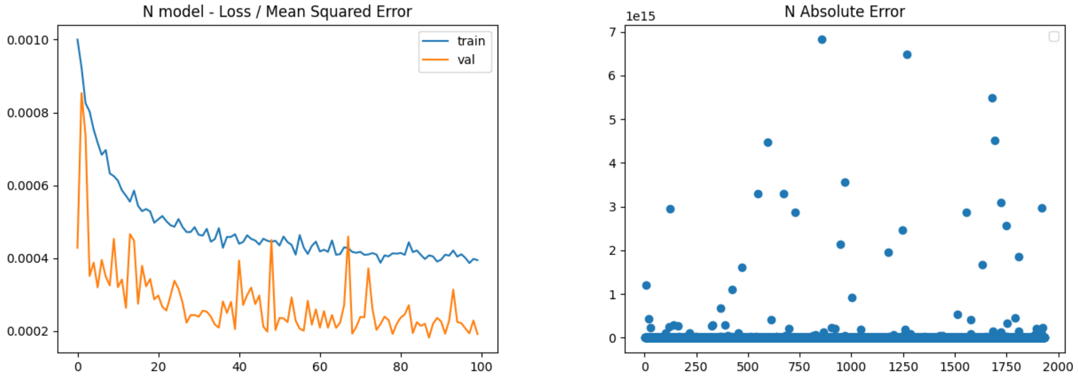
Figure 6.5: V model training history and absolute error in predictions not using a dropout layer. A diminishing MSE can be observed through the training epochs and the validation MSE stands above training MSE for most of the time.



(a) Epochs' History

(b) Absolute Error

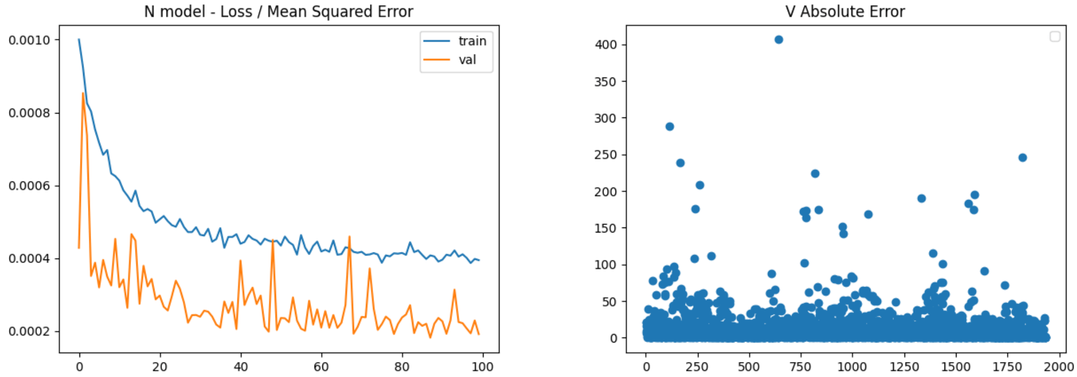
Figure 6.6: T model training history and absolute error in predictions not using a dropout layer. A diminishing MSE can be observed through the training epochs and the validation MSE stands in line with training MSE for most of the time.



(a) Epochs' History

(b) Absolute Error

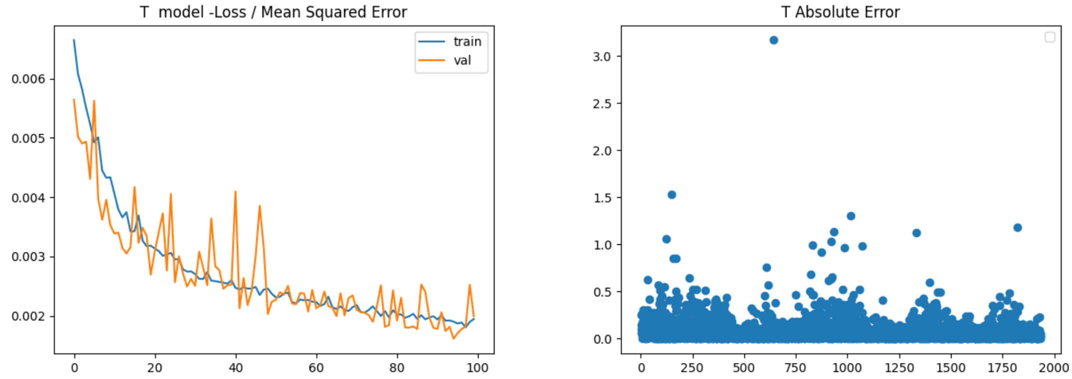
Figure 6.7: N model training history and absolute error in predictions using a dropout layer and after outliers discarding. A diminishing MSE can be observed through the training epochs and the validation MSE stands below training MSE for most of the time.



(a) Epochs' History

(b) Absolute Error

Figure 6.8: V model training history and absolute error in predictions using a dropout layer and after outliers discarding. A diminishing MSE can be observed through the training epochs and the validation MSE stands below training MSE for most of the time.



(a) Epochs' History

(b) Absolute Error

Figure 6.9: T model training history and absolute error in predictions using a dropout layer and after outliers discarding. A diminishing MSE can be observed through the training epochs and the validation MSE stands in line with training MSE for most of the time.

Table 6.2: Different Models MSEs comparison on testing data. A comparison between the developed models with and without outliers is made against a random based and a median based model. The developed models show an out-performance based on MSE comparison when compared to the random and median based models.

Output	Outliers model test MSE	No Outliers model test MSE	Random model test MSE	Median model test MSE
N	1.58×10^{-4}	1.783×10^{-4}	1.65×10^{-1}	3.33×10^{-1}
V	5.99×10^{-4}	6.45×10^{-4}	1.63×10^{-1}	2.34×10^{-1}
T	1.39×10^{-3}	2.31×10^{-3}	1.67×10^{-1}	1.82×10^{-1}

The random model consisted of generating random values with the range of the corresponding outputs. The median model consisted of generating the median of the range of values with the range of the corresponding outputs. We then compared the results of testing amongst the different used models. The results were as follows.

Considering the median models, the MSE obtained for testing using model N was of 3.33×10^{-1}. For model T the testing MSE was of 2.34×10^{-1} and the testing MSE of model T was of 1.82×10^{-1}.

Considering the random models, the MSE obtained for testing using model N was of 1.65×10^{-1}. For model T the testing MSE was of 1.63×10^{-1} and the testing MSE of model T was of 1.67×10^{-1}.

In Table 6.2 one can observe the comparison of the used model versus the random and median models. As can be observed, the models used during this work outperform significantly the random and the median model letting us know that it is possible to learn the output features at hand.

6.1.2 Real Values Graphical Comparison

After getting the MSEs for each of the models, we decided then to compare the predicted and the real outputs of 15 random supplied files. The results for the N,V and T outputs can be seen in Figures 6.10, 6.11 and 6.12 respectively.

Analysing the aforementioned Figures one can reach some conclusions. Firstly, the models which were trained containing outliers seem to better accompany the expected results. As mentioned before such is possibly true given the physical significance of such outliers being present more times than seems to be true at a first glance. The model used to predict N performs better than the ones used to predict V and T which might be explained by the rapid changes experienced by the V and T profiles. Ultimately we can infer that even though our MSE was extremely low when analysing the prediction of each line and measuring the median error of the whole file, our models are not reasonable enough at predicting the expected values.

Some reasons on why this might and that should be considered when thinking of future work might be the range of values, no access to MULTI-VP, loss metric and model data configuration for training. Explanations and future work and alternatives for these results can be found in chapter 7.

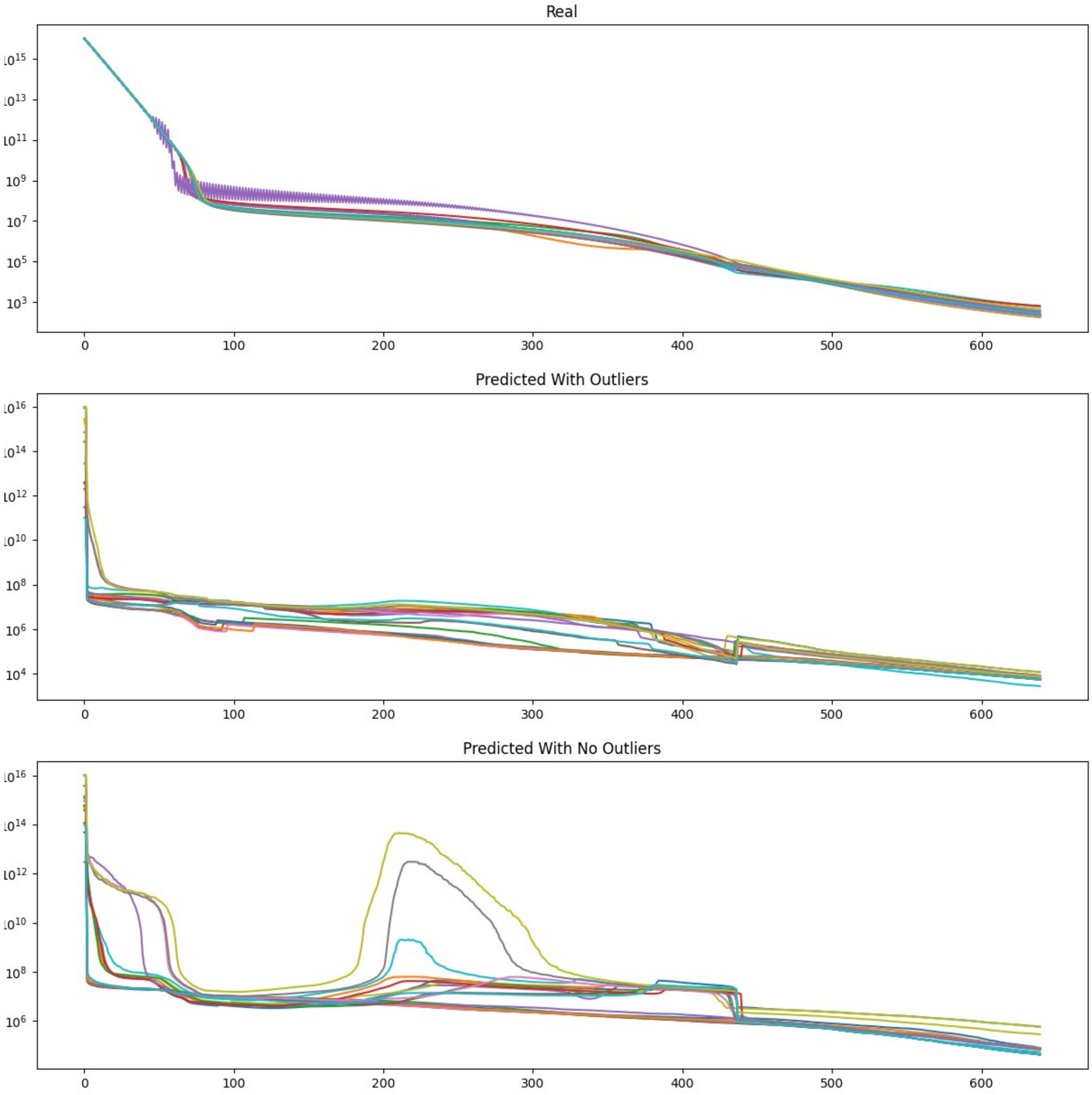


Figure 6.10: Comparison of the real expected N output with the predicted ones from both the models with with and no outliers. A clear discrepancy in values can be seen particularly when using the model trained with no outliers.

Because our results were so drastically different than expected and there were a lot of irregularities in the predicted profiles , we were not able to test our second research question regarding the reduction of computation times. Such was due to the fact that the simulation is not capable of digesting such large radial variations.

As such we decided to test a new approach to our data preprocessing which is described in 6.2

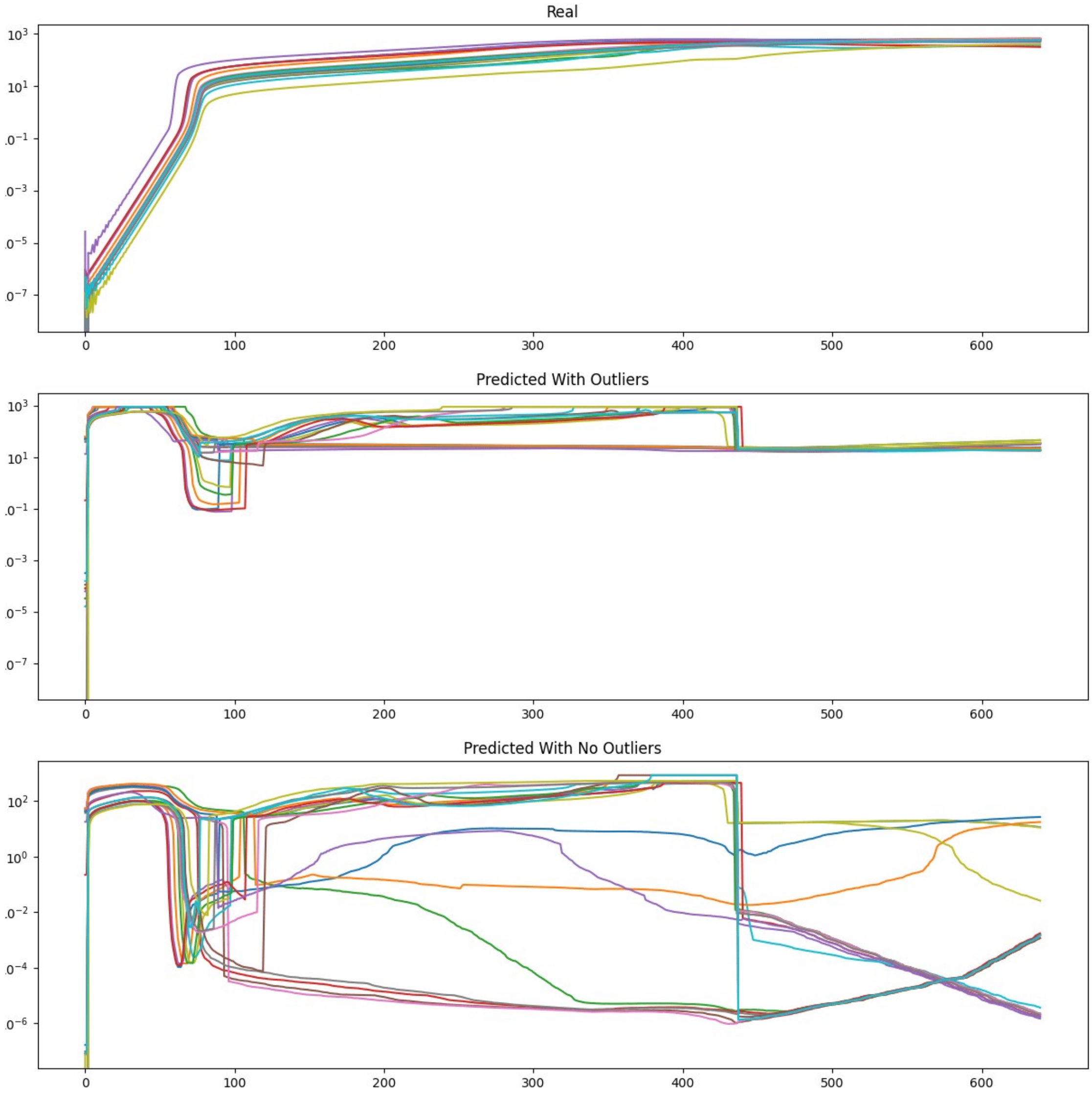


Figure 6.11: Comparison of the real expected V output with the predicted ones from both the models with with and no outliers. A clear discrepancy in values can be seen particularly when using the model trained with no outliers.

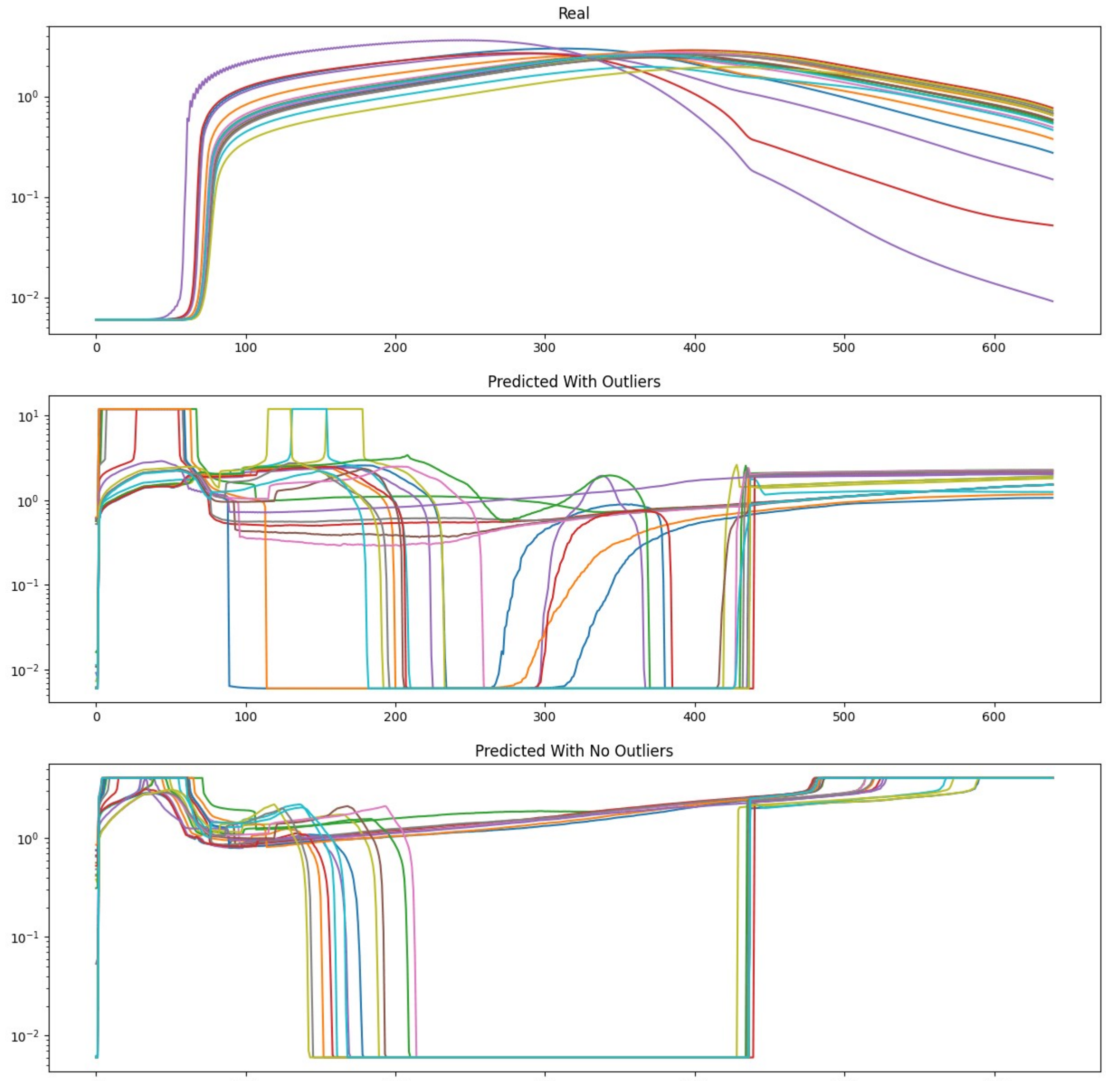


Figure 6.12: Comparison of the real expected T output with the predicted ones from both the models with with and no outliers. A clear discrepancy in values can be seen particularly when using the model trained with no outliers.

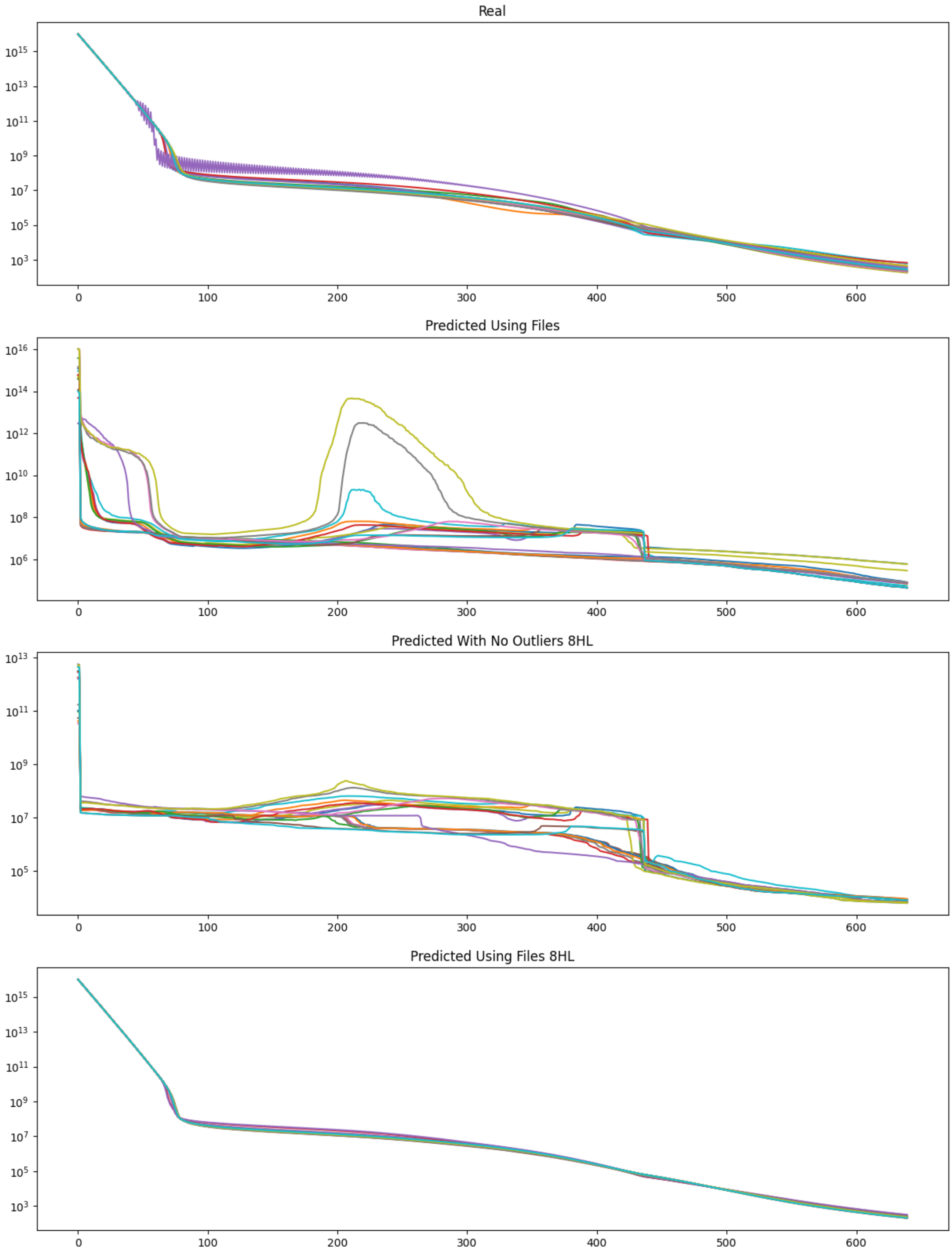


Figure 6.13: Comparison of the real expected N output with the predicted ones from the model using files, the model with outliers using more hidden layers and the model using files with more hidden layers. A clear improvement in the predicted values can be seen especially in the last model.

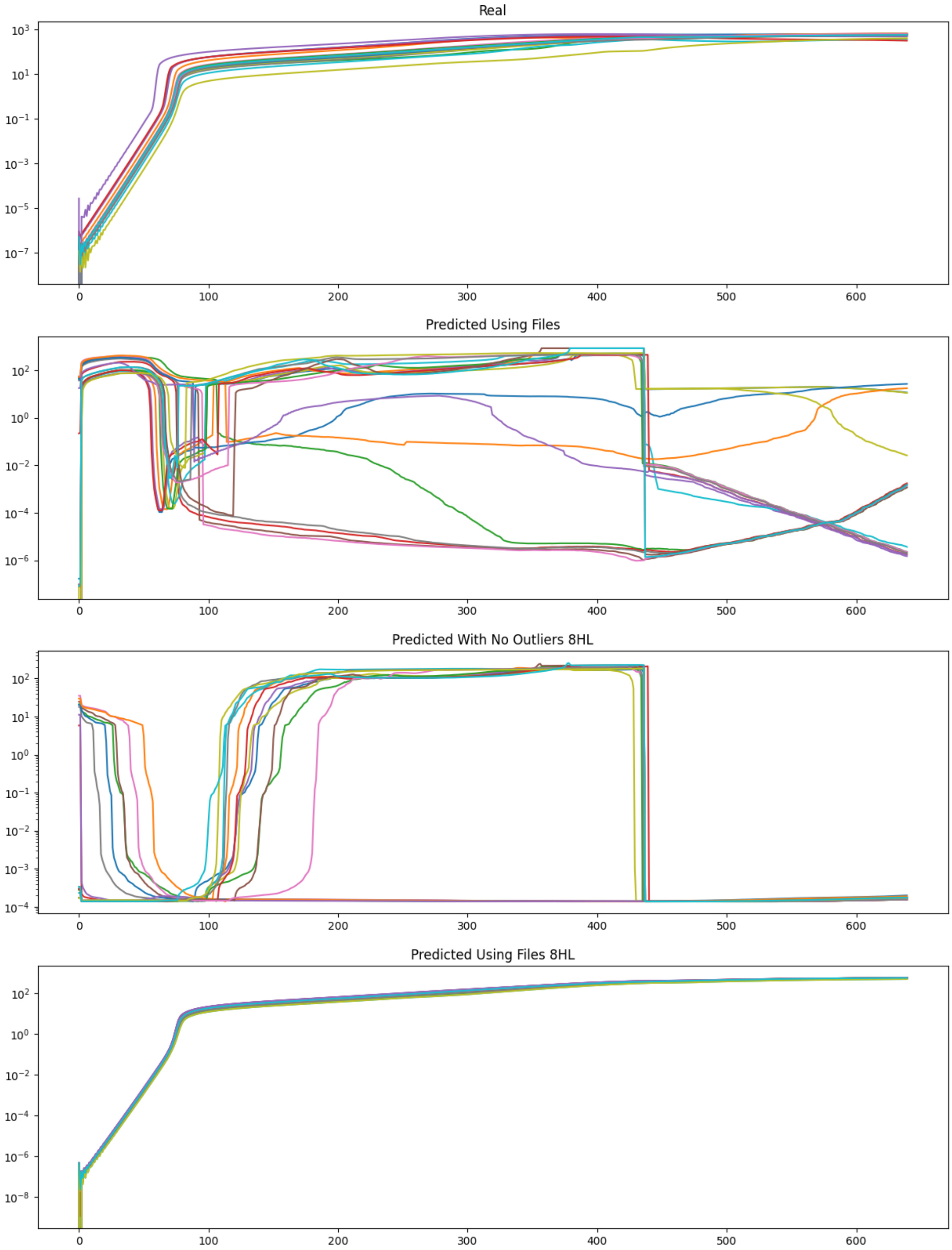


Figure 6.14: Comparison of the real expected V output with the predicted ones from the model using files, the model with outliers using more hidden layers and the model using files with more hidden layers. A clear improvement in the predicted values can be seen especially in the last model.

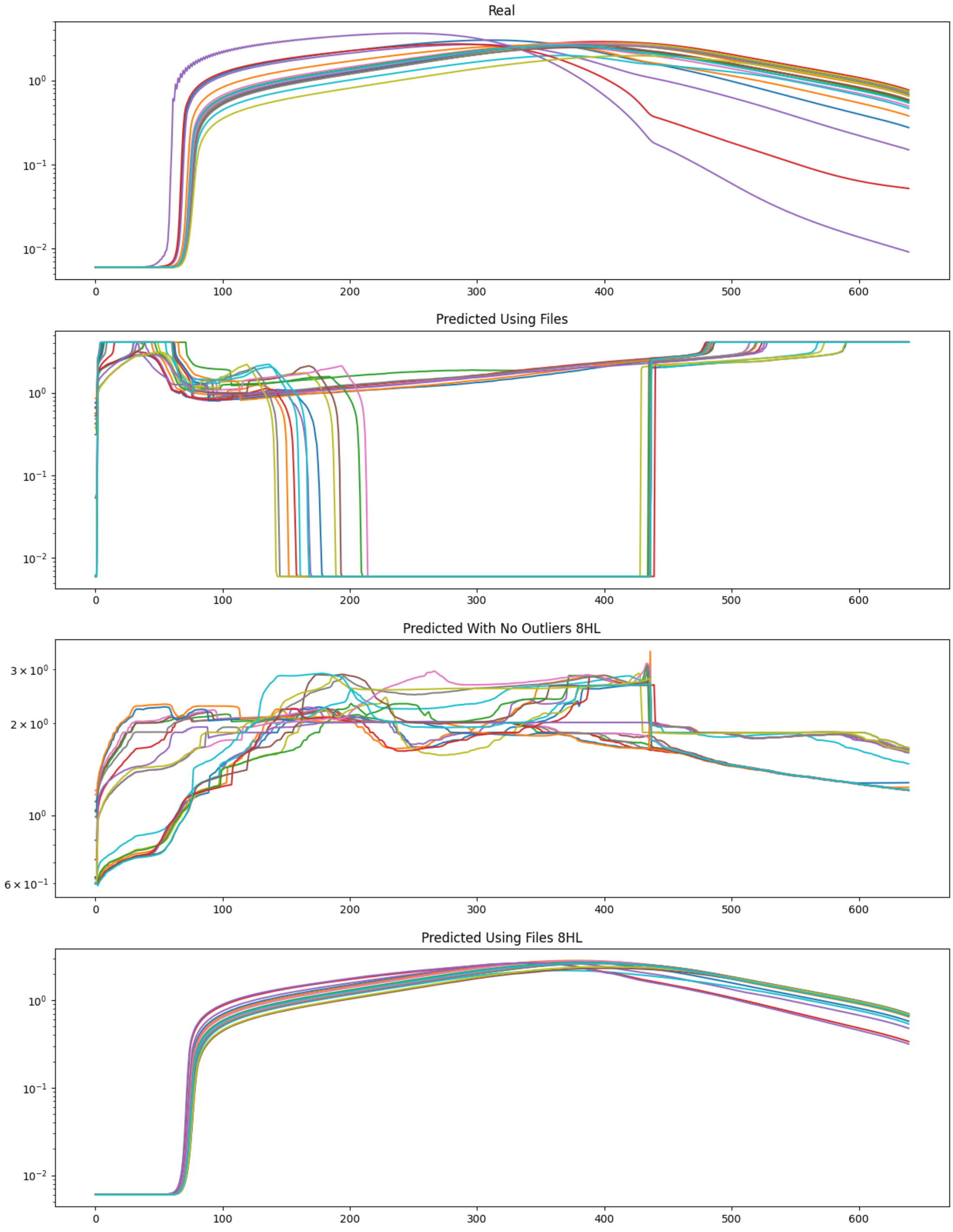


Figure 6.15: Comparison of the real expected T output with the predicted ones from the model using files, the model with outliers using more hidden layers and the model using files with more hidden layers. A clear improvement in the predicted values can be seen especially in the last model.

6.2 Flow-Based Approach

Instead of sampling one random line from each provided file we decided to train a new model consisting of the same hidden layers and nodes mentioned before for each output but, instead of feeding the lines to it we would use each file as an input. To do so, we used 6000 of the provided files and trained the models for 500 epochs. The results for the experiments with the files as input using two hidden layers like before were as follows: The model used for the N output had a training MSE of 2.61×10^{-2}, a validation MSE of 2.87×10^{-2} and a testing MSE of 2.87×10^{-2}. The model used for the V output had a training MSE of 2.16×10^{-2}, a validation MSE of 2.17×10^{-2} and a testing MSE of 2.17×10^{-2}. The model used for the T output had a training MSE of 1.93×10^{-2}, a validation MSE of 1.92×10^{-2} and a testing MSE of 1.93×10^{-2}. However, as can be seen in Figures 6.13, 6.14 and 6.12, this approach did not improve our previous results.

By interpreting such results, we started questioning whether the complexity of the Neural Networks being used was not high enough to find relations between the different features. We decided then to test if a higher complexity of ANNs would improve our results and incremented the number of hidden layers to 8. The model used for the N output had a training MSE of 4.85×10^{-2}, a validation MSE of 4.72×10^{-2} and a testing MSE of 4.7164×10^{-2}. The model used for the V output had a training MSE of 1.01×10^{-1}, a validation MSE of 1.02×10^{-1} and a testing MSE of 1.02×10^{-1}. The model used for the T output had a training MSE of 4.45×10^{-2}, a validation MSE of 4.51×10^{-2} and a testing MSE of 4.51×10^{-2}. Even-though the MSEs augmented when compared to the last model, this improved the results by a lot which can be seen in the last sub-Figures of Figures 6.13, 6.14 and 6.12.

We decided then to go back and test if our k-fold validation had hit some kind of local minima. To do so we tested the no outliers model with eight hidden layers and verified the there wasn't a considerable improvement in most layers. This helped us conclude that using the files as input instead of the lines was the determining factor in improving our results.

6.3 Simulation Runs

Contrary to the results in subsection 6.1.2, the flow-based approach results did not present large radial variations. As such, we were able to test 15 randomly selected predicted files in MULTI-VP.

Taking a first glance at the runs' results in Figure 6.16, one can observe that convergence is done correctly, meaning we get the same final result, and most profiles are closer to the final solution. It can be seen that the convergence process is a little more straightforward in most cases, with a smoother initial transient. In particular, the density profile generally seems closer to the final solution than the standard profiles.

We proceeded by performing a convergence test. The results of such an experiment can be found in Figure 6.17, that shows the temporal evolution of the relative variations of each quantity between two consecutive data outputs on each run, i.e., the ratio $(X_i - X_{i-1})/X_{i-1}$ for any given

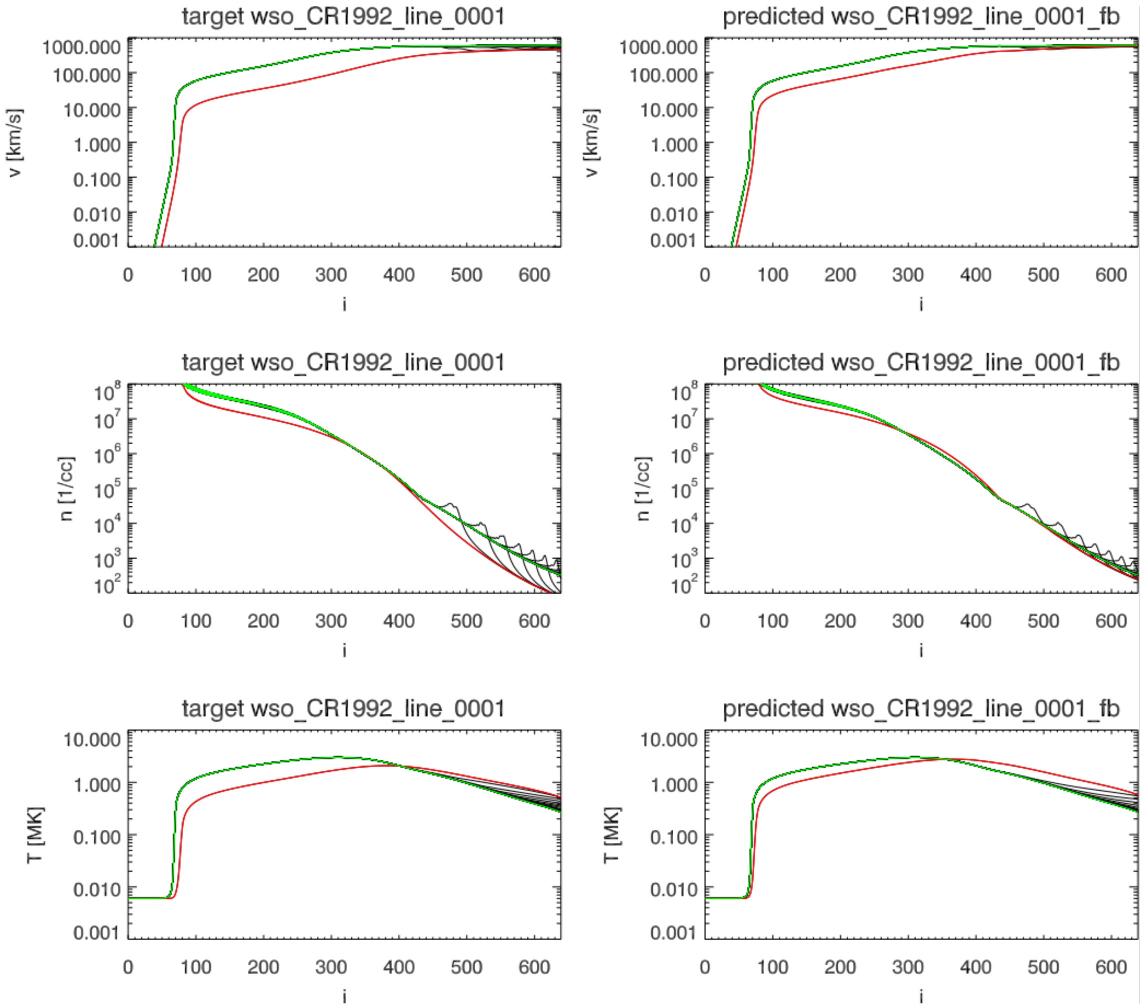


Figure 6.16: Illustrative example of comparison of standard and predicted profiles performance using one of the 15 predicted profiles. The standard profile can be seen on the left panels, while the new profile can be seen on the right panels. Red lines represent the initial condition, green lines represent the final solutions, and black lines represent some intermediate instants, with the same delta t on both sides.The abscissa represents the grid-point number.

quantity X. Figure 6.17 shows that the predicted profiles often converge faster than the standard profiles.

The initial transients are different, being sometimes smoother with the new predicted files. The integration time step that the code uses adapts automatically to the complexity of the simulated flow at any given instant, and therefore differs between the two realizations. That is, the number of total iterations required for the code to converge becomes different in the two cases.

As such, we used that same number of iterations, instead of elapsed time, as a measure of performance. It results in the values shown in Table C.1) that consecutively indicate the analyzed run, the convergence times for each case, the respective number of iterations, and the speedup factor. This speedup factor was calculated by using the simple formula speed_fac = standard_number_iterations/predicted_number_iterations.

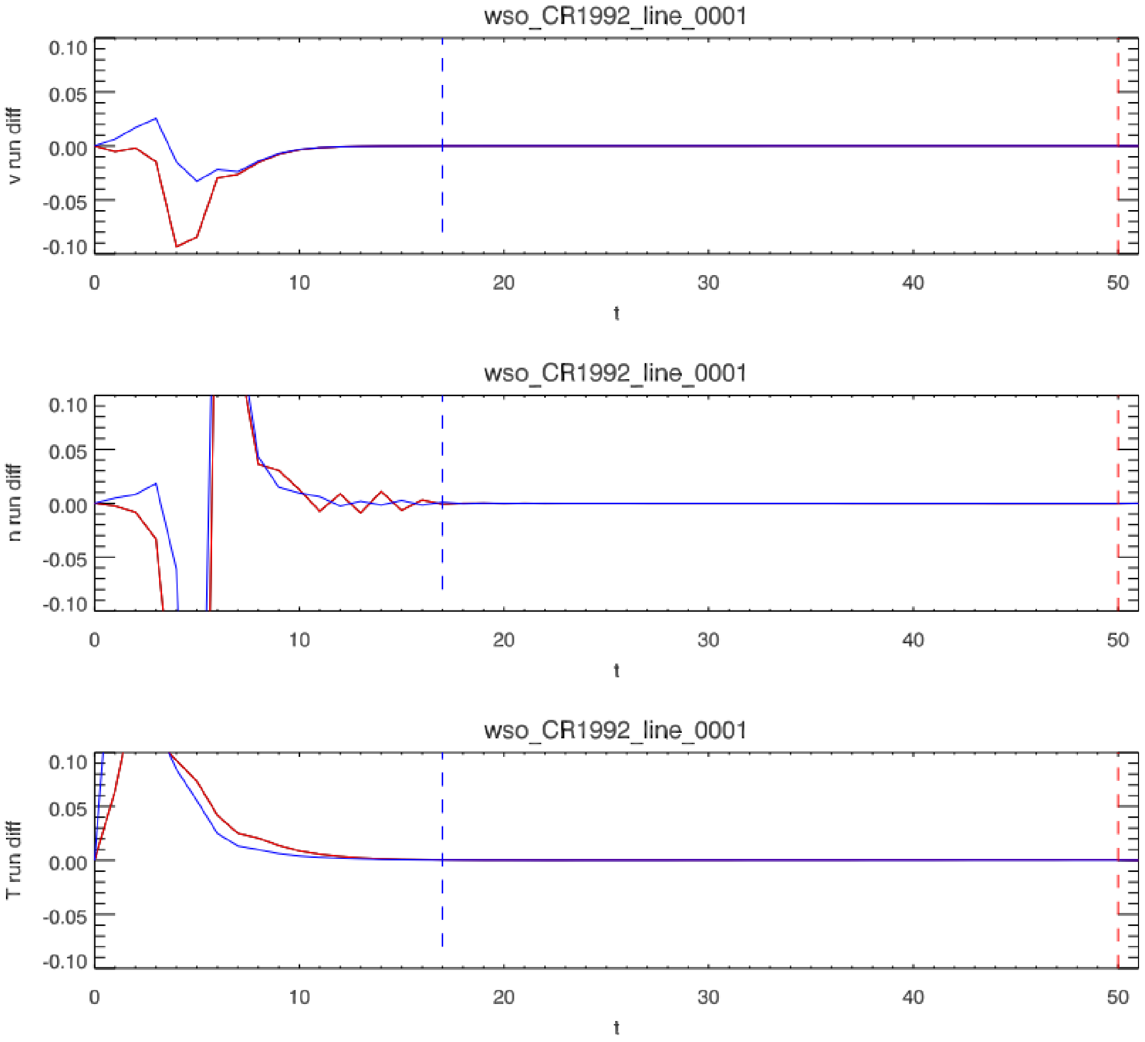


Figure 6.17: Illustrative example of comparison concerning convergence using one of the 15 predicted profiles. Each graph shows the relative variations measured at a reference altitude(1/10 of the Sun-Earth distance) between successive outputs of the code ("running differences") as a function of time, the standard profiles are represented in red and the predicted ones in blue. First for speed, then density and lastly temperature. The abscissa shows the elapsed time (in code units, equivalent to 1.5 h of physical time). The vertical dashes show the instant at which convergence was detected, defined as a threshold of relative variations for the three quantities (all three must oscillate less than the threshold value).

We then tested the same runs and same diagnostics, but with a 10x higher data output rate to better see the relaxation/convergence process. Relaxation time detections are more accurate this way. As one can see in Figure 6.18, there is a series of oscillations at well-defined frequencies that follow the initial impulse. This does not represent numerical noise: it is a well-defined oscillatory mode (which corresponds to an acoustic-type compressible mode or, more precisely, to the slow MHD mode) that is excited as a secondary response to perturbations in the structure of the transition region between the chromosphere and the solar corona (ie, between the cold/dense and hot/lightly dense parts of the atmosphere). One could think of this as a canonical impulse/response

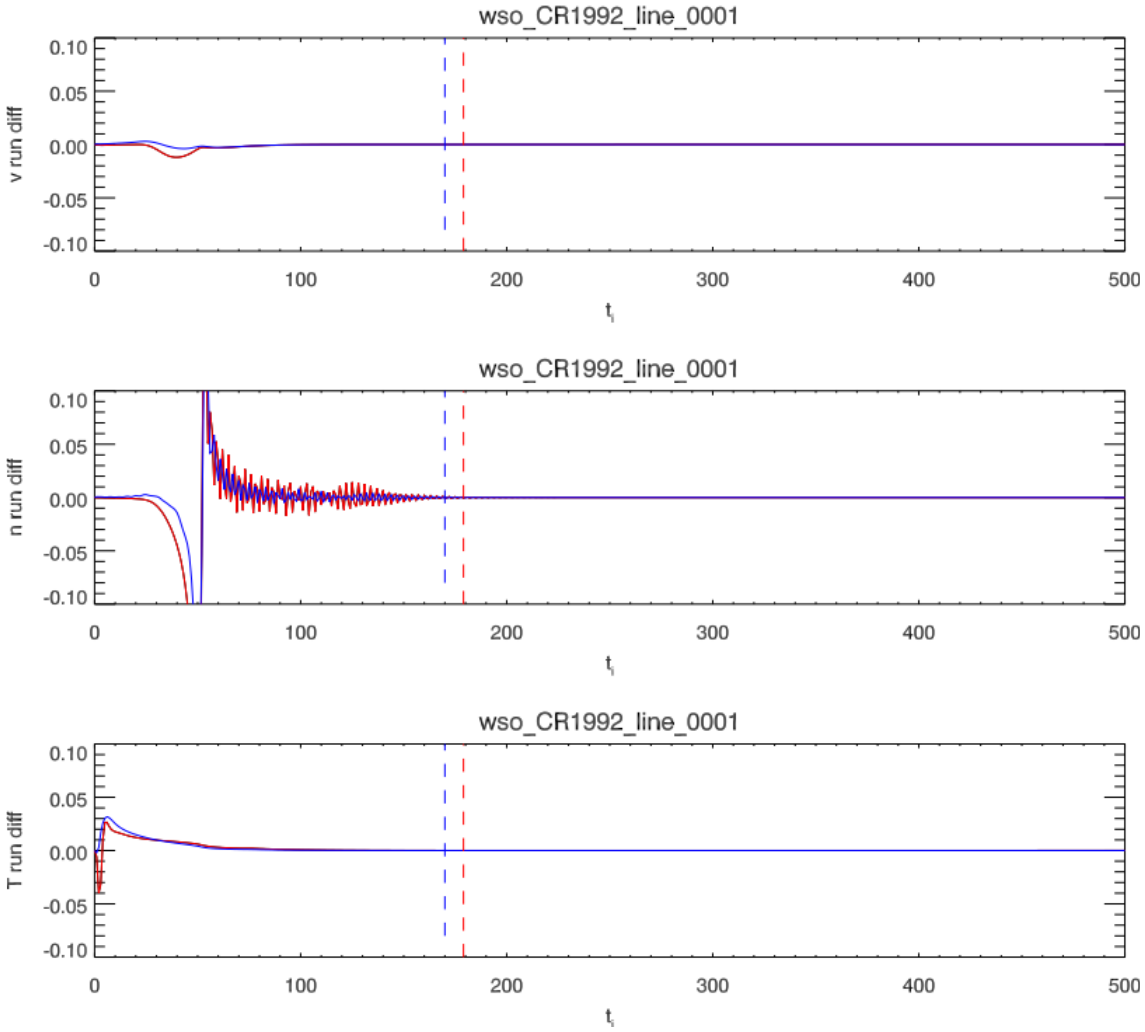


Figure 6.18: Illustrative example of comparison concerning convergence with a 10 times higher output data rate, using one of the 15 predicted profiles. Each graph shows the relative variations measured at a reference altitude between successive outputs of the code ("running differences"), the standard profiles are represented in red and the predicted ones in blue. First for speed, then density and lastly temperature. The abscissa shows the elapsed time (in code units). The vertical dashes show the instant at which convergence was detected, defined as a threshold of relative variations for the three quantities (all three must oscillate less than the threshold value).

process (although there is an intermediate process). The abcissa is indicated using a code time unit 10x smaller than on the previous figure, such that a $\Delta t_i = 10$ now corresponds to 1.5 h of simulated physical time.

The blue curves (cases with the predicted input files) generally seem to generate lower amplitude transients, which is twofold beneficial from a numerical point of view because it makes the calculation more robust (less likely to produce transients excessively stiff for the code), and because it allows for the code to keep a more moderate integration time step.

The results of this experiment can be found in Tables D.1. Analysing the referred table and performing a simple mean of the speedup we came to a mean speedup of 1.13.

It is shown that in the majority of cases, there is a small but existing improvement in the speedup of convergence times. However, to better understand the nature of the groups and whether or not there are statistical differences between their means, further analysis is required. To test the null hypothesis that both groups have identical means, we opted to use the Student's paired t-test. This test relies on the assumption of a normal distribution of the population and equal variances, although multiple studies have shown it is fairly robust to violations of at least one assumption and is able to handle small sample sizes [79].

We used the Shapiro-Wilk test with $\alpha = 0.05$ to check for distribution normality of the measured effect and obtained a $p - value$ of 0.903 meaning that we cannot reject the hypothesis that the distribution is normal. We then performed a Student's paired t-test and obtained a $p - value$ score of 0.01306 allowing us to reject the null hypothesis as the result is significant at $p < 0.05$. This supports our hypothesis that there is a statistically significant effect when using the predicted profiles.

6.4 Validation Threats

Campbell *et al.* [80] [81] define internal validation (cause and effect) as validation referring to whether an experiment makes a difference or not, and whether there is sufficient evidence to support the claim. External validity (generalization), however, refers to the generalizability of experiment outcome.

We identify the following threats to the validity of our results:

Data Integrity, Representativeness and Bias because MULTI-VP uses large volumes of data, the dimensionality of the ML modeling features make it challenging to ensure data's integrity and representativeness. Because our data is manually selected by humans there is also a tendency for the data to be biased. Another important matter is that the data comes pre-processed which might imply some noise or over processing of these (external).

Explainability Challenges Machine learning models (especially neural network-based models) are difficult to explain and are often viewed as black boxes. Assessment of the variable selection process and explainability of driving factors become difficult due to the complexity and architecture of neural networks. Even if ML models perform better than traditional models, the lack of explainability may cause ML models to be restricted in use by specialized data scientists. Since the model would be used by astronomers and astrophysicists a GUI of some sort could be presented as a solution to this threat (internal).

Parameter and Method Selection Machine learning models involve scaling, normalization, parameter optimization, randomization and activation functions. ML algorithms are sensitive to the selection of these parameters and methods. The way normalization, parameter optimization and feature selection are conducted when developing ML models can impact test error estimation, in our case the MSE and the absolute error (internal).

Loss Function Selection There are many loss functions that can be used to used to compare and attest results between machine learning models. The selected loss function can directly impact and blind-sight the results when not chosen properly. In our case, MSE proved not to be the best possible choice since for the model using the files as inputs even though the MSE is better using two hidden layers, the results are clearly demonstrated to be better when using eight hidden layers (internal).

6.5 Hypothesis Evaluation and Research Questions Discussion

This evaluation process aimed to prove the hypothesis presented in section 4.2 being this

> *Neural networks can be used to shorten the computation time needed for solar wind flux-tubes simulations made by the simulator* MULTI-VP *by learning to provide good initial guesses from previous runs.*

Given the results of our experiments, we conclude that the challenges that we focused on were tackled and that evidence indicating that the hypothesis is possibly true, was collected. The resulting answers to the proposed research questions are as follows:

RQ1 *Can neural networks acquire skill in initial guess estimation of solar wind flux-tubes simulations?* We have determined that our models are better at acquiring skill in initial-guess estimation of solar wind flux-tubes simulations. However, even though skill is acquired, the metrics used for measuring such skilled proved not to be enough for us to have a usable predicted dataset. To solve these issues, we used a different approach to our models and verified that they could accurately predict the outputs.

RQ2 *Do initial guess estimations from Neural Networks improve solar wind flux-tubes simulation times?* We were able to conclude that initial guess estimations from Neural Networks improve solar wind flux-tubes simulation times. It is shown that in the majority of cases, there is a small but existing improvement in the speedup of convergence times. The results of the Student's paired t-test $p - value$ score of 0.01306 show that the results are significant at $p < 0.05$.

6.6 Implementation Details

Python has become a broadly adopted language in machine learning applications and data science. This usefulness derives fundamentally from the extensive and active ecosystem of third-party packages [82]. This language was chosen given its code readability and many Machine Learning Modules, examples, and pre-existing documentation.

6.6.1 Libraries and Modules

The most important *Python* libraries used in this work were as follows:

pickle The *pickle* module implements binary protocols for serializing and de-serializing a *Python* object structure.

pandas *pandas* is a Python package presenting fast, adaptable, and robust data structures designed to make working with data both intuitive and straightforward. It strives to be the primary high-level building block for doing practical, real-world data analysis in *Python.*

NumPy *NumPy* is the elemental package for scientific computing in *Python.* It is a *Python* library that provides a multidimensional array object, various derived objects, and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

seaborn *Seaborn* is a *Python* data visualization library based on *matplotlib.* It provides a high-level interface for drawing attractive and informative statistical graphics.

matplotlib *Matplotlib* is a comprehensive library for creating static, animated, and interactive visualizations in *Python.*

Scikit-learn *Scikit-learn* is an open-source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection and evaluation, and many other utilities.

TensorFlow *TensorFlow* is an open-source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning models and algorithms and makes them useful through a common metaphor. It uses *Python* to provide a convenient front-end API for building applications with the framework while executing those applications in high-performance *C++.*

Keras Keras is an API designed for making deep learning more accessible in *Python.* Keras follows best practices for decreasing cognitive load offering consistent and straightforward APIs, minimizing the number of user actions required for everyday use cases, and providing clear and actionable error messages. It also possesses thorough documentation and developer guides.

6.6.2 Replication Package

The replication package for this thesis can be found in https://github.com/FilipaBarros/initial-estimation-in-flux-tube-sim

To use the code, one must simply download it and run each file in the order explicit in the README file.

6.7 Summary

In this chapter, the experiments performed during this work were presented. Normalization of data was explained as well as chosen configurations were justified through loss measures (MSEs) and by the use of k-fold validation and Keras tuner. Different models with/without outliers and dropout were tested and explained and finally compared with median and random based models outperforming these. Our results were then further analyzed in comparison with the expected outputs, and even though the MSEs were very low, these proved not to be a sufficient source of validation for the expected predictions. We then developed and tested a new model by using files instead of lines as inputs and realized the results had not improved significantly. Questioning if the complexity of the developed network was high enough (since there were no improvements with more and more physically significant data being used) we tested the use of a higher level of layers on the network. With this final model, we achieved results that were more similar to those expected, and that could be used in the *MULTI-VP* simulation. We then proceeded to predict and test 15 randomly selected files with our ANN and used these predicted files on MULTI-VP comparing their performance to the one achieved by standard files. We were then able to conclude that in the majority of cases, there is a small but existing improvement in the speedup of convergence times and used the student's t-test to validate our results. Afterward, possible validation threats were discussed, and our research questions were analyzed. Finally, implementation details were given in section 6.6.

Chapter 7

Conclusions

In this chapter, the conclusions taken at the end of the developed work are presented in section 7.1. Contributions are presented in section 7.2 and future work is discussed in section 7.3.

7.1 Conclusions

Machine Learning has become a go-to approach for solving space weather problems. In our state of the art, we were able to conclude that not only was machine learning being used for predicting phenomena; it was also being used in a variety of fields to help improve physical simulations of such phenomena. The main goal of this work was to figure out if and how machine learning techniques could be used to improve *MULTI-VP*'s simulator performance. As such, we decided to use a neural network approach (since this method was one of the most commonly used) and to come to our central hypothesis described in 4.2.

The experiments performed during this work were presented in 6. Normalization was made, and configurations were chosen and justified through loss measures (MSEs) and by the use of k-fold validation and Keras tuner. Different models with/without outliers and dropout were tested and explained and finally compared with median and random based models outperforming these. Our results were then further analyzed compared to the expected outputs, and even though the MSEs were very low, these proved not to be a sufficient source of validation for the expected predictions. We then developed and tested a new model using files instead of lines as inputs and realized the results had not improved significantly. Questioning if the complexity of the developed network was high enough (since there were no improvements with more and more physically significant data being used) we tested the use of a higher level of layers on the network. With this final model, we achieved results that were more similar to those expected, and that could be used in the *MULTI-VP* simulation. We then predicted 15 randomly selected files with our ANN and used these files on MULTI-VP comparing their performance to the one achieved by standard files. We were then able to conclude that in the majority of cases, there is a small but existing improvement in the speedup of convergence times and used the student's t-test to validate our results achieving statistical significance at $p < 0.05$. Further work is discussed and can be found in section 7.3.

This dissertation contributes with a survey and analysis of machine learning techniques applied to space weather as well a new approach for generating *MULT-VP* usable predictions and a replication package to validate and use the work performed.

7.2 Contributions

During this dissertation, three main contributions were made, being these:

Literature review: An analysis was made to the state of the art regarding Machine Learning Approaches to Solar Weather

A new approach for generating predictions: Our solution encompasses the generation of predictions for initial and full output conditions of N, V, and T values.

A replication package has been developed and is currently open-source to anyone who wishes to validate our results or even to use them for their own predictions.

7.3 Future Work

The solution developed during the course of this dissertation solved the problem of using machine learning to predict *MULTI-VP* simulation values to be used as inputs. However, the implementation contains which can be expanded upon and solved in future work.

As mentioned previously in section 4.4, there are some validations threats to the way the dissertation was carried out, such as but not limited to data integrity, representation, and bias, as well as parameter and method selection and loss function selection.

To attack the issues faced with data integrity, representativeness, and bias, we propose using **preprocessing** techniques. Since some noisy data can be seen in the actual data to be predicted, one technique that might improve the results would be to apply data smoothing using a moving average. We also propose the use of a *Python* **generator** so that we are able to feed the rest of the data to the *Keras* model being used. Due to lack of memory, we are using about half the available data, and the model could much improve from the use of all the provided files. Another preprocessing technique worth mentioning for future work is the use of**feature engineering**. As stated in chapter 5 there is a high correlation between features in the dataset. One possibility would be to restructure the features to extract other features or even to remove useless features to simplify the model and provide less computation time.

Some problems mentioned before with the last model which is currently being used is the lack of **k-fold** validation and **tuning**. These could provide better insight into why the model works and if this model performance can be surpassed using a different configuration of hidden layers and nodes.

Another problem detected in chapter 6 was the inconsistency of the MSE metric being used accompanying better results. As such, future work should consist of developing a distance-based loss function to evaluate the models better.

We would also suggest experimenting the process with more predicted files by automating its prediction preventing bias in our selected files.

Finally, a new approach to the problem could be made with the use of **reinforcement learning** Using reinforcement learning whilst having access to the MULTI-VP simulation in real-time to have the simulation train the neural network model.

We can conclude that the work developed in this dissertation has space for improvement, not only in its validation but also in its optimization and enrichment.

Appendix A

Data representation

Representation of a MULTI-VP generated flow is given below in A.1.

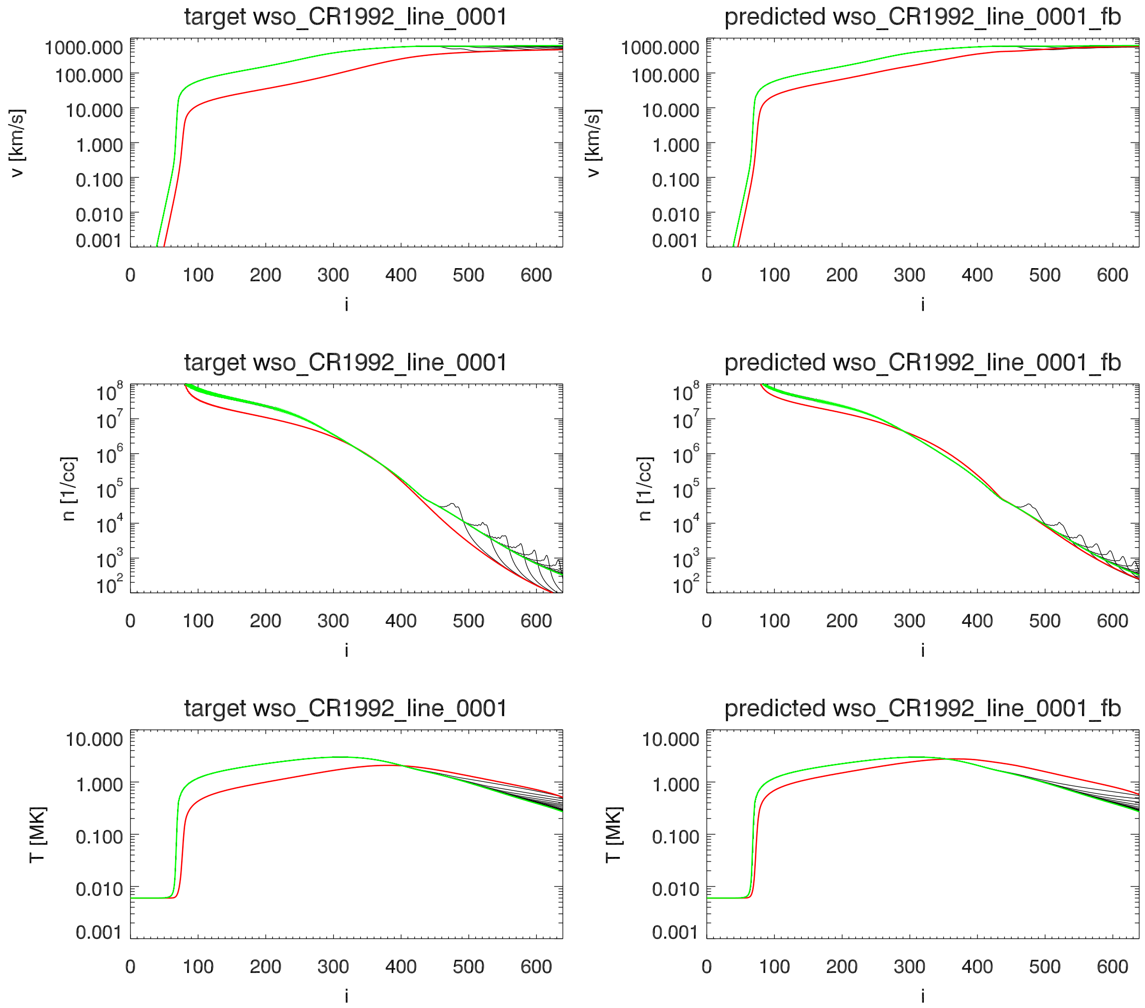
Table A.1: Data representation using one of the flows provided

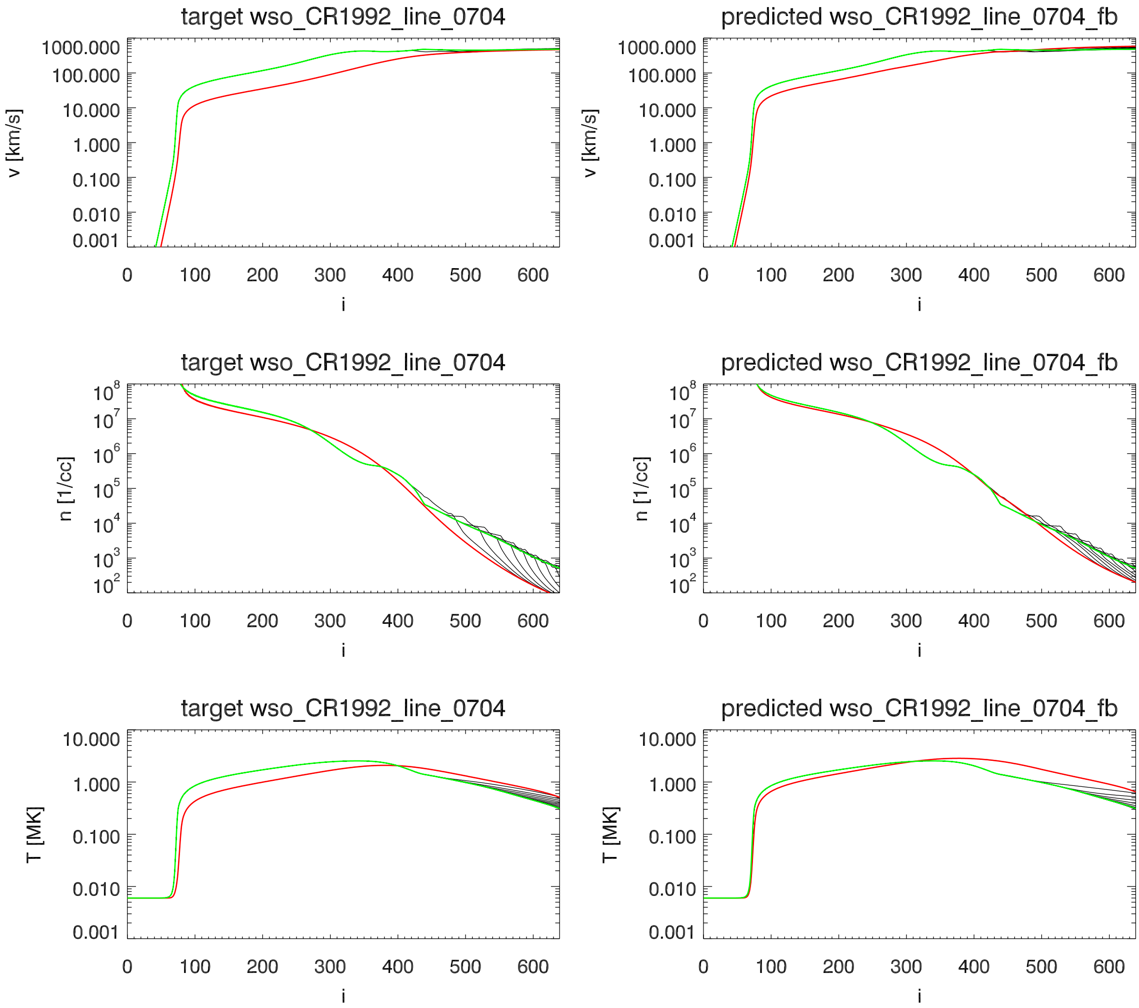
#	R [Rsun]	L [Rsun]	lon [Carr]	lat [Carr]	B [G]	A/A0	alpha [deg]	V/Cs	propag_dt [d]	n [cm^-3]	v [km/s]	T [MK]
0	1.00	1.00	340.73	-73.68	23.32	0.99	-0.14	6.17e-08	0.00	1.00e+16	7.93e-07	0.01
1	1.00	1.00	340.73	-73.68	23.313274	0.99	-0.14	3.61e-08	0.00	8.19e+15	4.64e-07	0.01
2	1.00	1.00	340.74	-73.68	23.30	0.99	-0.14	4.96e-08	0.00	6.71e+15	6.38e-07	0.01
...	...	...	...	...	...	...	...	...	...	...	...	...
638	31.16	31.24	2.49	-87.50	0.00	20641.01	0.00	7.01e+00	0.40	3.29e+02	6.13e+02	0.27
639	31.42	31.50	2.49	-87.50	0.00	20990.81	0.00	7.05e+00	0.41	3.22e+02	6.14e+02	0.27

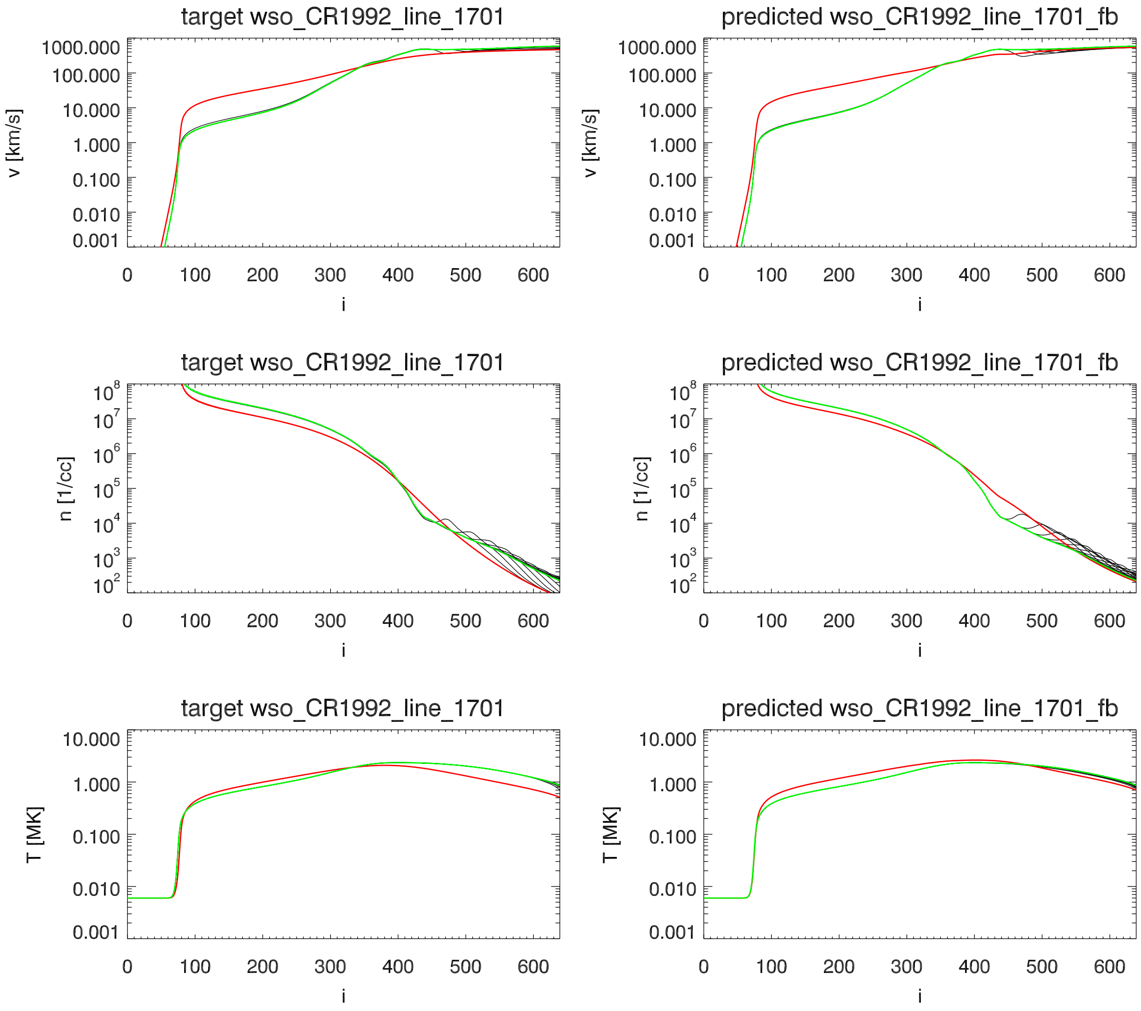
Appendix B

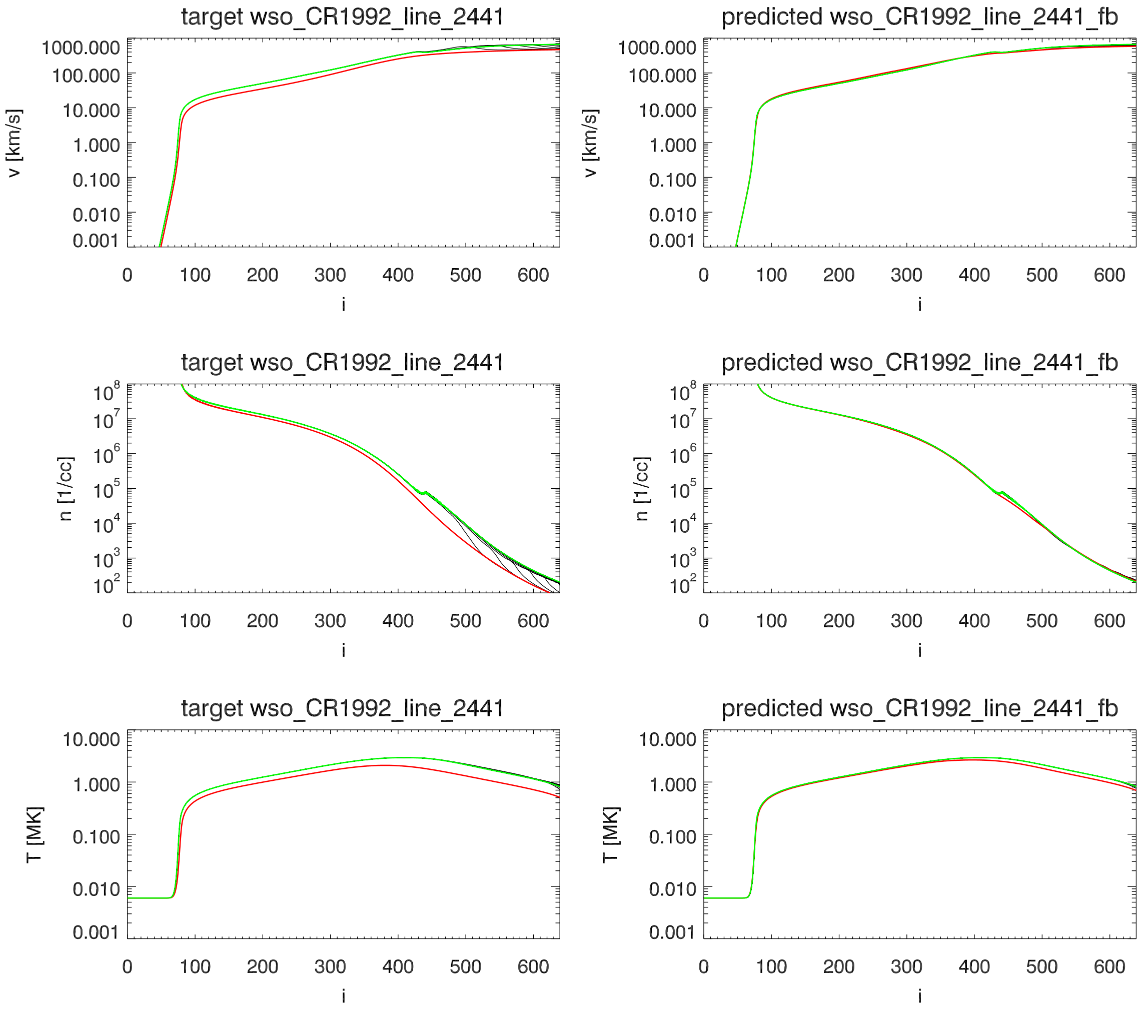
Results from MULTI_VP Runs

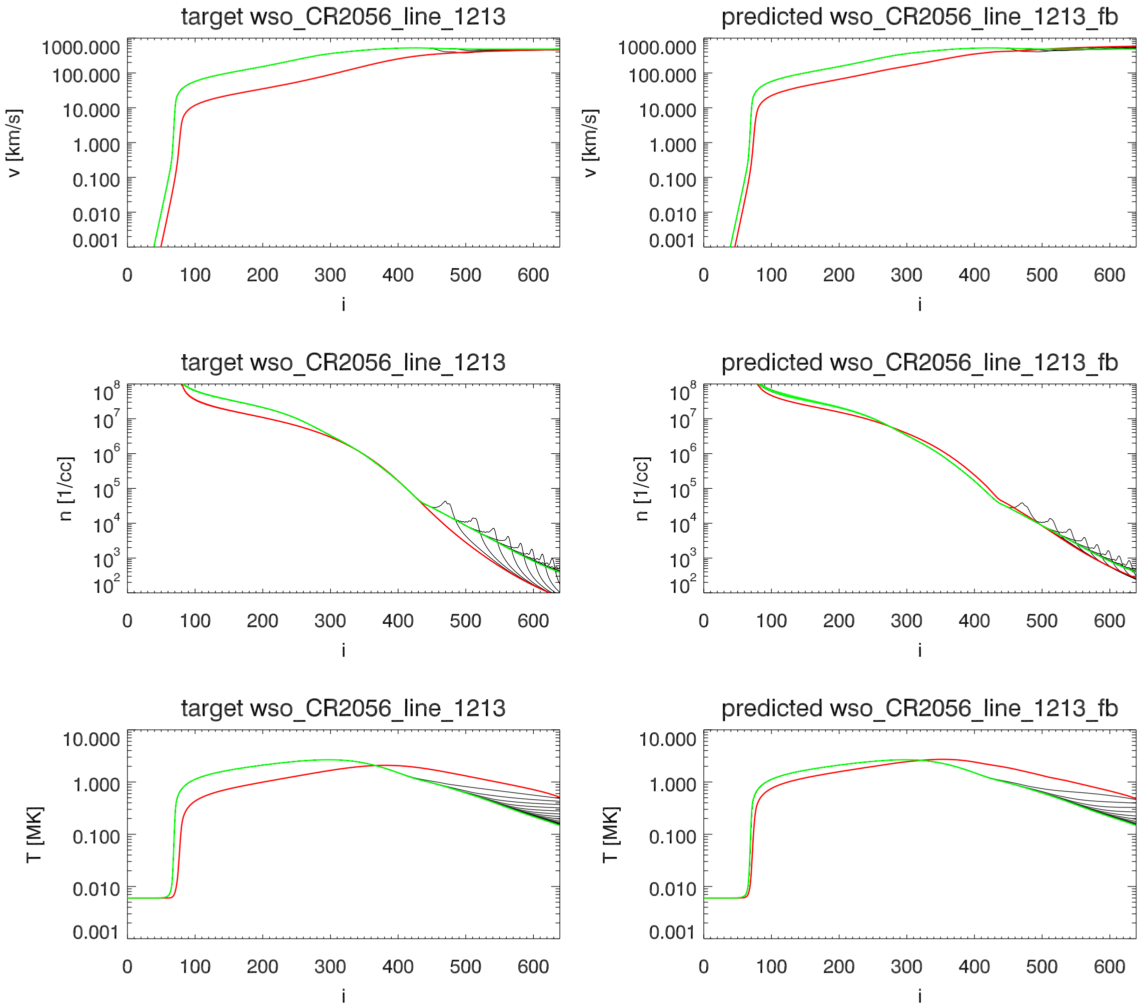
Comparison of standard and predicted profiles performance is shown bellow. The standard profile can be seen on the left while the new profile can be seen on the right. Red lines represent the initial condition and green lines, while black lines represent some intermediate instants, with the same delta t on both sides.

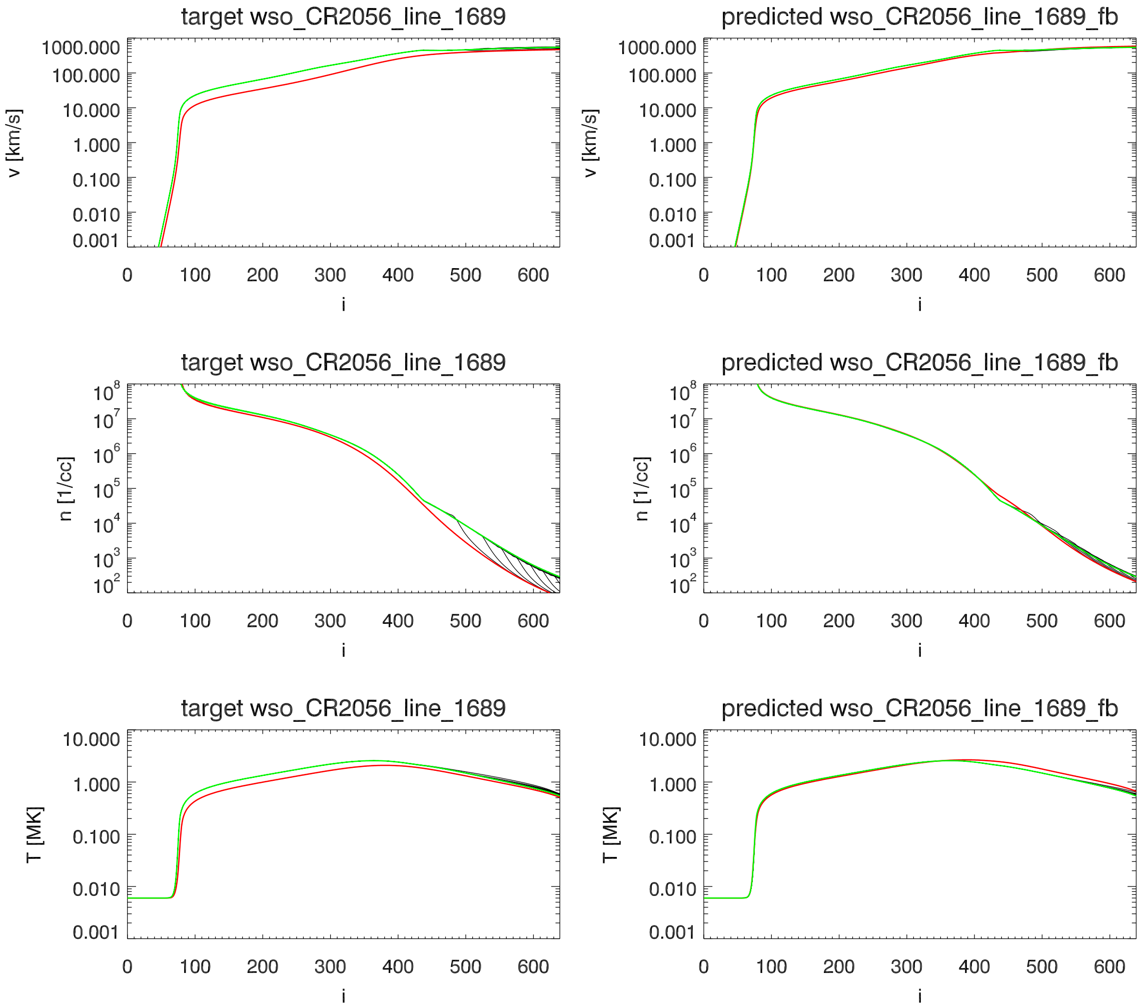

target wso_CR1992_line_0001
predicted wso_CR1992_line_0001_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
target wso_CR1992_line_0001
predicted wso_CR1992_line_0001_fb
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
target wso_CR1992_line_0001
predicted wso_CR1992_line_0001_fb
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

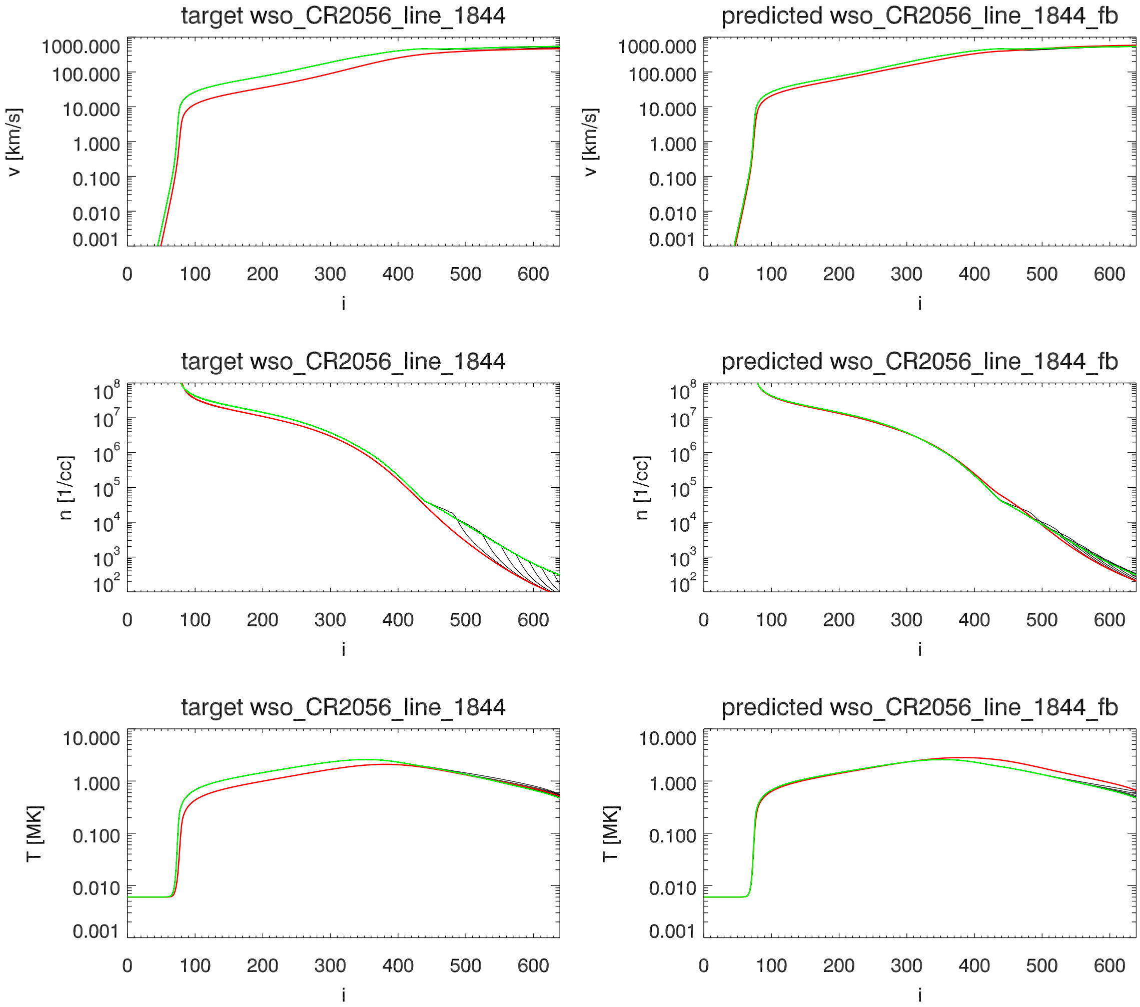

target wso_CR1992_line_0704
predicted wso_CR1992_line_0704_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

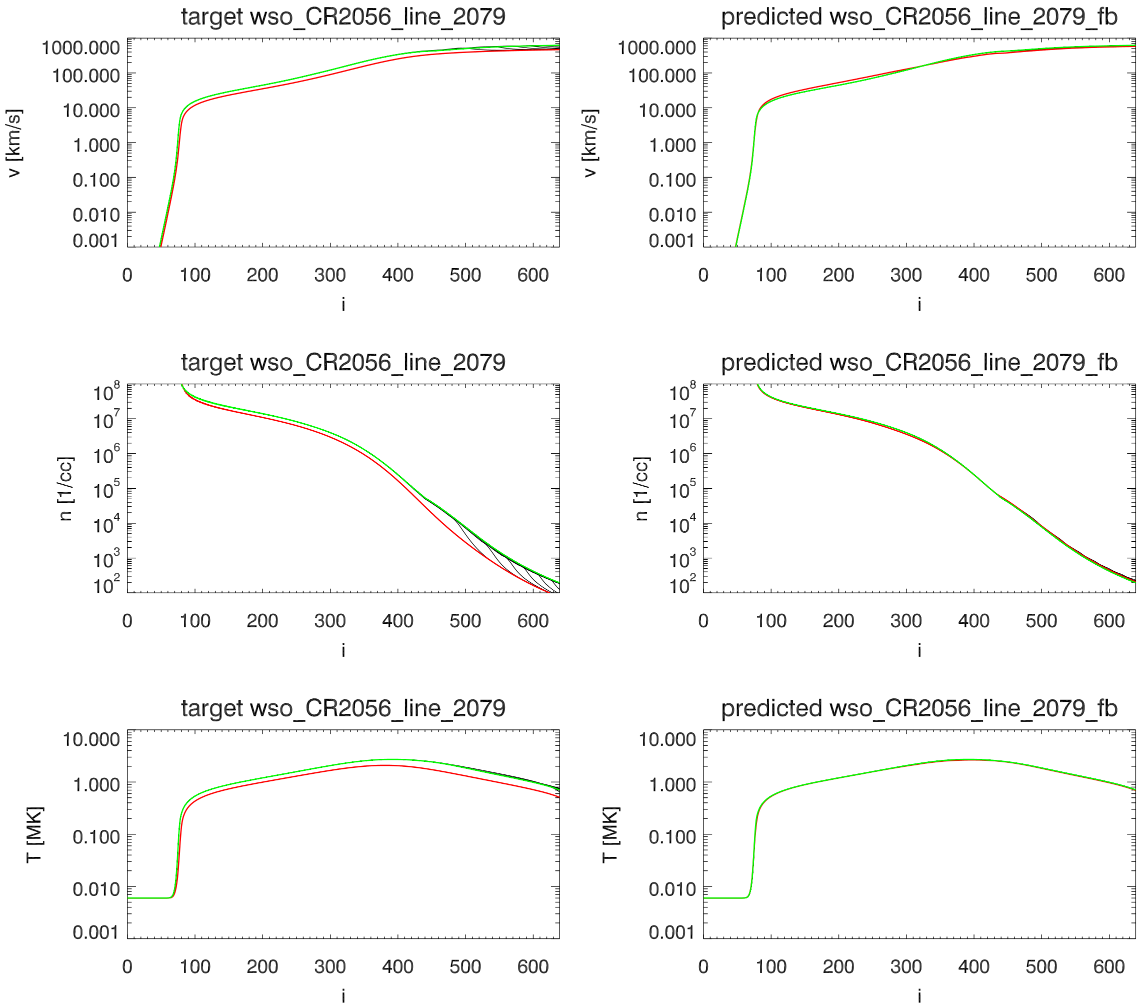

target wso_CR1992_line_1701
predicted wso_CR1992_line_1701_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

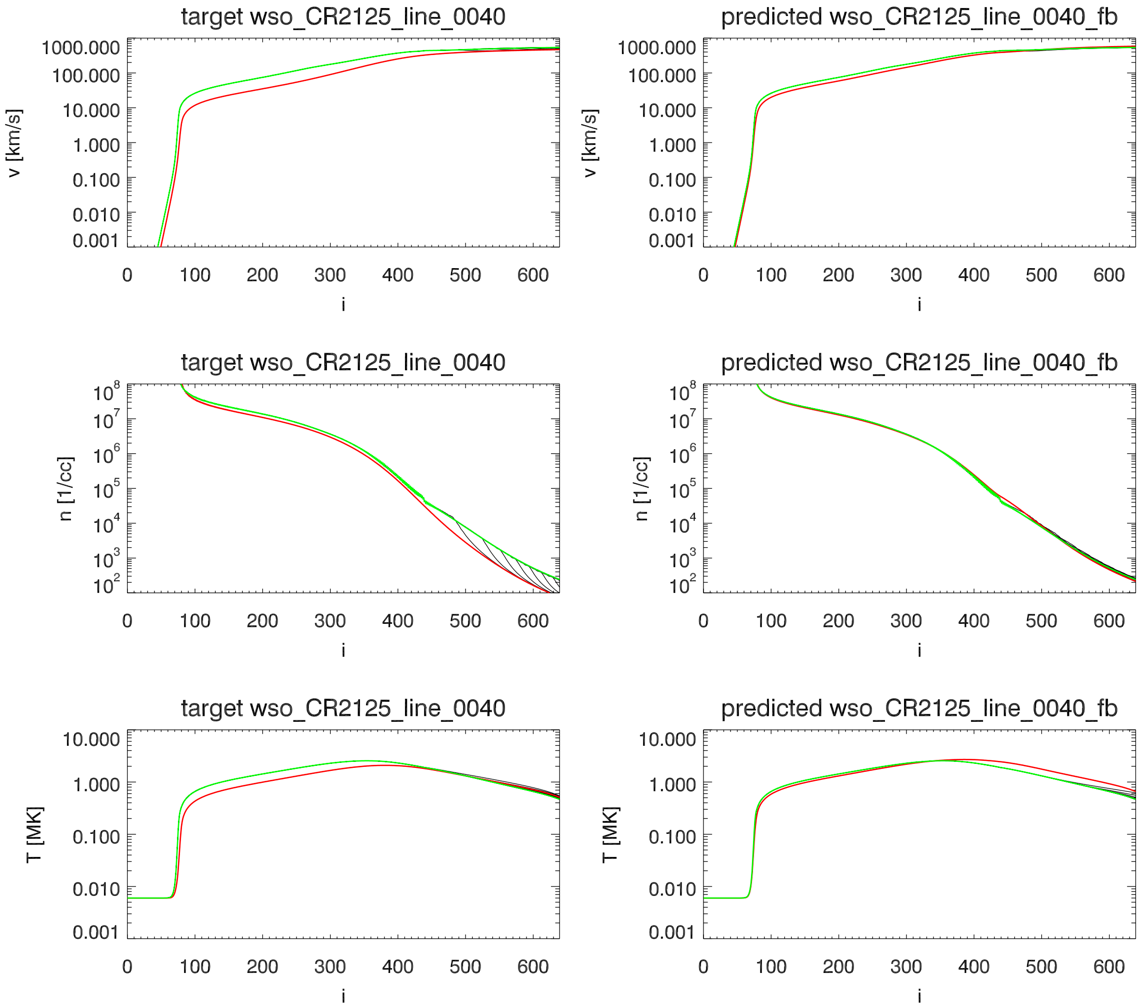

target wso_CR1992_line_2441
predicted wso_CR1992_line_2441_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

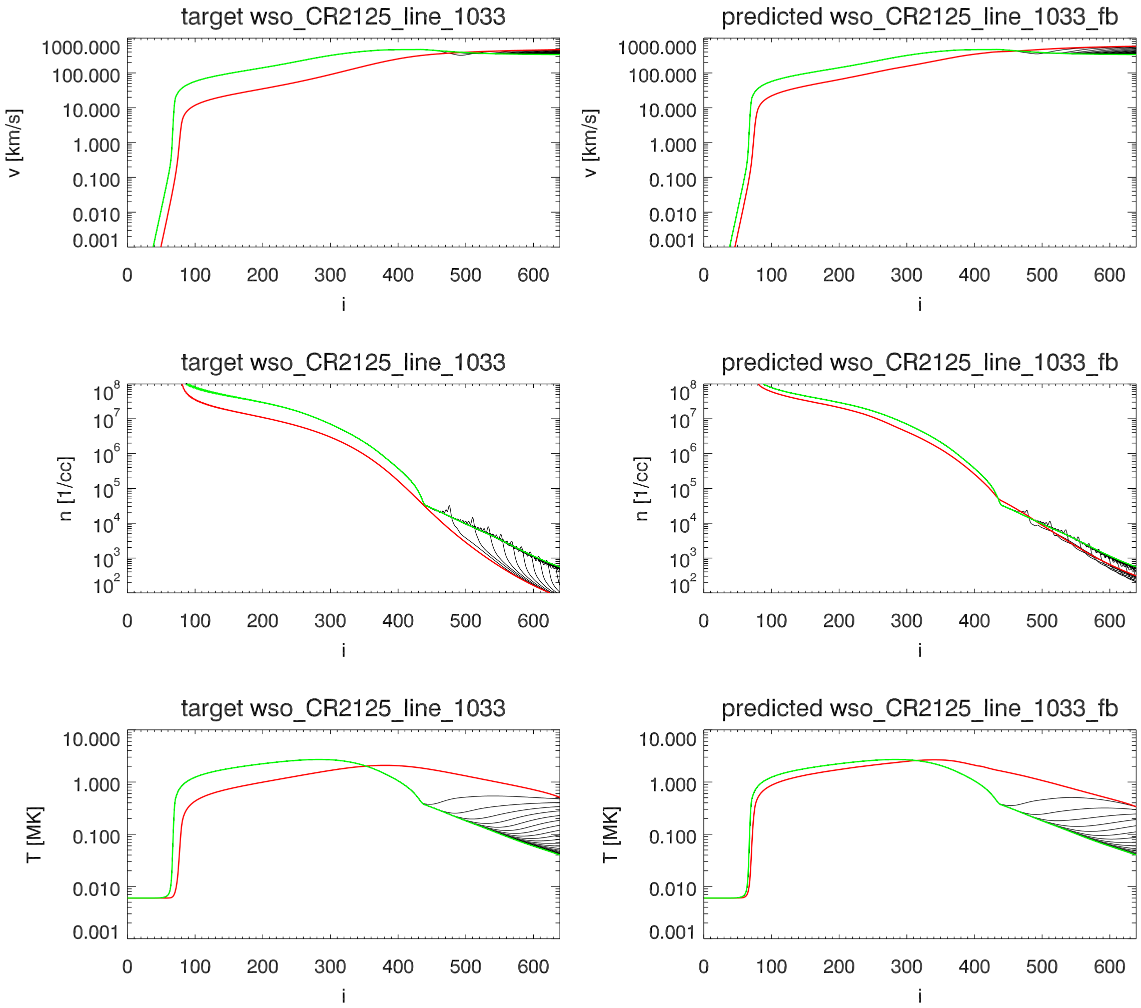

target wso_CR2056_line_1213
predicted wso_CR2056_line_1213_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

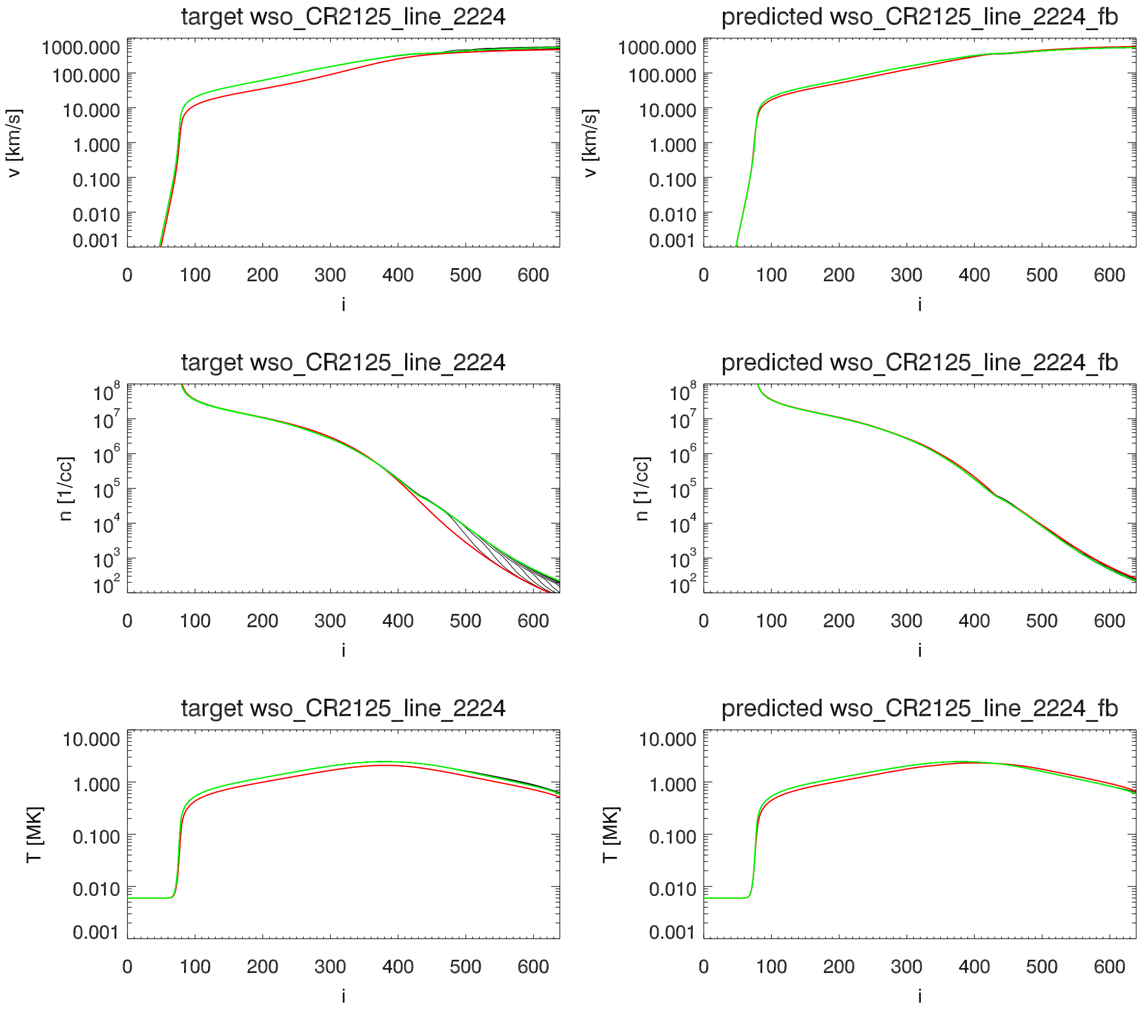

target wso_CR2056_line_1689
predicted wso_CR2056_line_1689_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

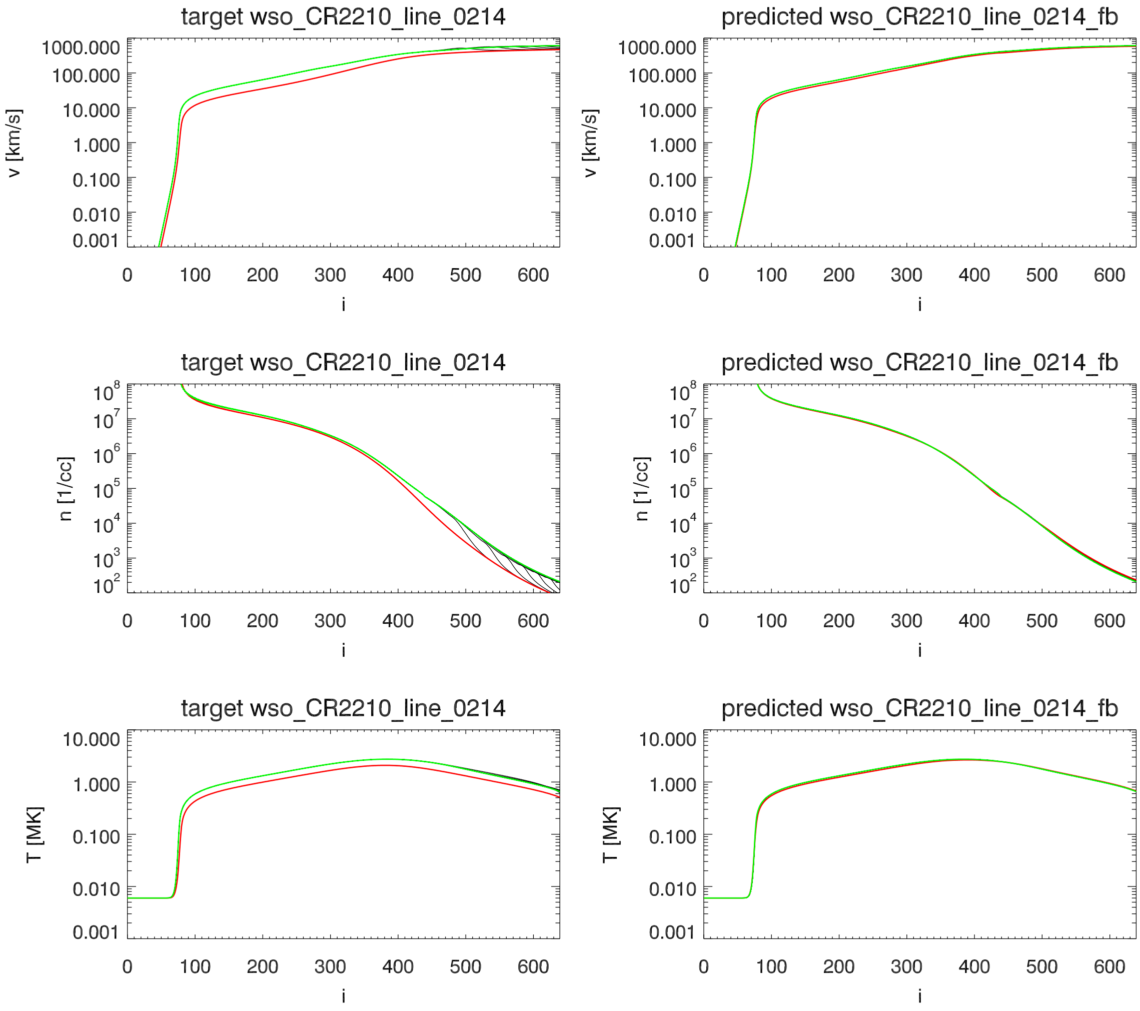

target wso_CR2056_line_1844
predicted wso_CR2056_line_1844_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
0
100
200
300
400
500
600
i

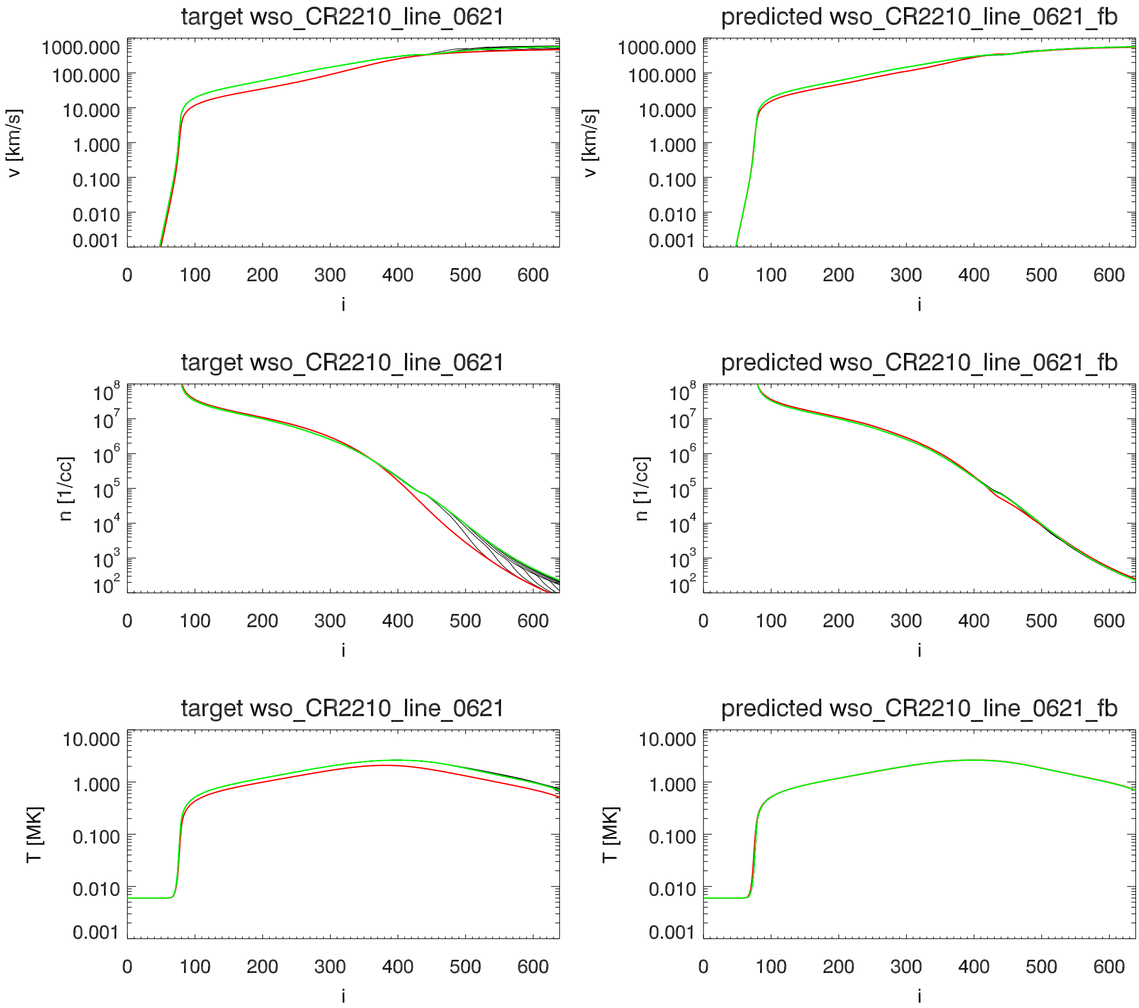

target wso_CR2056_line_2079
predicted wso_CR2056_line_2079_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
0
100
200
300
400
500
600
i

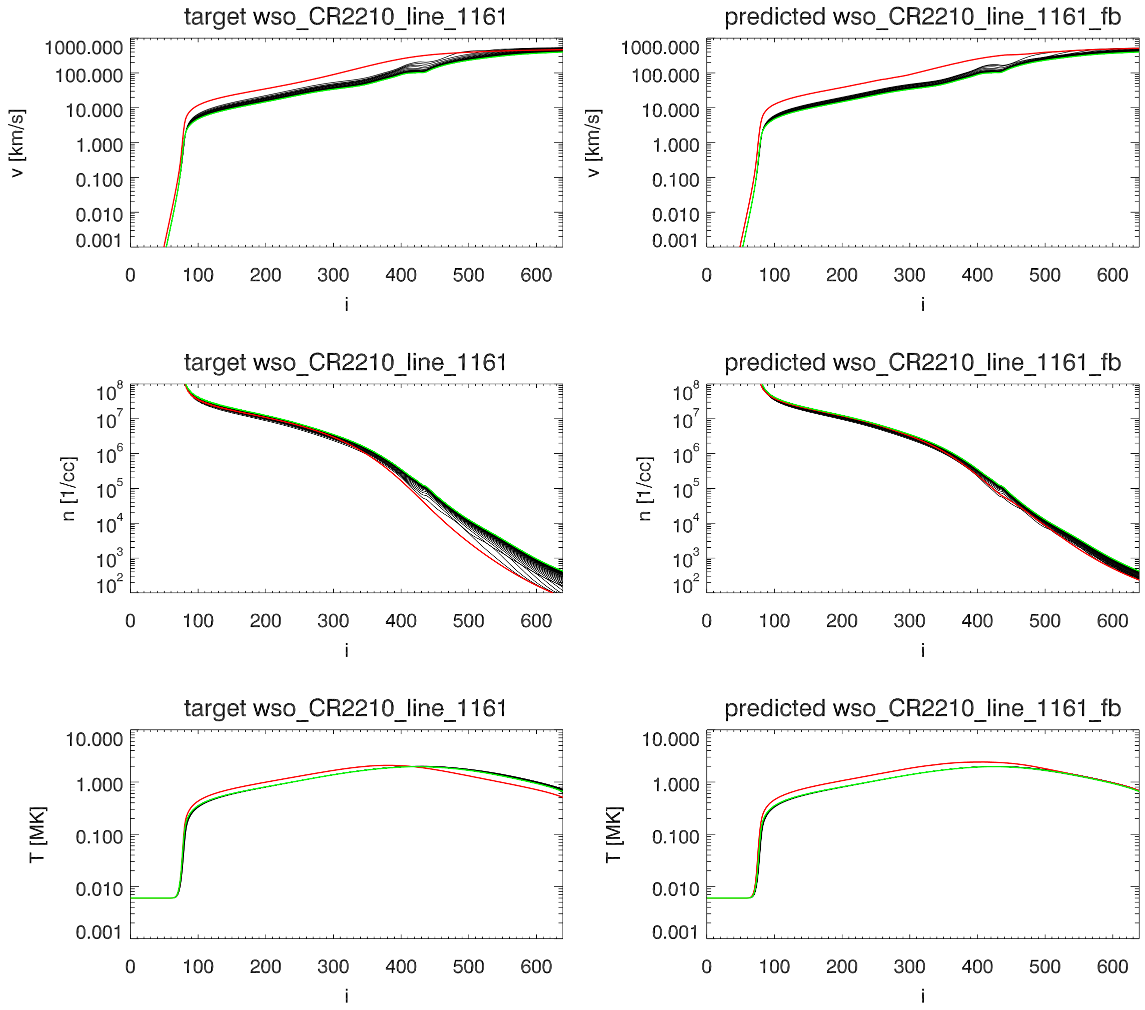

target wso_CR2125_line_0040
predicted wso_CR2125_line_0040_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
0
100
200
300
400
500
600
i

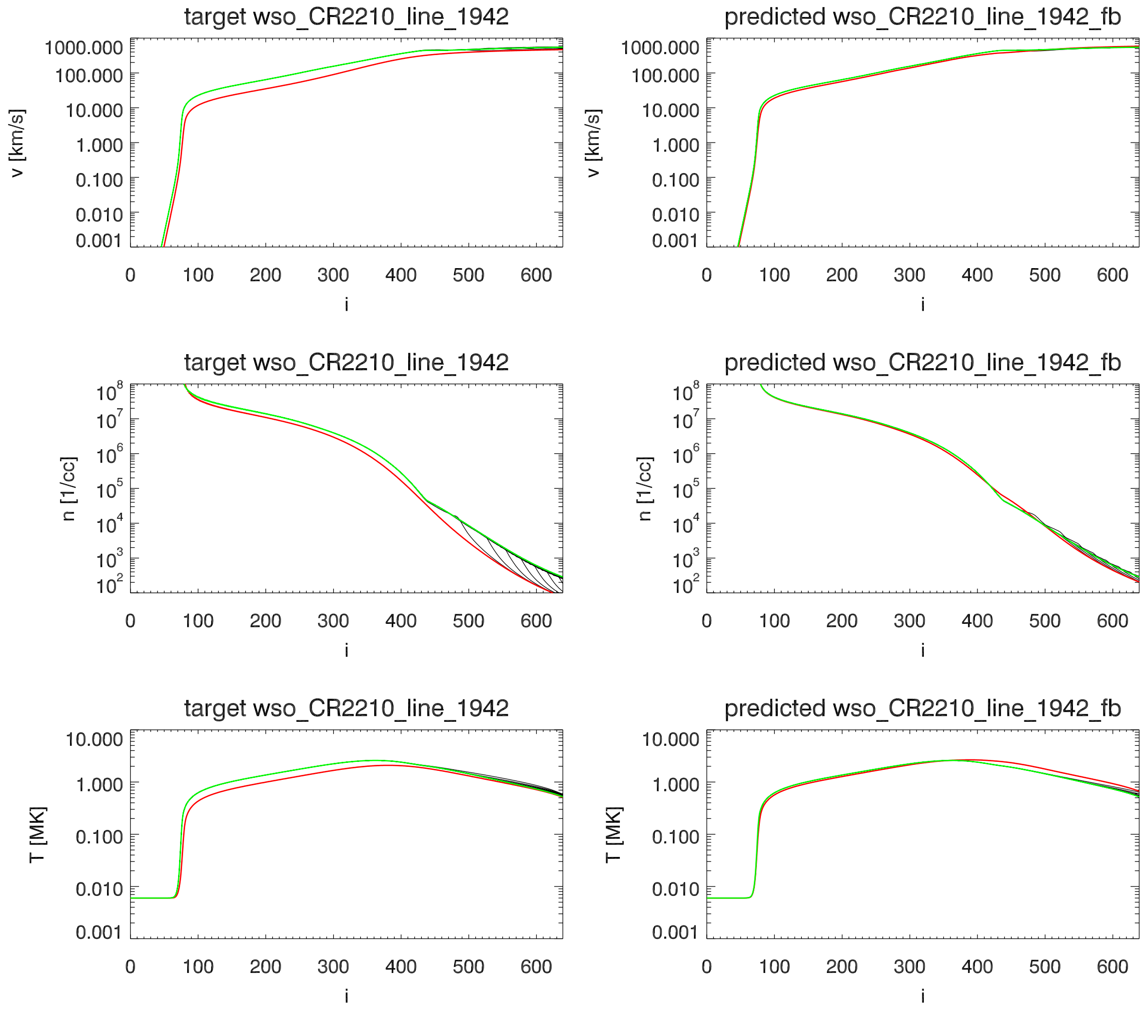

target wso_CR2125_line_1033
predicted wso_CR2125_line_1033_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
0
100
200
300
400
500
600
i


target wso_CR2125_line_2224
predicted wso_CR2125_line_2224_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i


target wso_CR2210_line_0214
predicted wso_CR2210_line_0214_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i


target wso_CR2210_line_0621
predicted wso_CR2210_line_0621_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

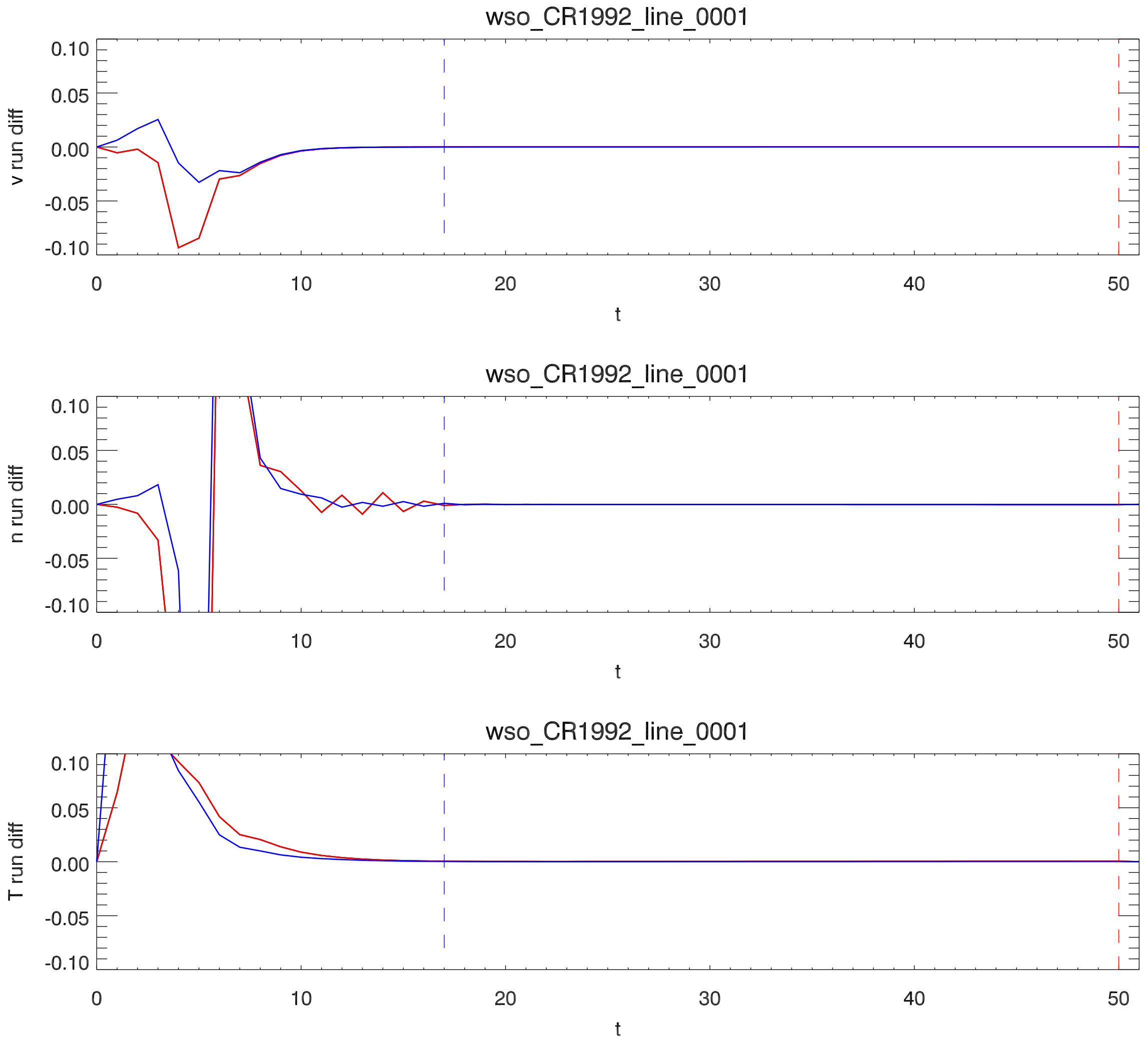

target wso_CR2210_line_1161
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i
predicted wso_CR2210_line_1161_fb
v [km/s]
target wso_CR2210_line_1161
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
predicted wso_CR2210_line_1161_fb
n [1/cc]
target wso_CR2210_line_1161
T [MK]
10.000
1.000
0.100
0.010
0.001
predicted wso_CR2210_line_1161_fb
T [MK]

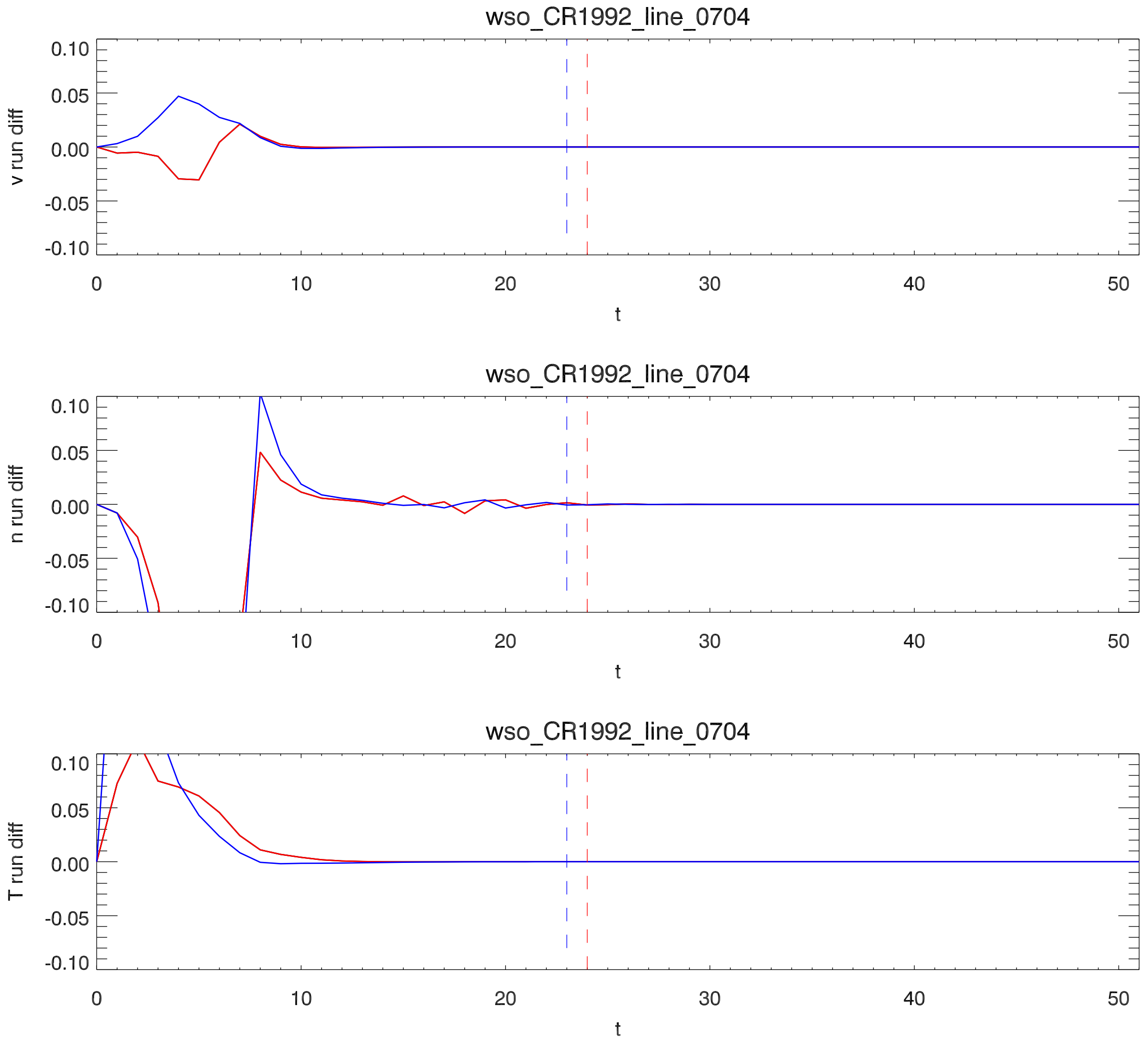

target wso_CR2210_line_1942
predicted wso_CR2210_line_1942_fb
v [km/s]
1000.000
100.000
10.000
1.000
0.100
0.010
0.001
n [1/cc]
10^8
10^7
10^6
10^5
10^4
10^3
10^2
T [MK]
10.000
1.000
0.100
0.010
0.001
0
100
200
300
400
500
600
i

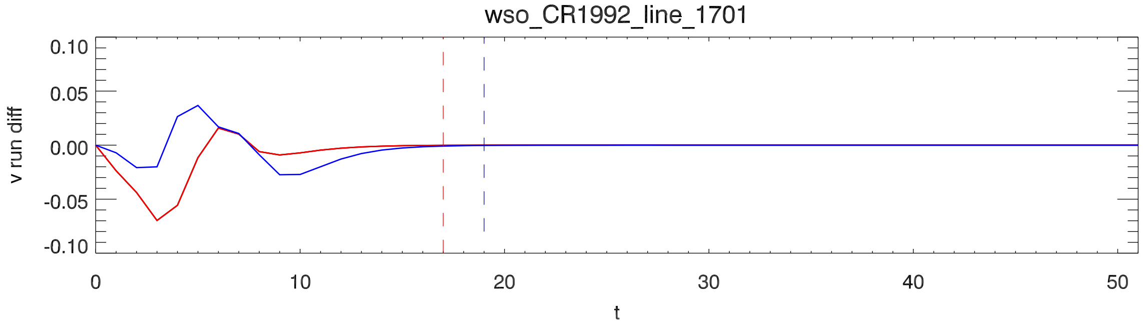
Appendix C

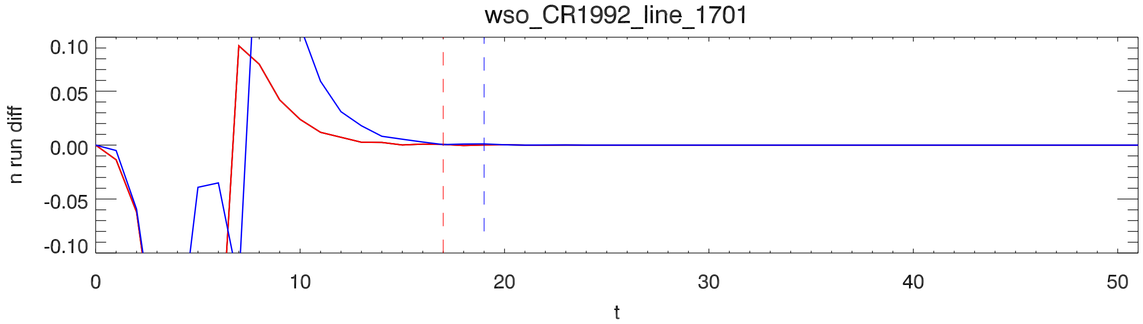
Results from MULTI_VP Runs with Convergence Analysis

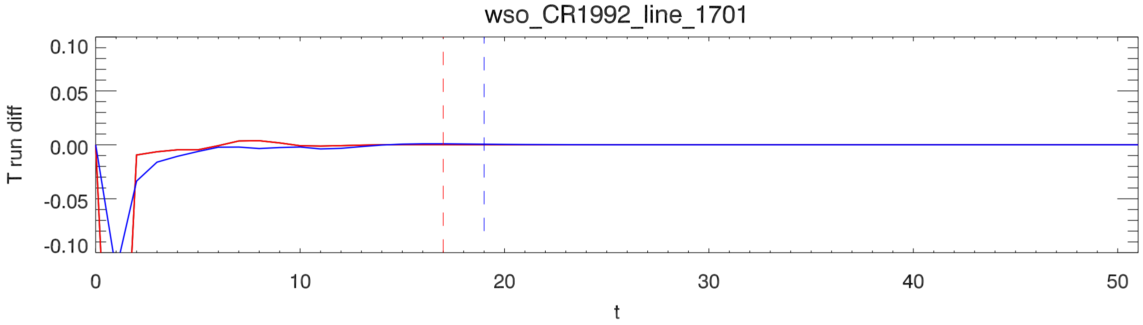
Comparison of standard and predicted profiles performance in convergence terms. Each graph shows the relative variations measured at a reference altitude between successive outputs of the code ("running differences"), the standard profiles are represented in red and the predicted ones in blue. First for speed, then density and lastly temperature. The abscissa shows the elapsed time (in code units). The vertical dashes show the instant at which convergence was detected, defined as a threshold of relative variations for the three quantities (all three must oscillate less than the threshold value).

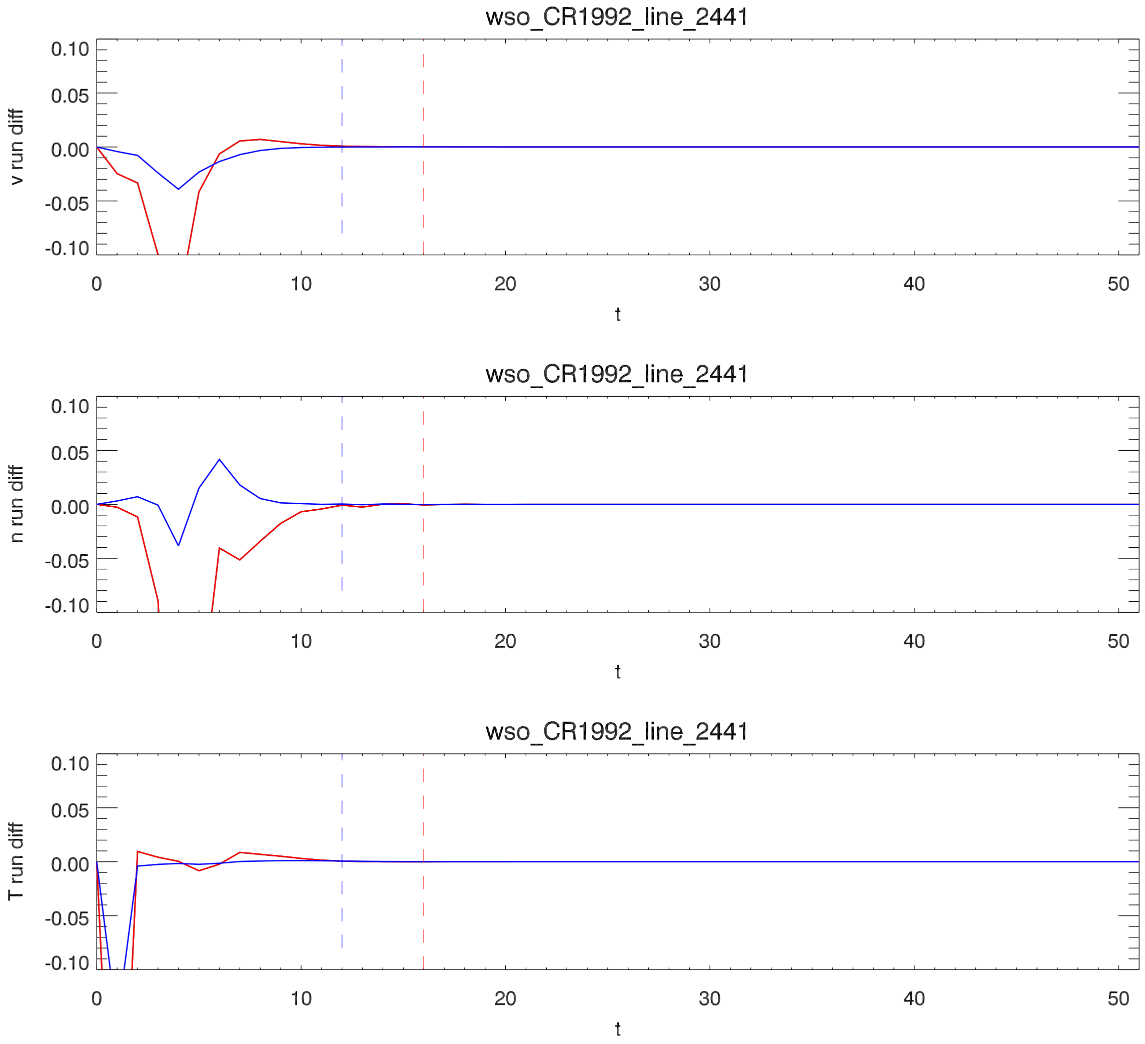

wso_CR1992_line_0001
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR1992_line_0001
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR1992_line_0001
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

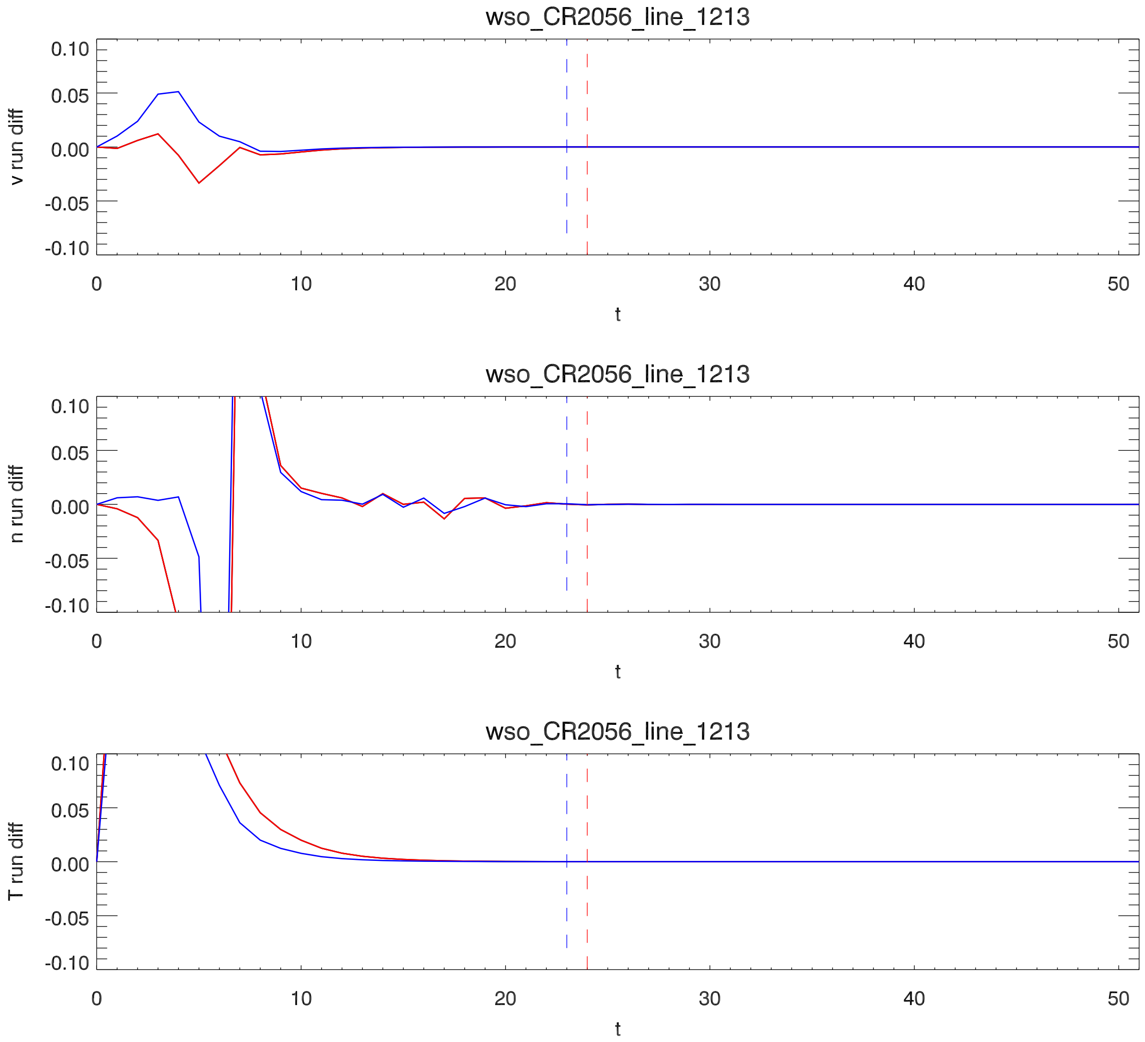

wso_CR1992_line_0704
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR1992_line_0704
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR1992_line_0704
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

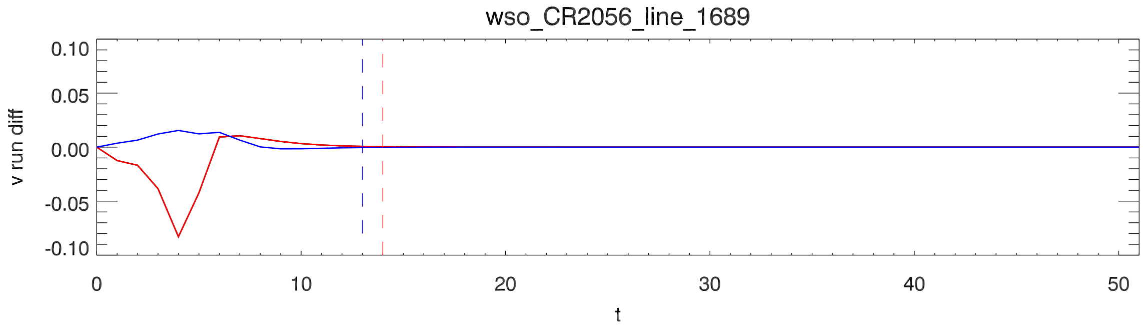

wso_CR1992_line_1701
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

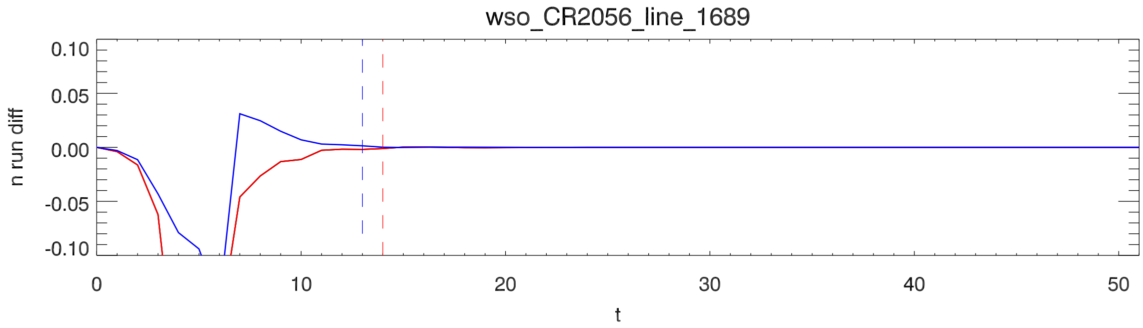

wso_CR1992_line_1701
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

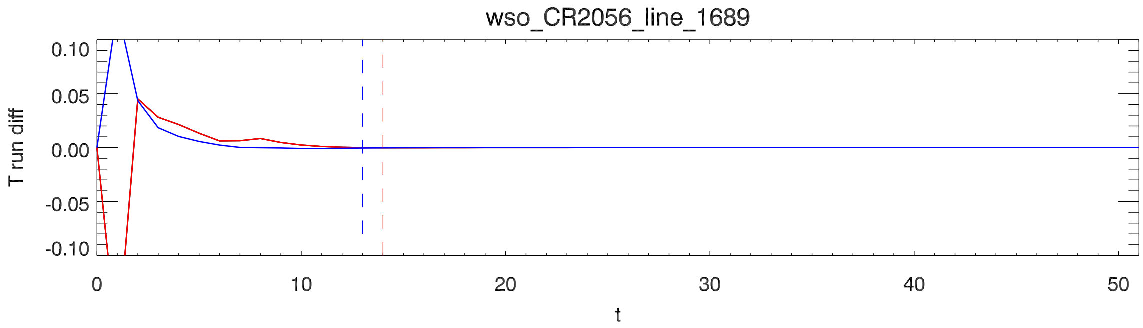

wso_CR1992_line_1701
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

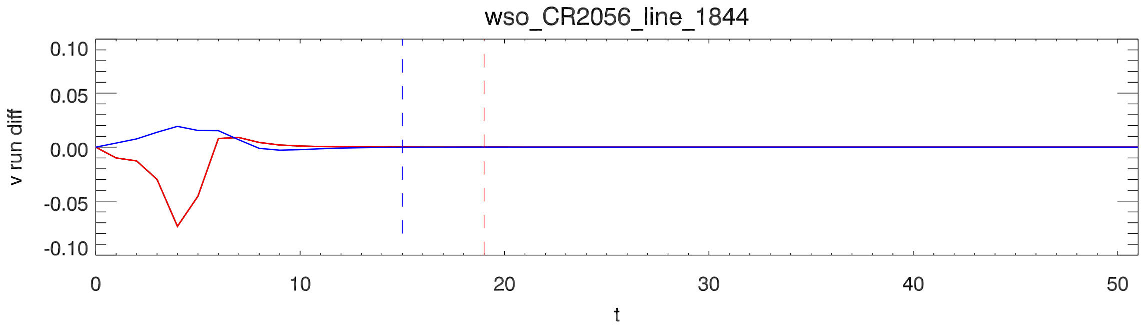

wso_CR1992_line_2441
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR1992_line_2441
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR1992_line_2441
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

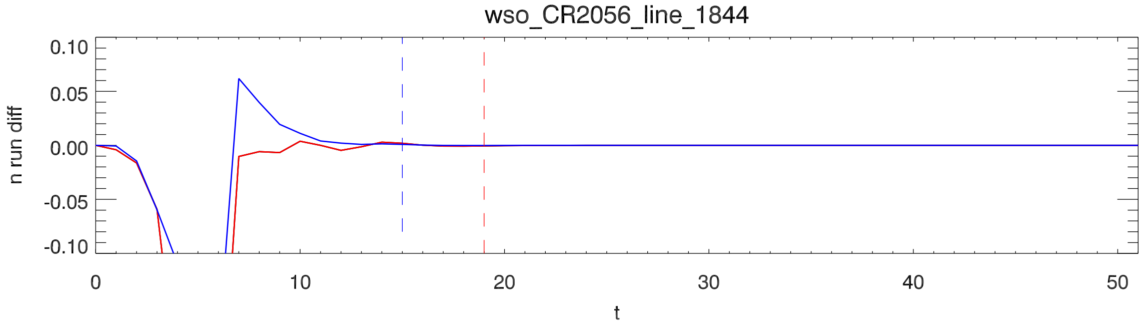

wso_CR2056_line_1213
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2056_line_1213
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2056_line_1213
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

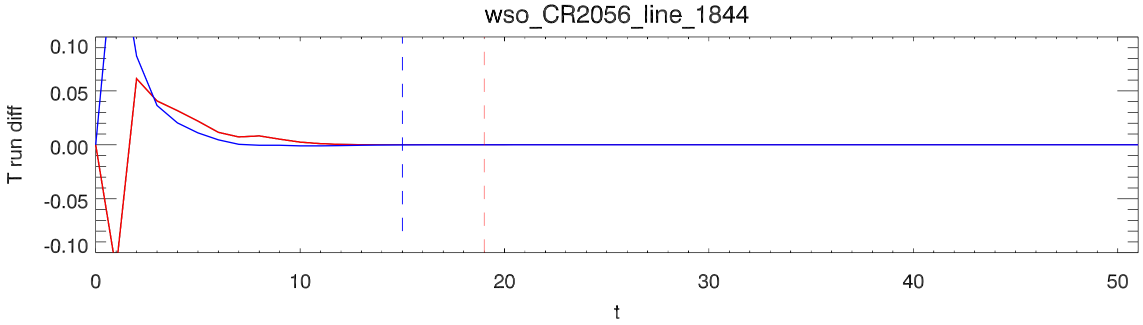

wso_CR2056_line_1689
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

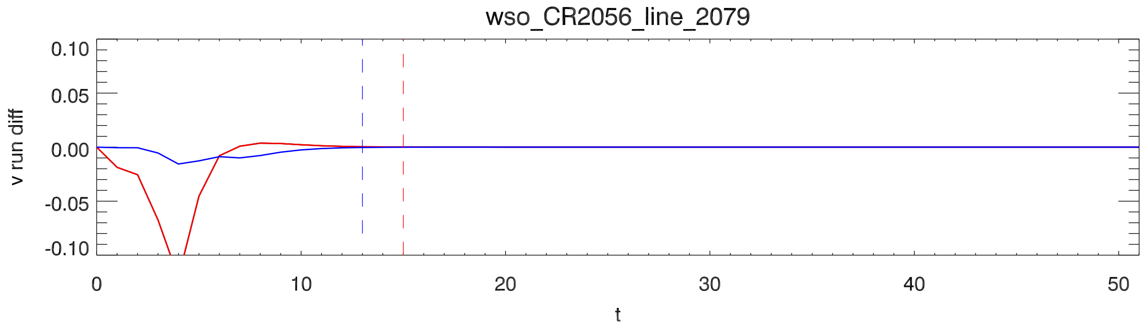

wso_CR2056_line_1689
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

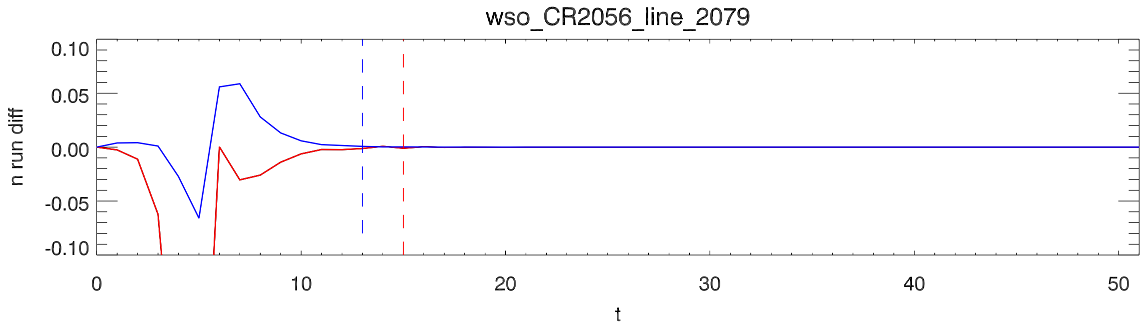

wso_CR2056_line_1689
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

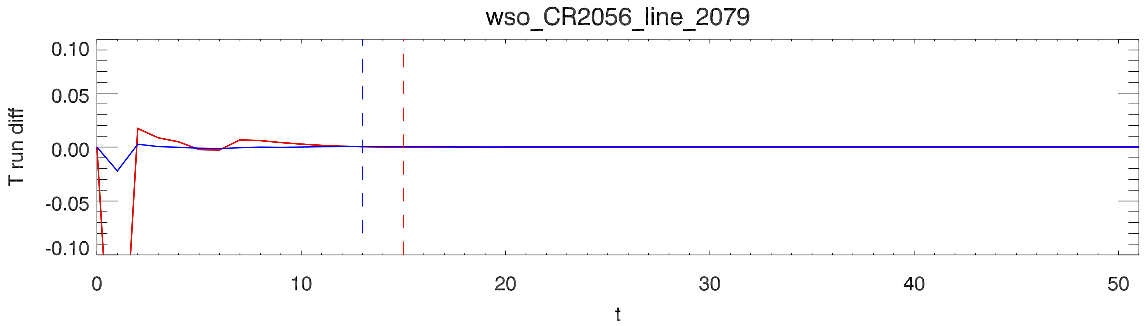

wso_CR2056_line_1844
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

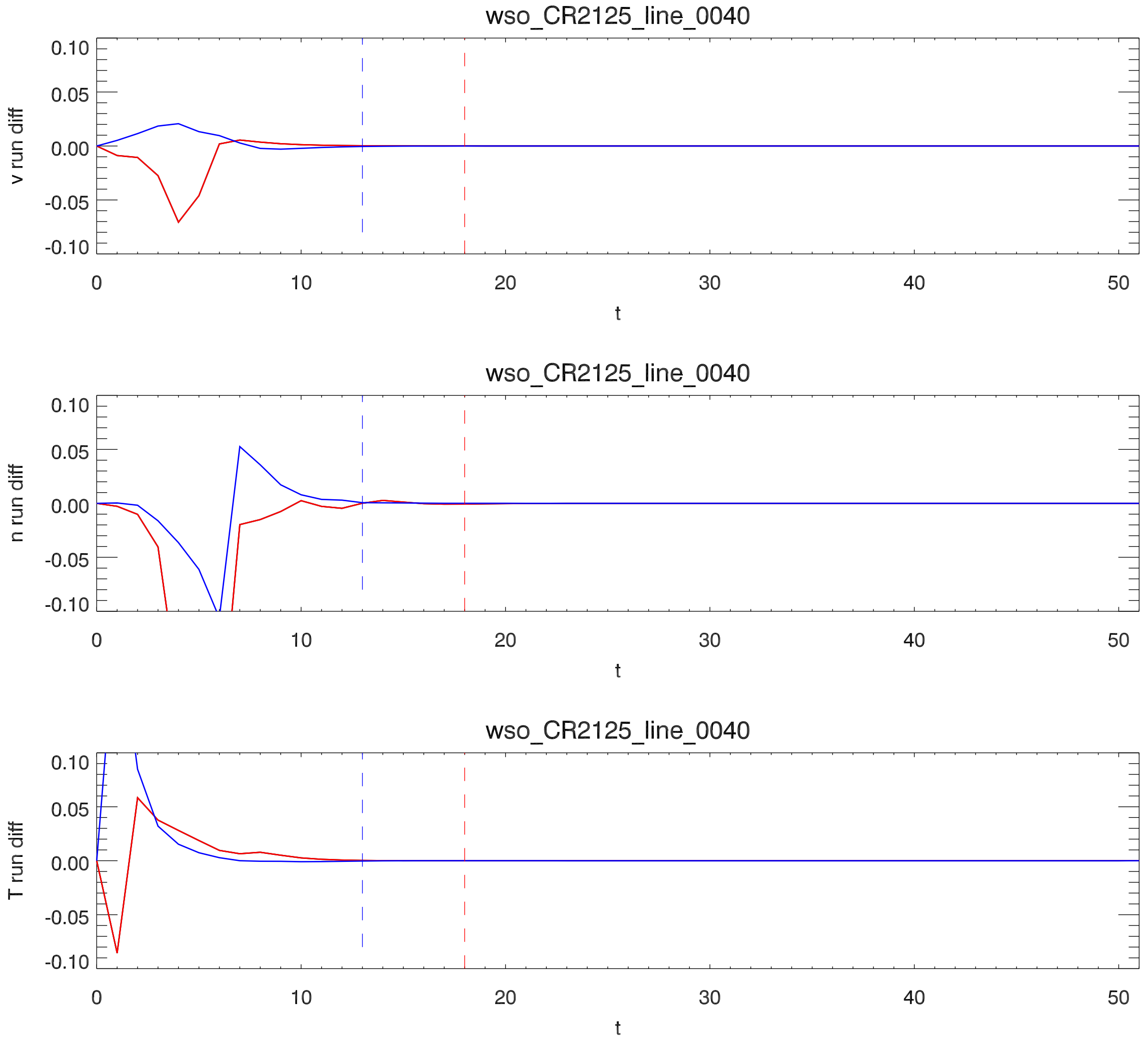

wso_CR2056_line_1844
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

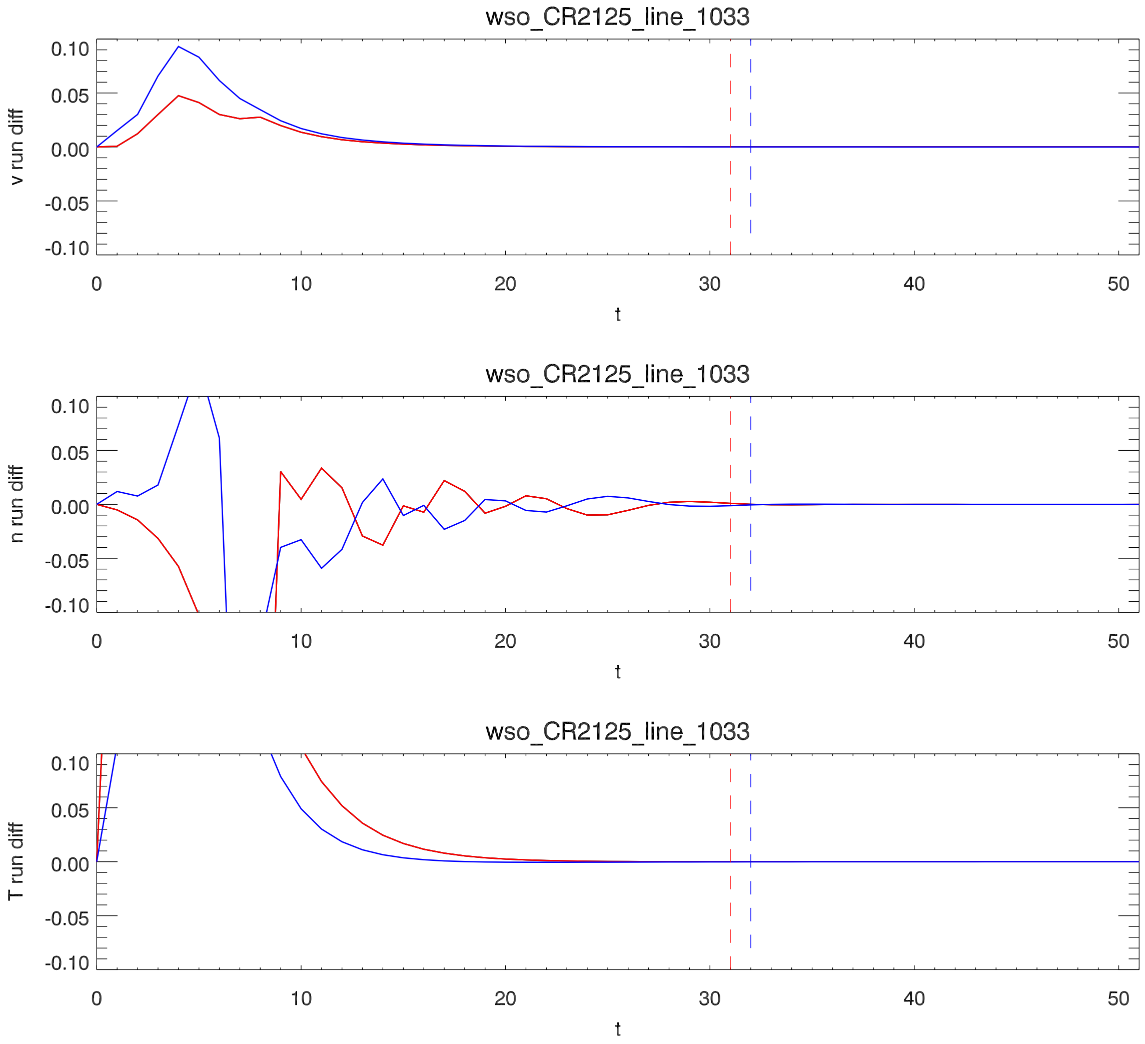

wso_CR2056_line_1844
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

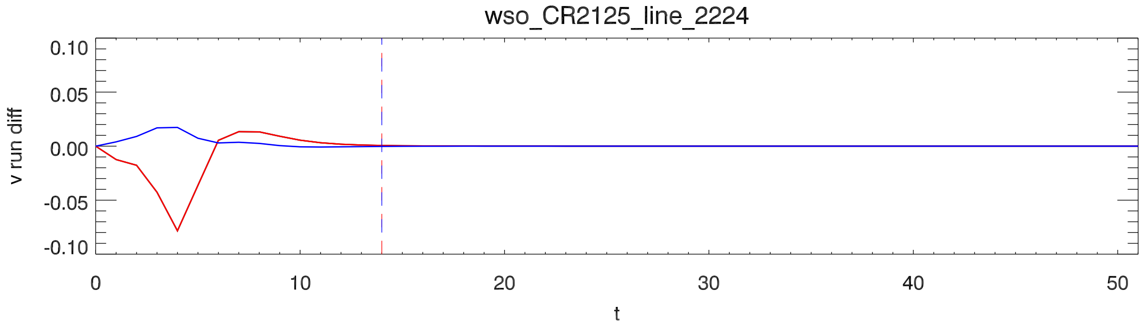

wso_CR2056_line_2079
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

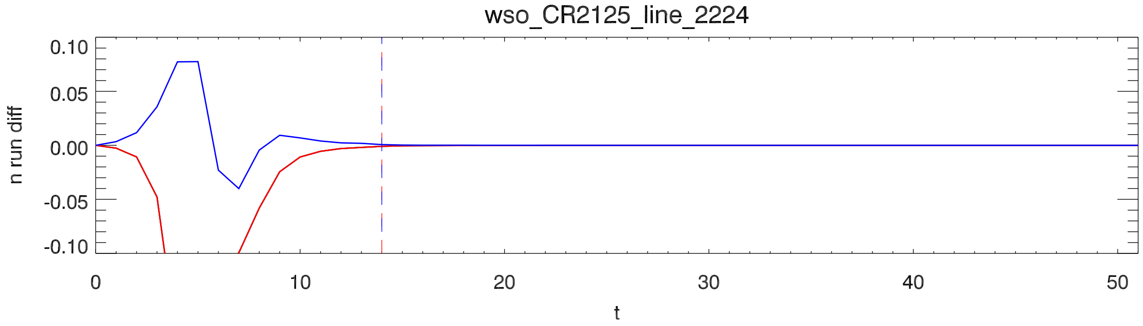

wso_CR2056_line_2079
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

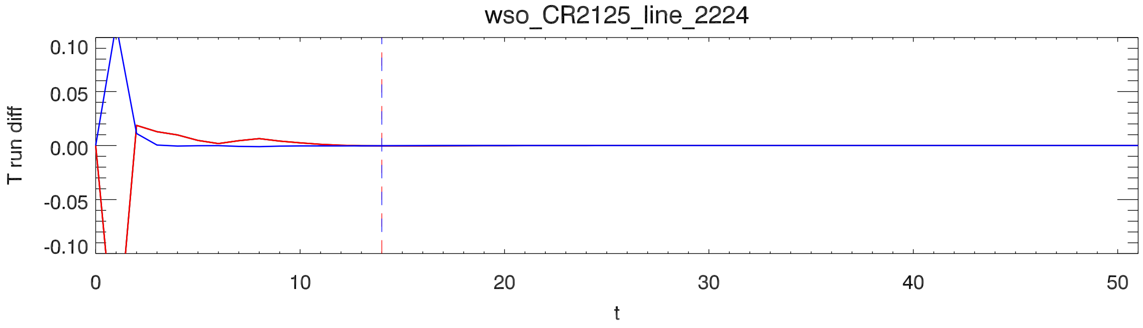

wso_CR2056_line_2079
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

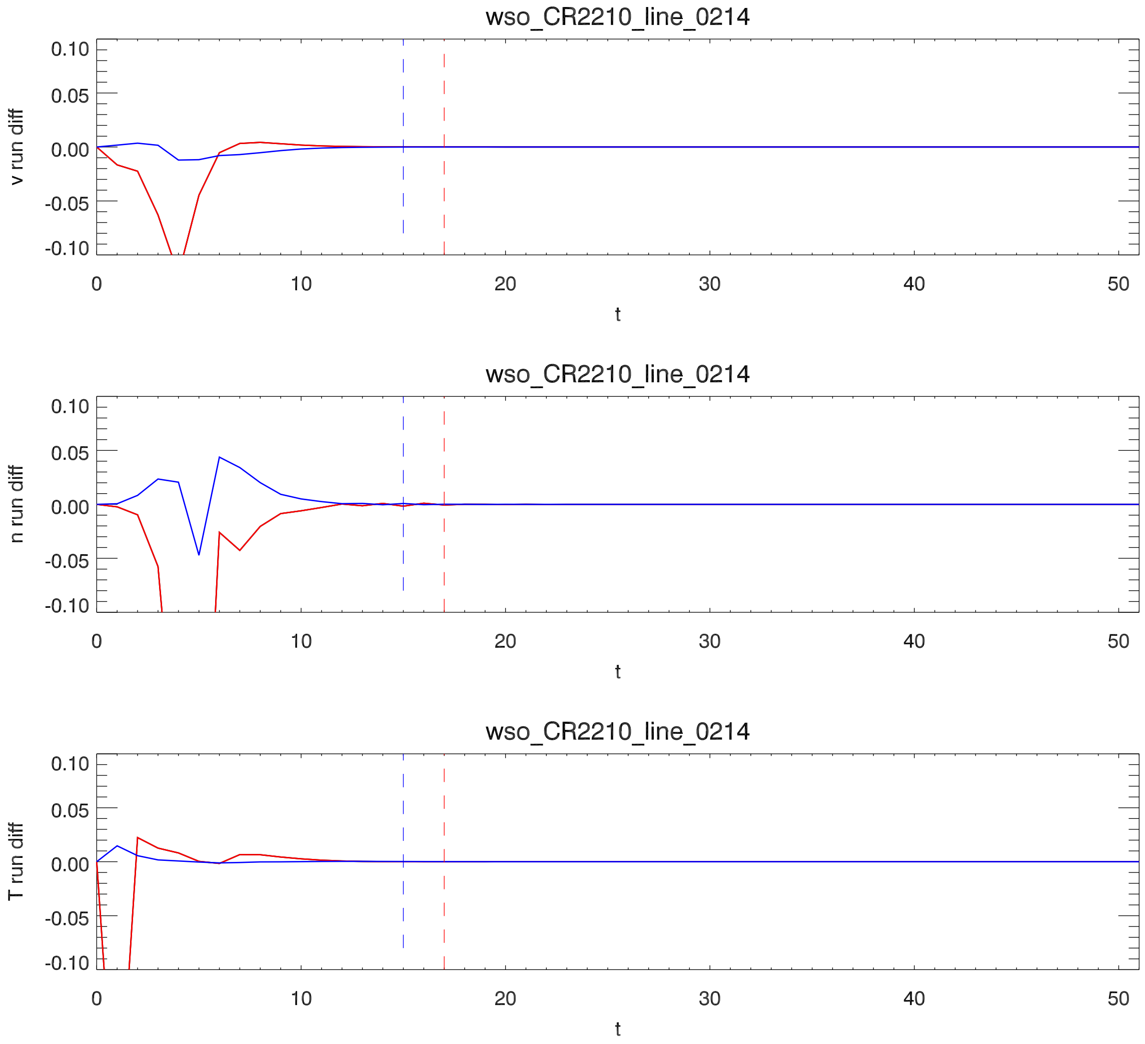

wso_CR2125_line_0040
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2125_line_0040
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2125_line_0040
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

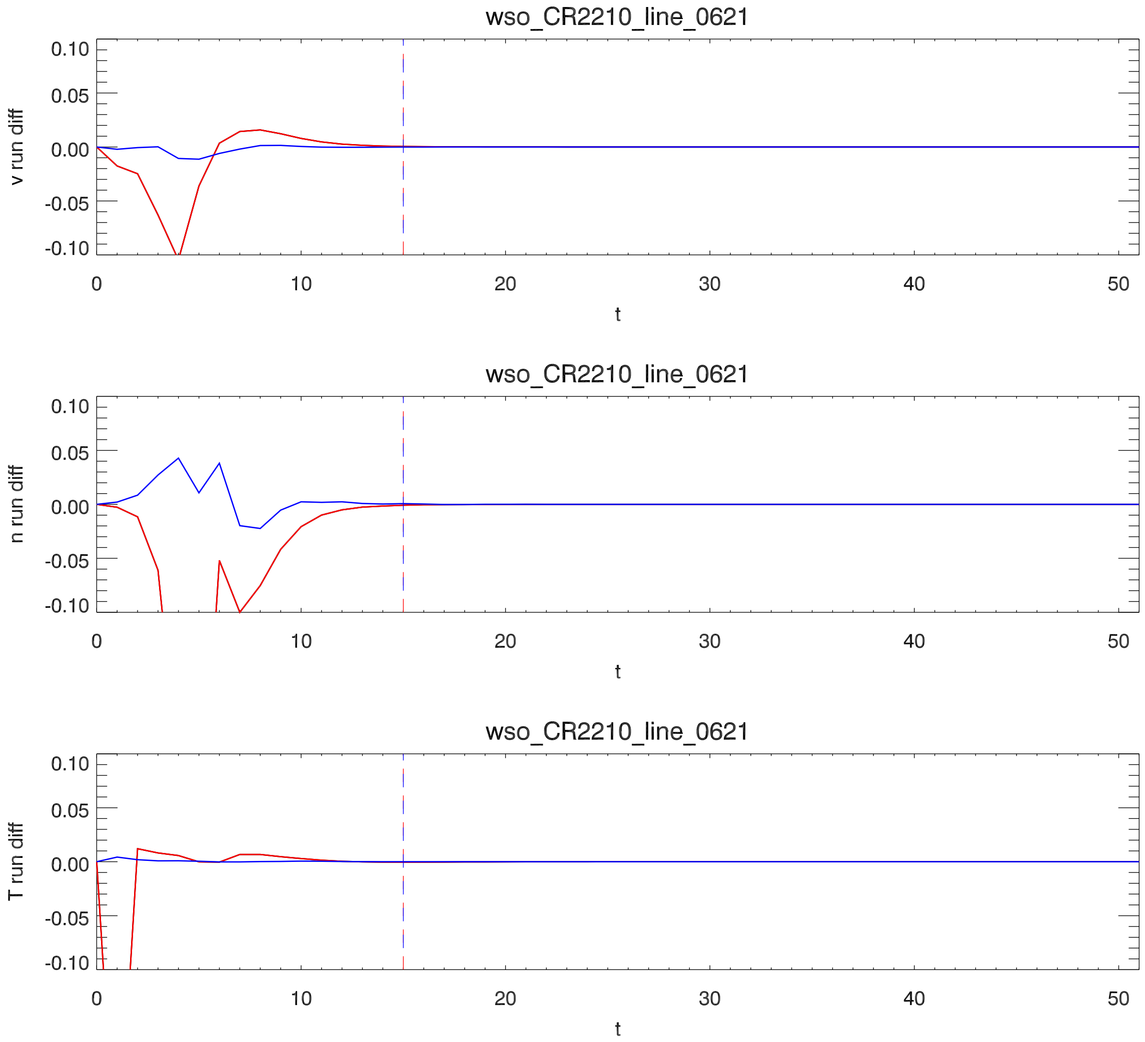

wso_CR2125_line_1033
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2125_line_1033
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2125_line_1033
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

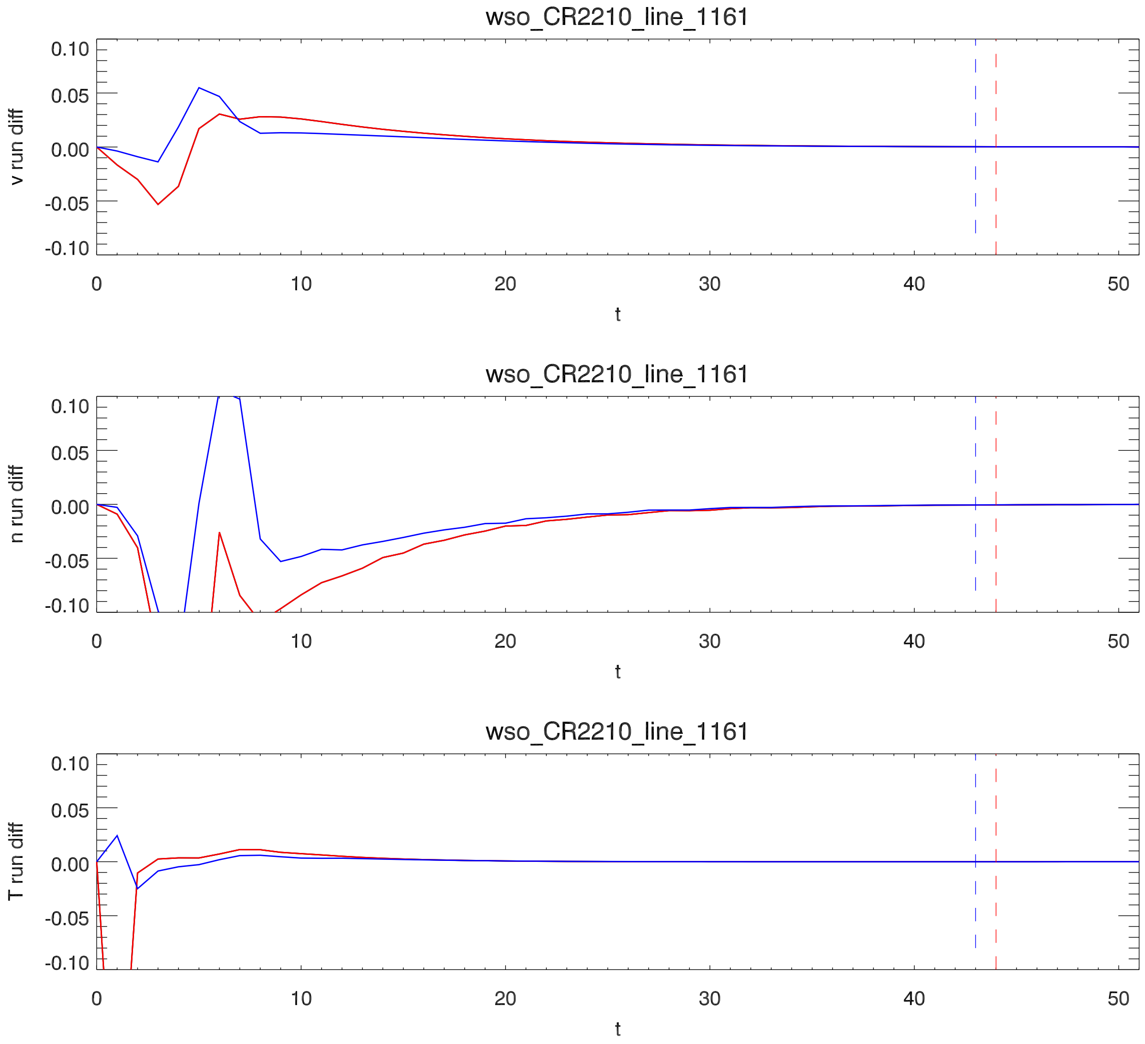

wso_CR2125_line_2224
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t


wso_CR2125_line_2224
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t


wso_CR2125_line_2224
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t


wso_CR2210_line_0214
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_0214
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_0214
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t


wso_CR2210_line_0621
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_0621
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_0621
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t


wso_CR2210_line_1161
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_1161
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_1161
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

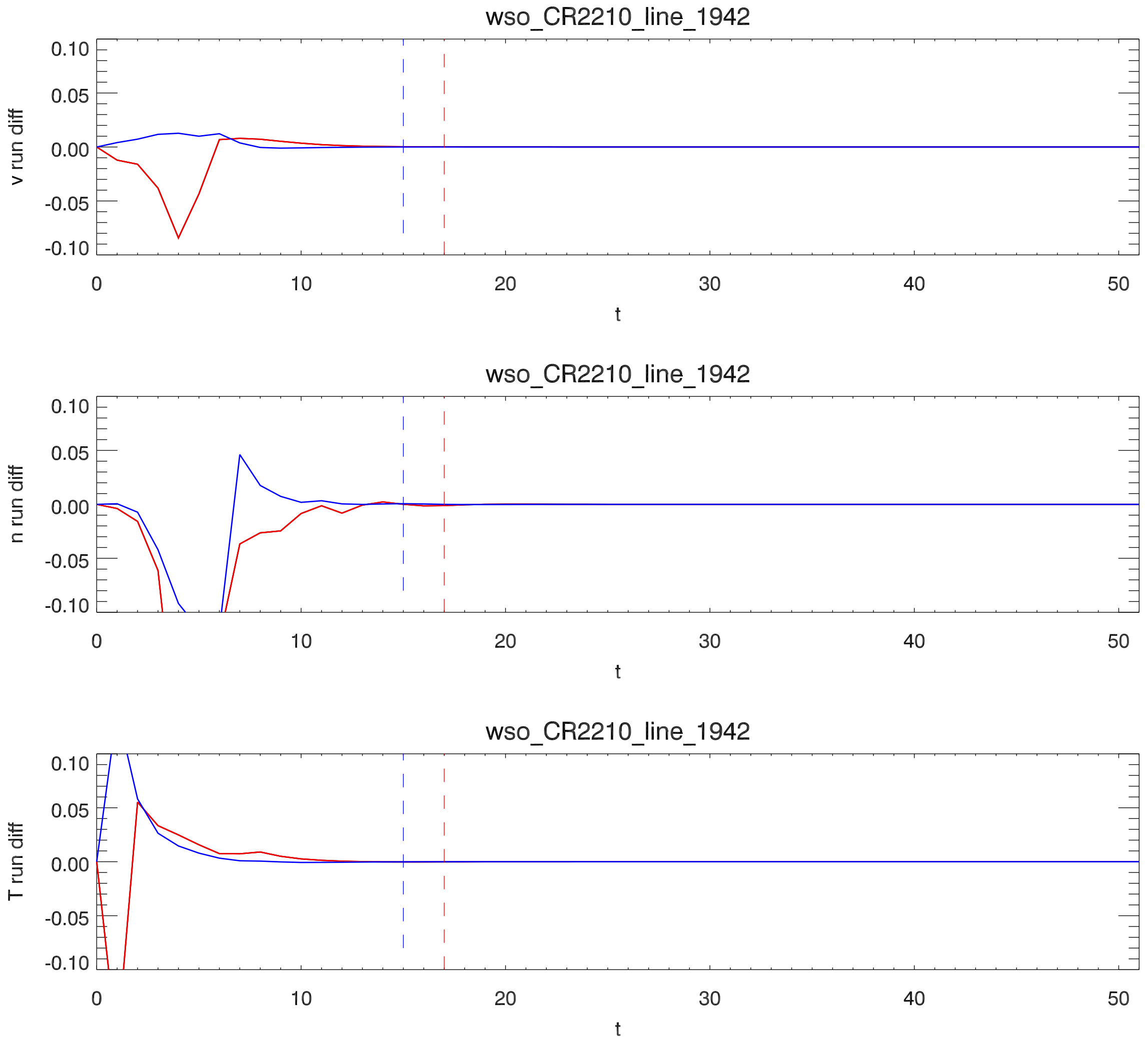

wso_CR2210_line_1942
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_1942
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t
wso_CR2210_line_1942
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
10
20
30
40
50
t

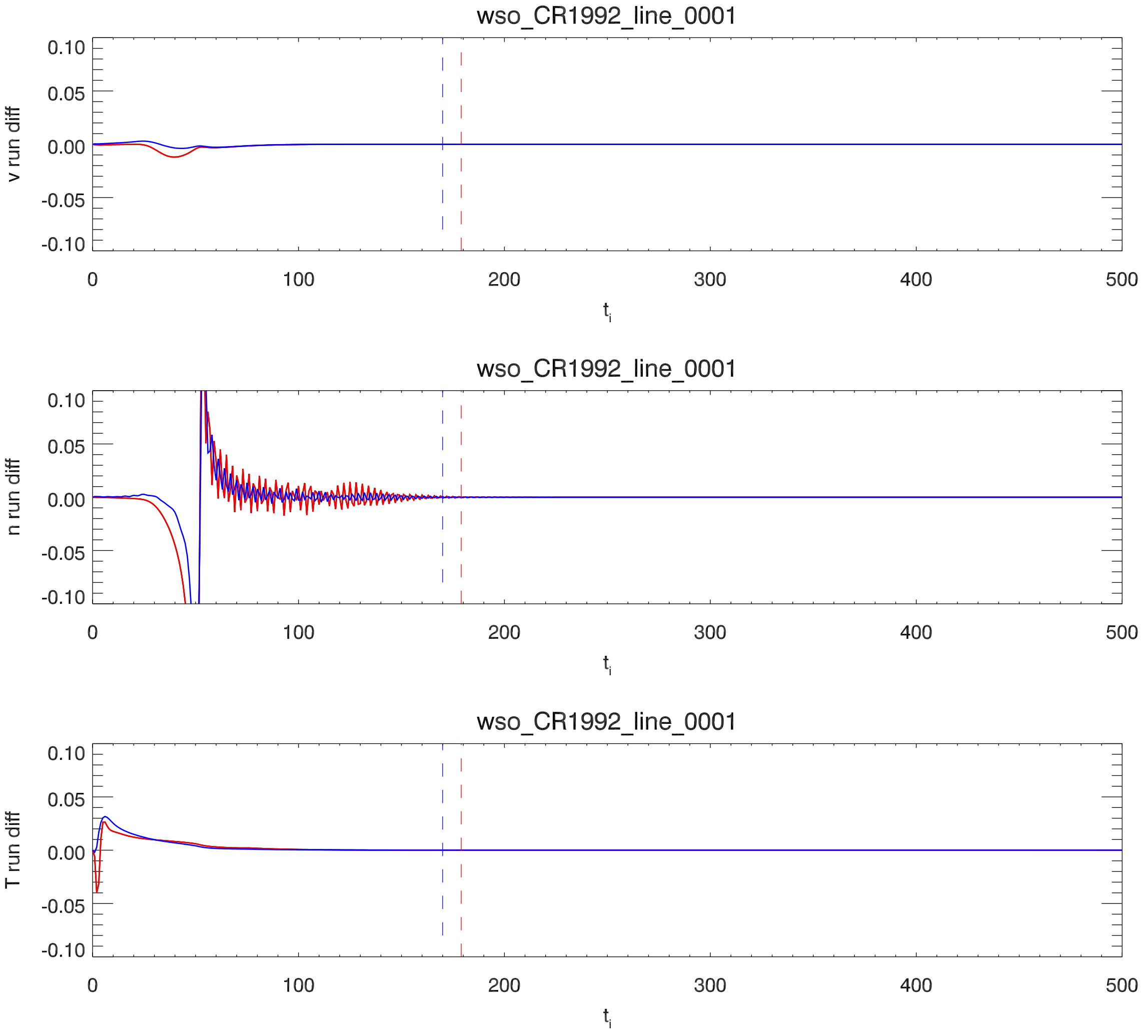
Table C.1: Analyzed run, the detected convergence times for each case, the respective number of iterations, and the speedup factor. This speedup factor was calculated by using a simple formula of $speed_fac = \frac{standard_number_iterations}{predicted_number_iterations}$.

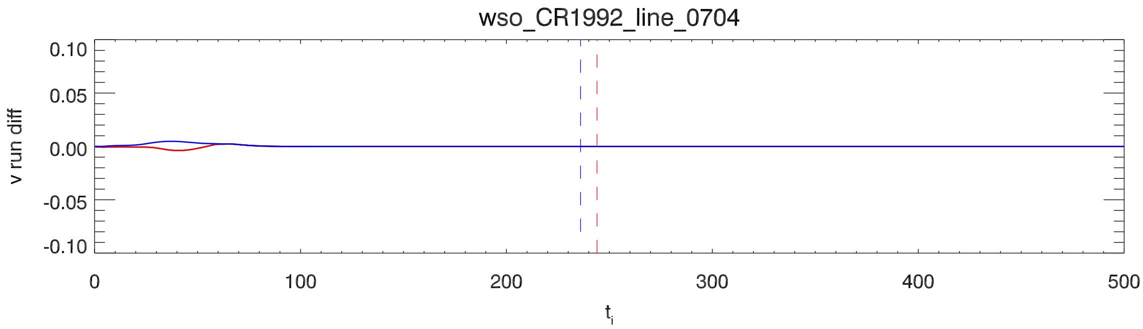
Files	standard times	predicted times	standard iterations	predicted iterations	speedup
CR1992_line_0001	50	17	7710864	2537277	3.039031
CR1992_line_0704	24	23	2686141	2545823	1.055117
CR1992_line_1701	17	19	422743	466801	0.905617
CR1992_line_2441	16	12	886770	658443	1.346768
CR2056_line_1213	24	23	3370315	3209258	1.050185
CR2056_line_1689	14	13	964518	887515	1.086763
CR2056_line_1844	19	15	1474551	1153225	1.278633
CR2056_line_2079	15	13	728292	624809	1.165623
CR2125_line_0040	18	13	1381900	988898	1.397414
CR2125_line_1033	31	32	3977693	4092486	0.97195
CR2125_line_2224	14	14	886276	880883	1.006122
CR2210_line_0214	17	15	1150994	1007776	1.142113
CR2210_line_0621	15	15	951685	945667	1.006364
CR2210_line_1161	44	43	996516	967496	1.029995
CR2210_line_1942	17	15	1145990	1001342	1.144454

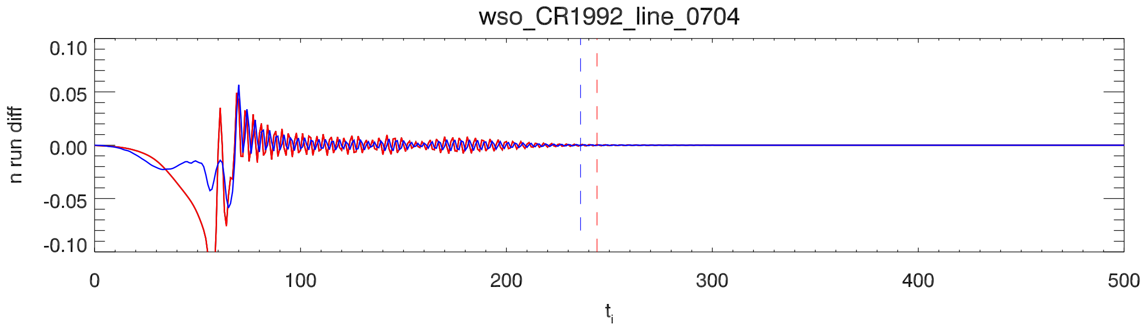
Appendix D

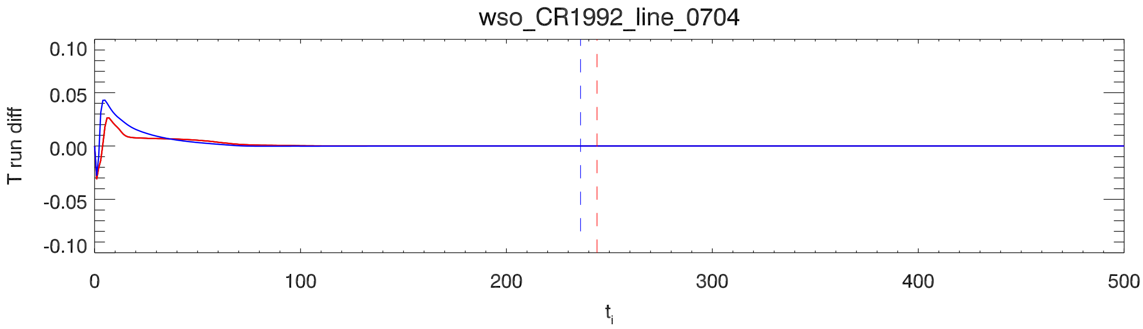
Results from MULTI_VP Runs with Convergence Analysis at 10x Data Rate

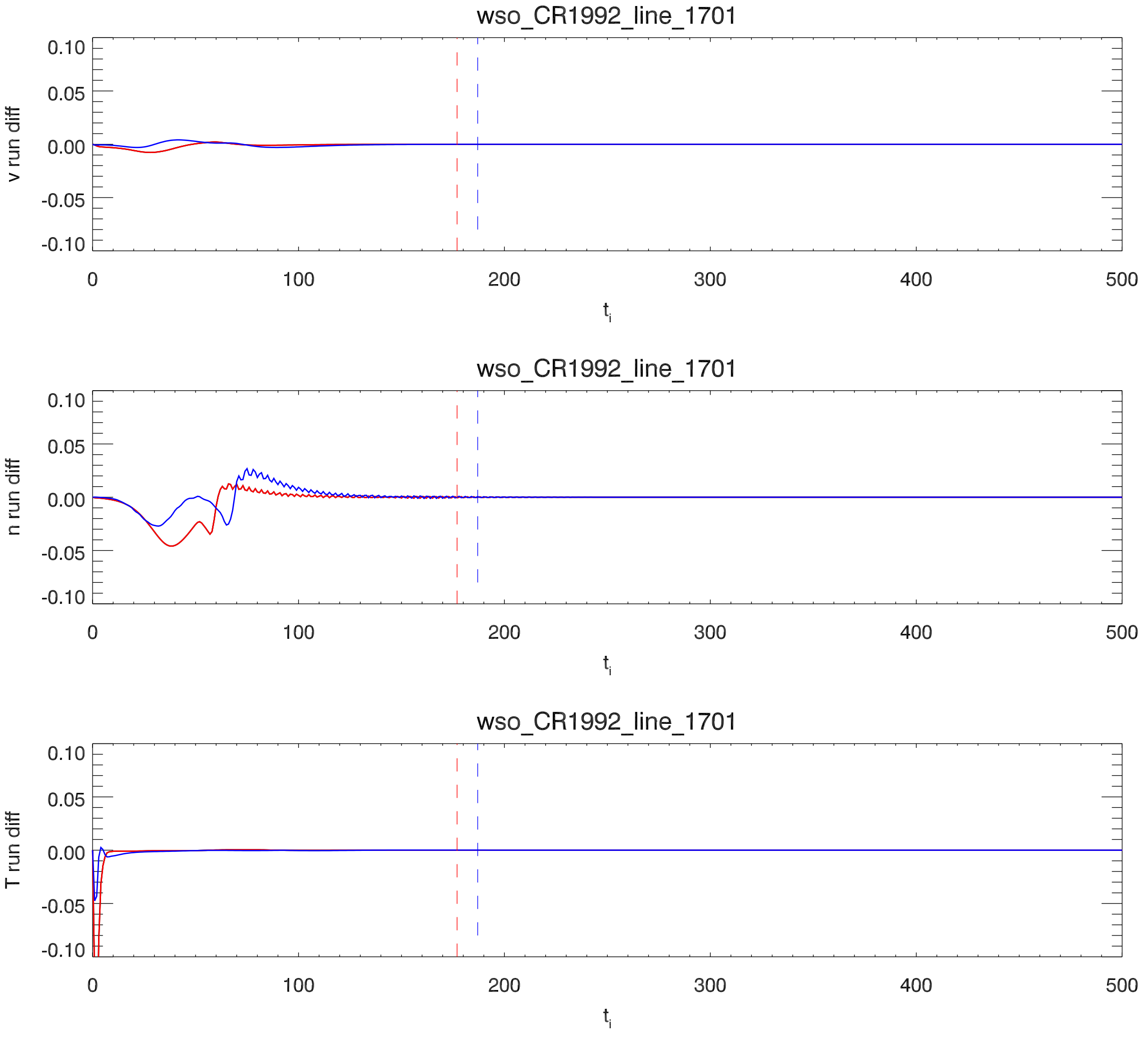
Comparison of standard and predicted profiles performance in convergence terms with a 10 times higher output data rate, using one of the 20 predicted profiles. Each graph shows the relative variations measured at a reference altitude between successive outputs of the code ("running differences"), the standard profiles are represented in red and the predicted ones in blue. First for speed, then density and lastly temperature. The abscissa shows the elapsed time (in code units). The vertical dashes show the instant at which convergence was detected, defined as a threshold of relative variations for the three quantities (all three must oscillate less than the threshold value).

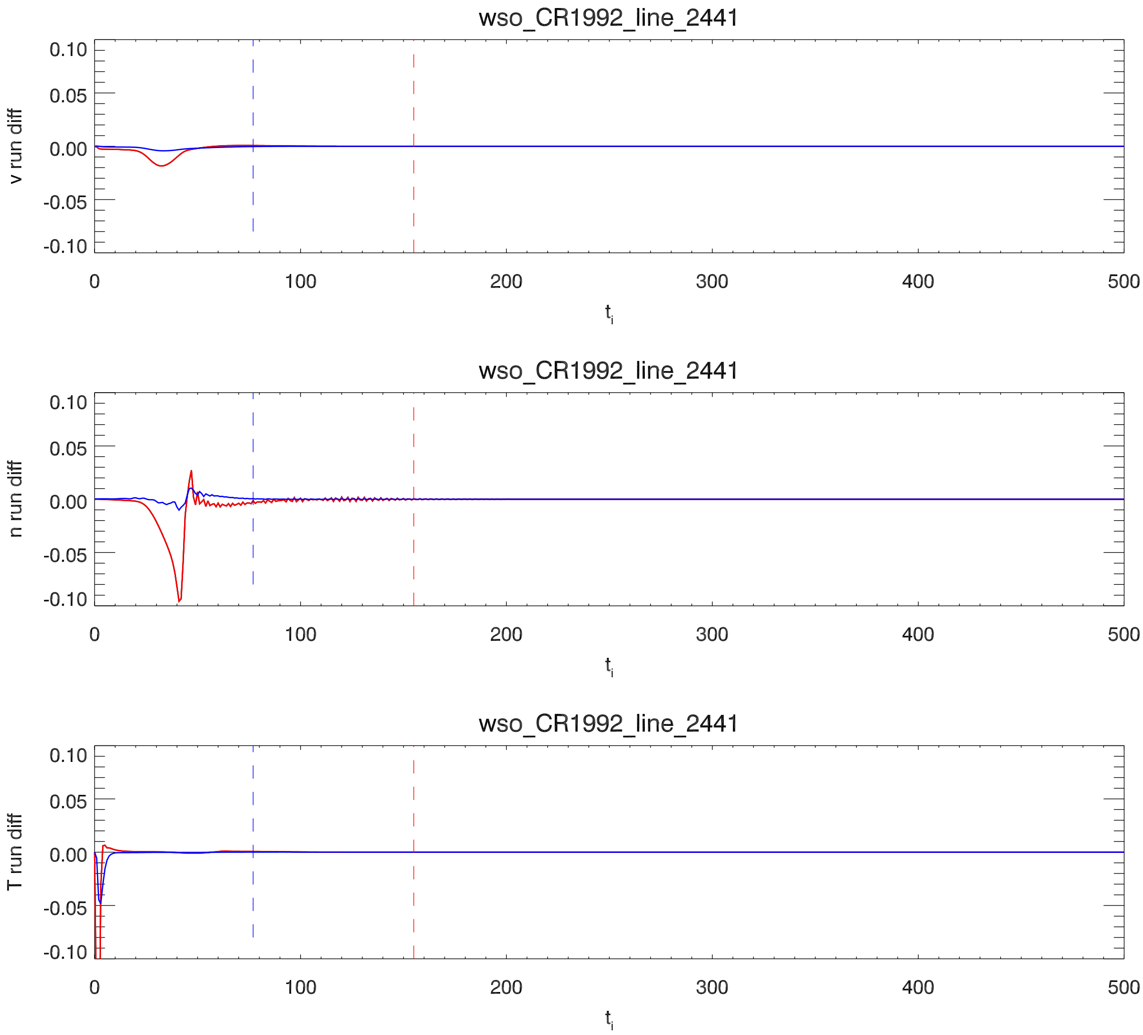

wso_CR1992_line_0001
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR1992_line_0001
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR1992_line_0001
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

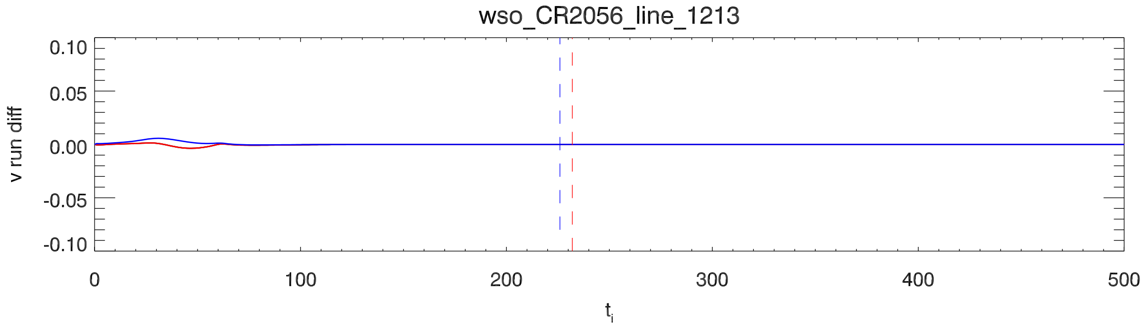

wso_CR1992_line_0704
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

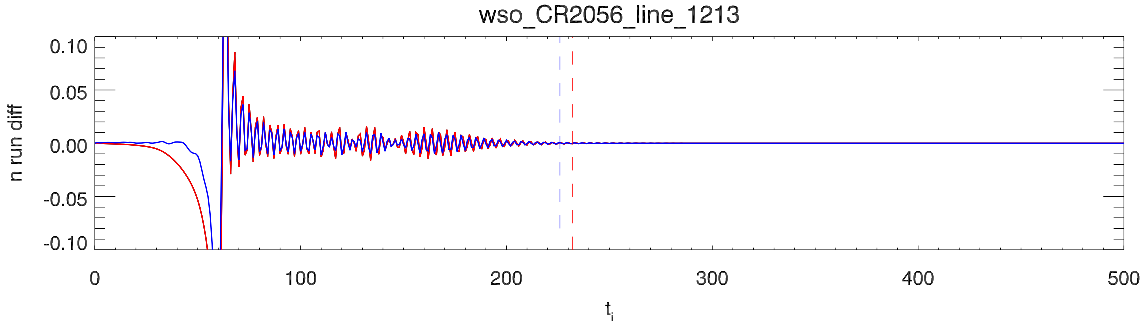

wso_CR1992_line_0704
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

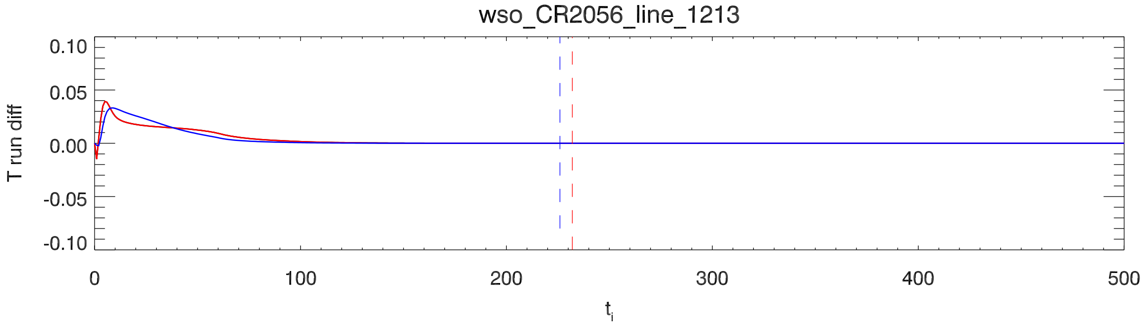

wso_CR1992_line_0704
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

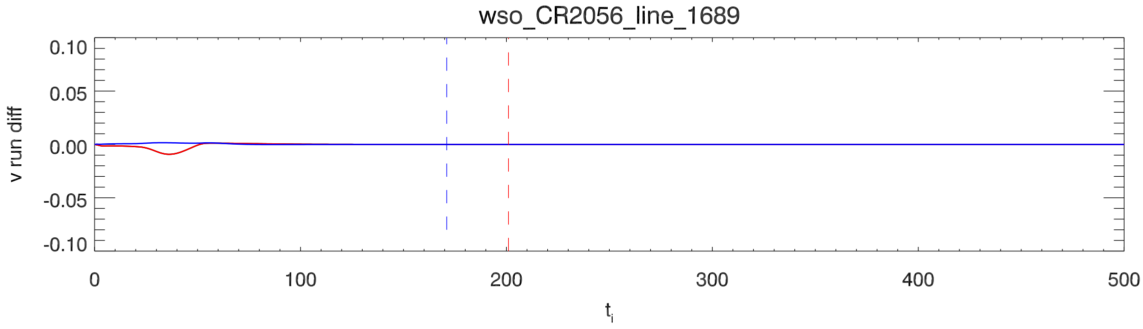

wso_CR1992_line_1701
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR1992_line_1701
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR1992_line_1701
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

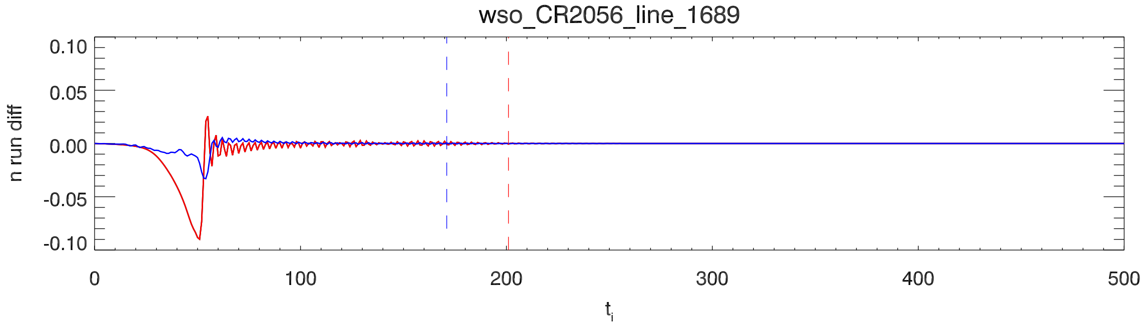

wso_CR1992_line_2441
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR1992_line_2441
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR1992_line_2441
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

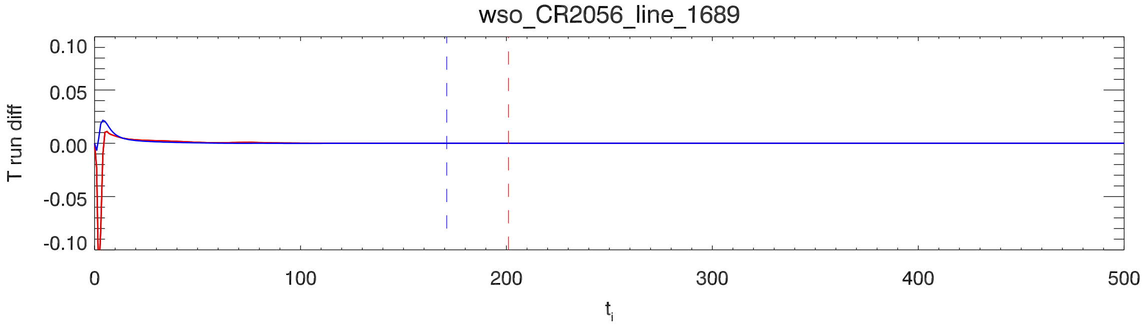

wso_CR2056_line_1213
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t$_i$

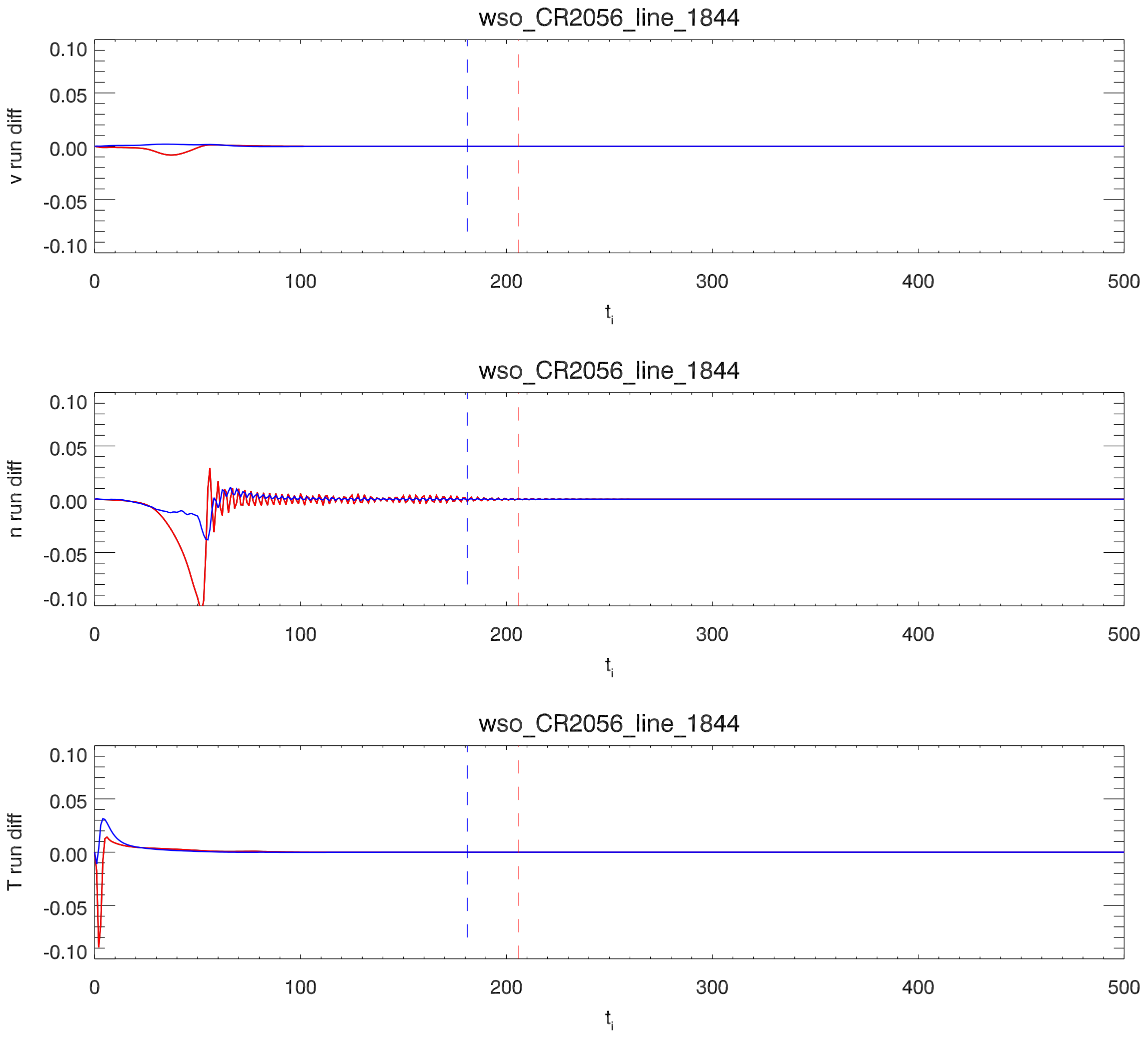

wso_CR2056_line_1213
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t$_i$

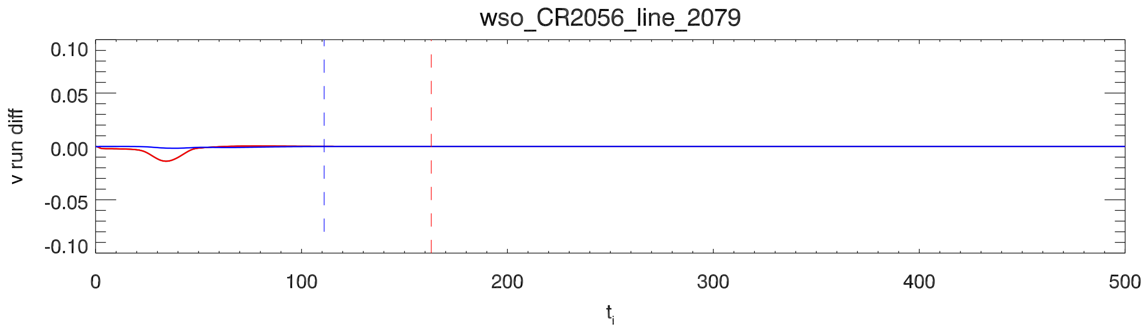

wso_CR2056_line_1213
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t$_i$

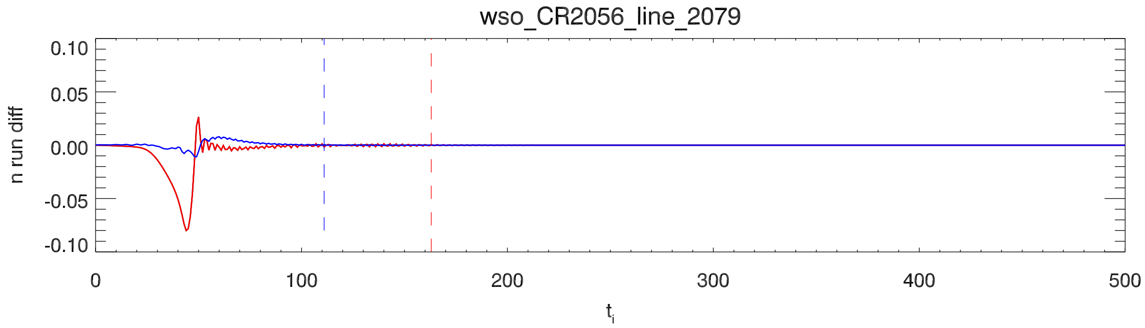

wso_CR2056_line_1689
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

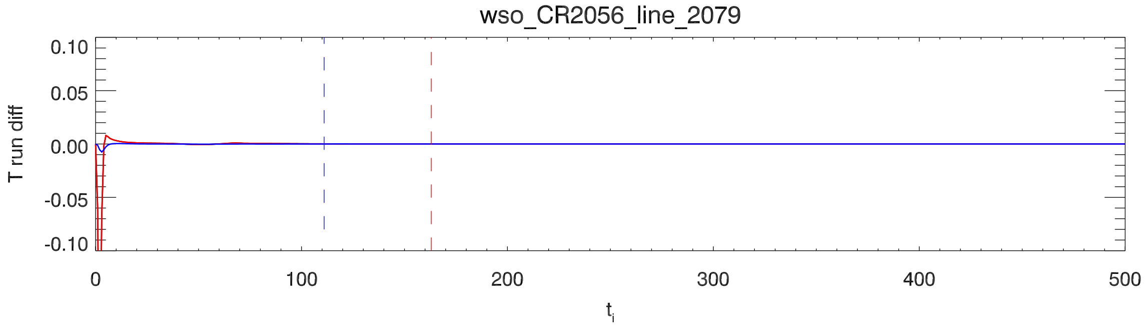

wso_CR2056_line_1689
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

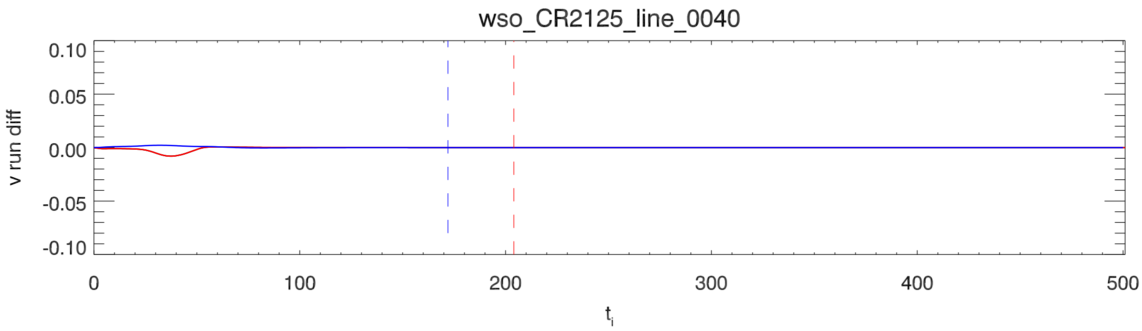

wso_CR2056_line_1689
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

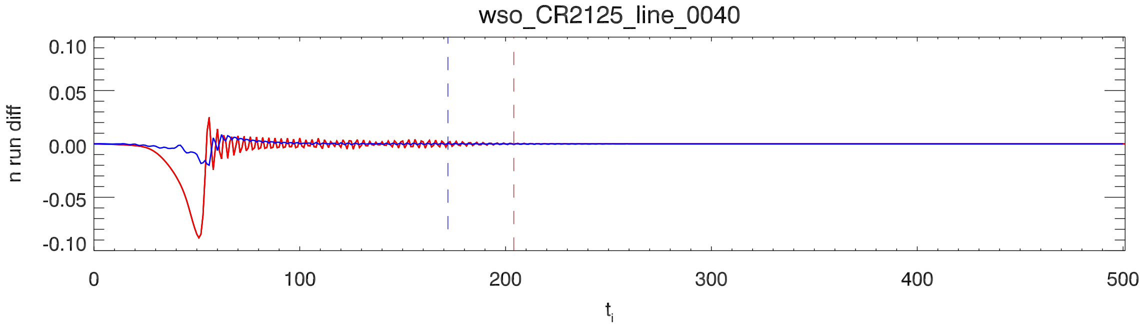

wso_CR2056_line_1844
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR2056_line_1844
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i
wso_CR2056_line_1844
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

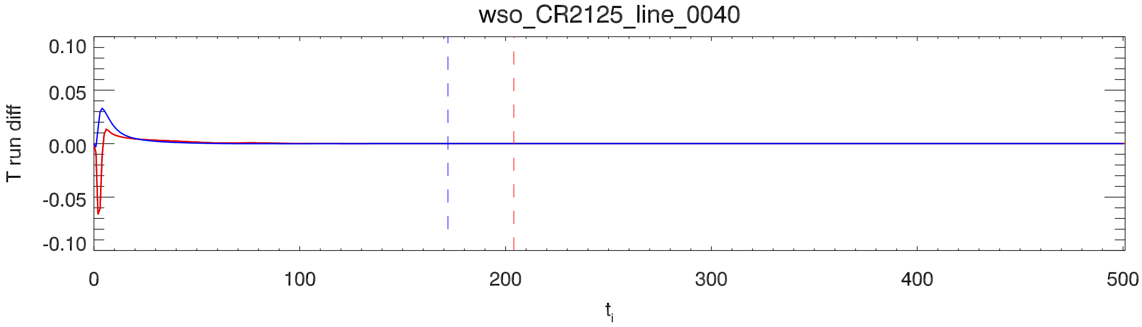

wso_CR2056_line_2079
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

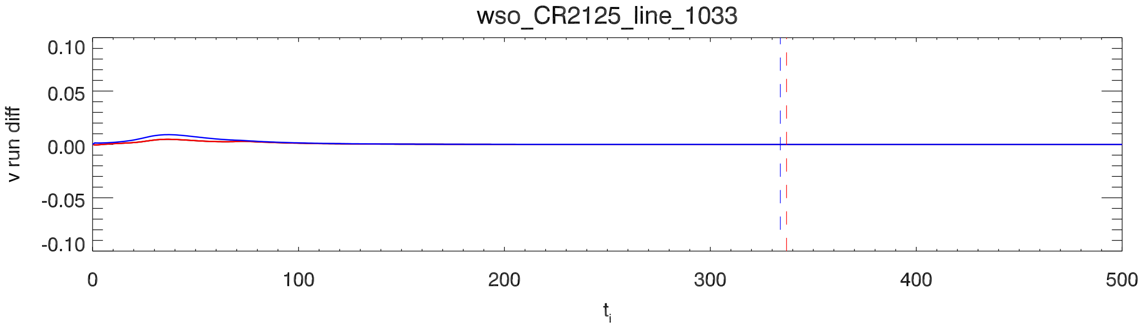

wso_CR2056_line_2079
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

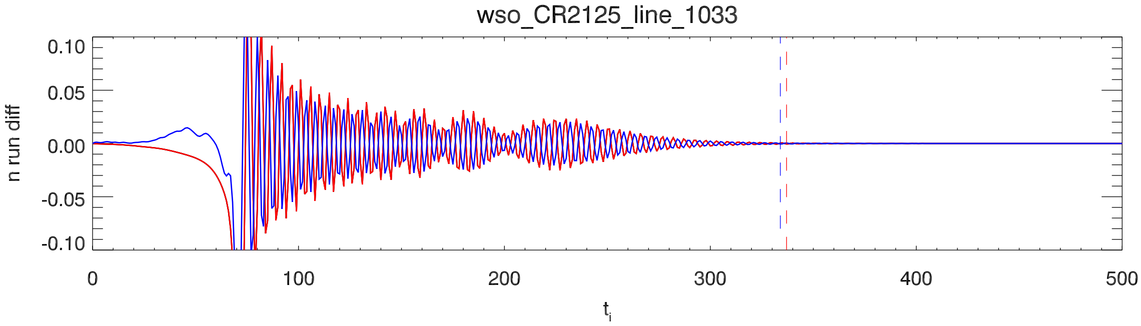

wso_CR2056_line_2079
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

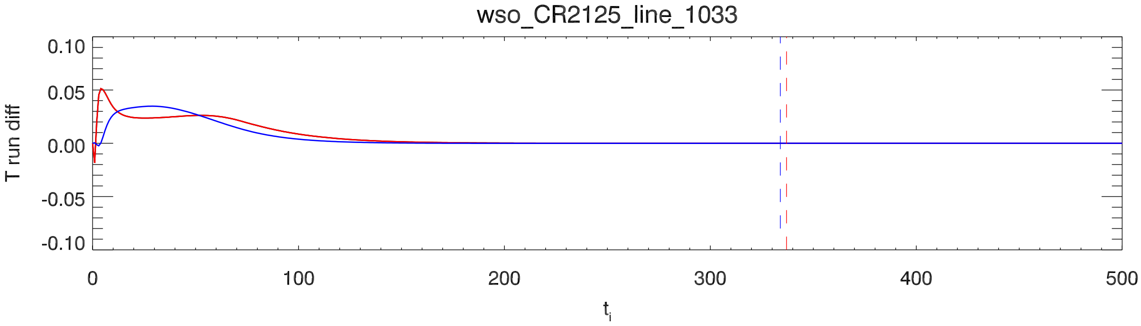

wso_CR2125_line_0040
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

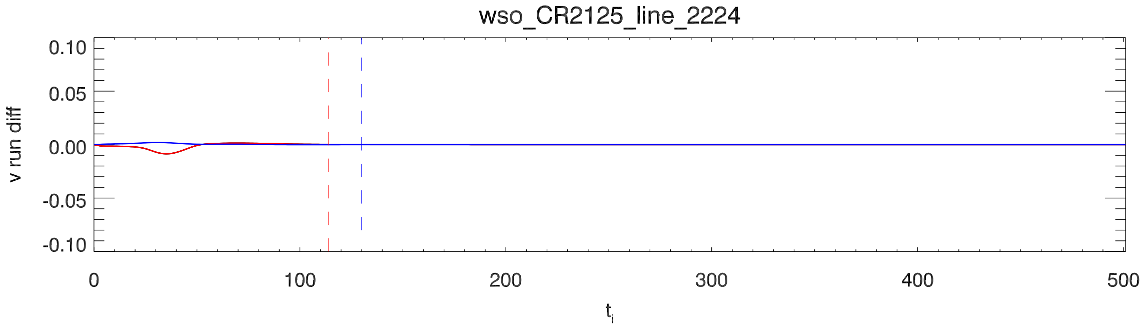

wso_CR2125_line_0040
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

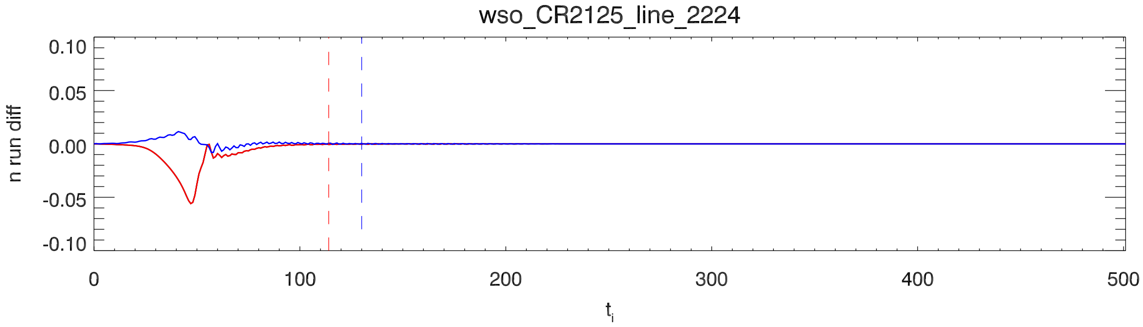

wso_CR2125_line_0040
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

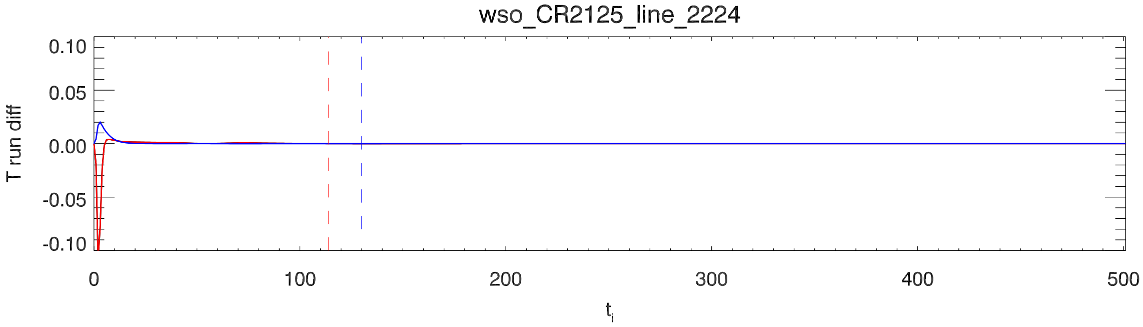

wso_CR2125_line_1033
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

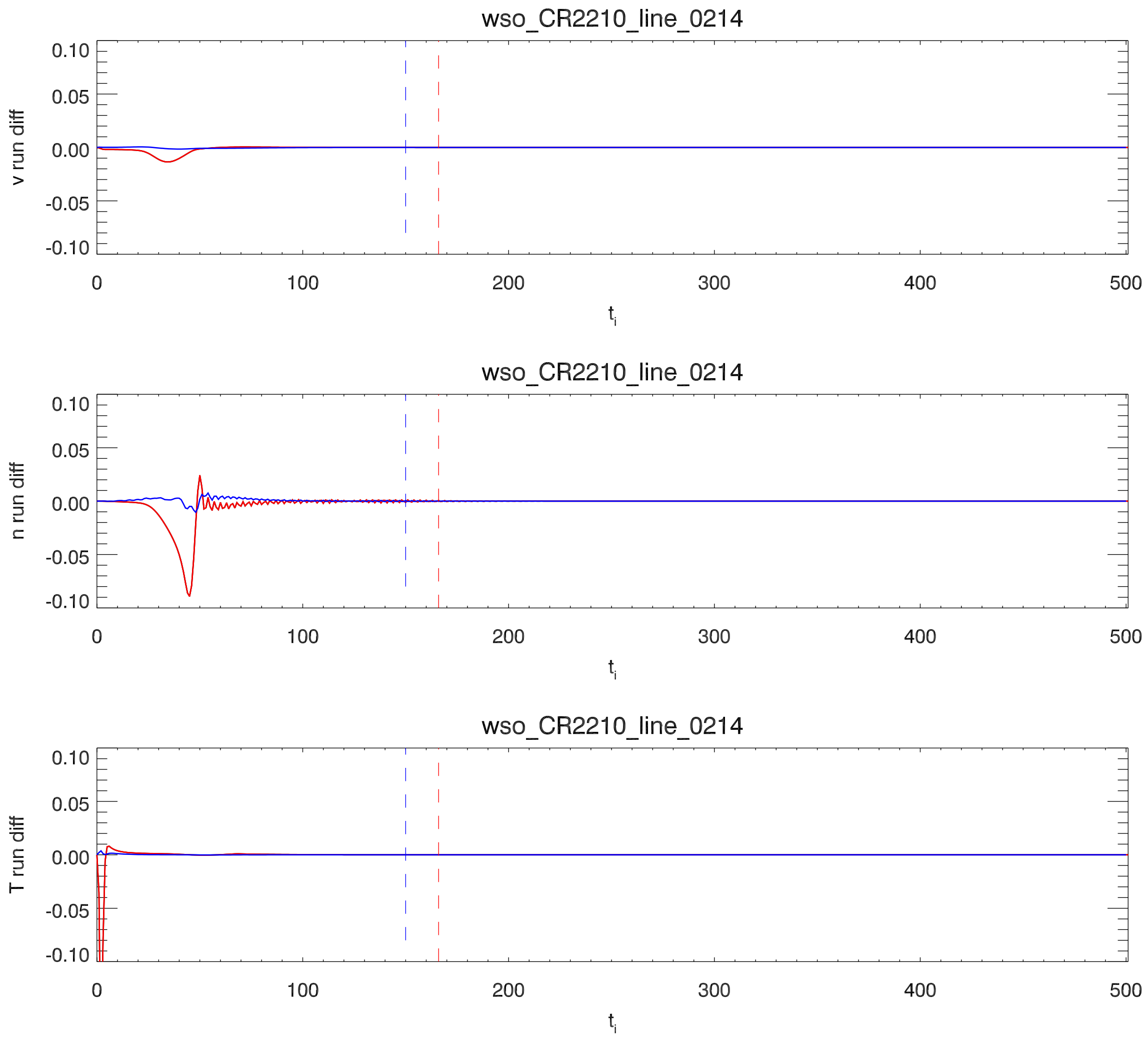

wso_CR2125_line_1033
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2125_line_1033
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2125_line_2224
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

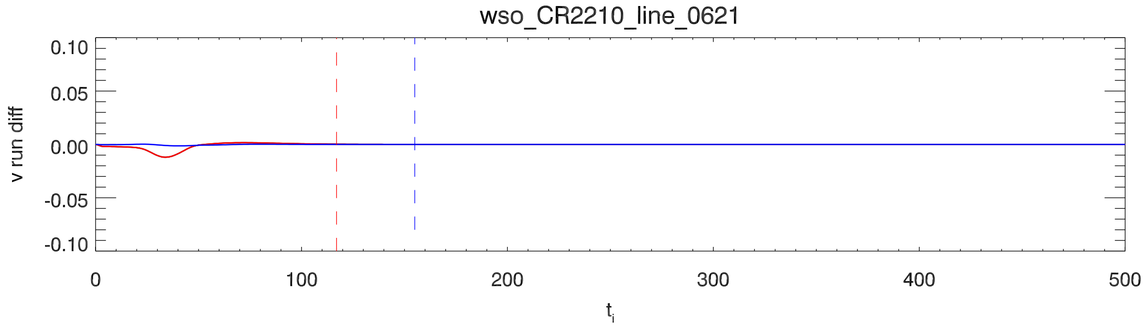

wso_CR2125_line_2224
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

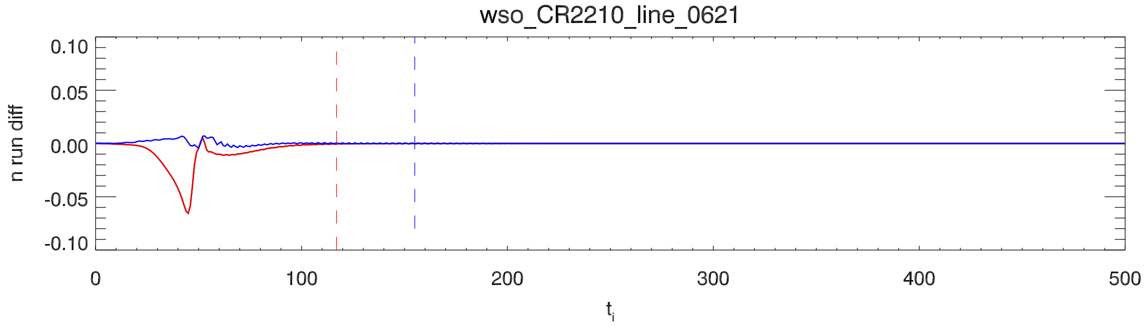

wso_CR2125_line_2224
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

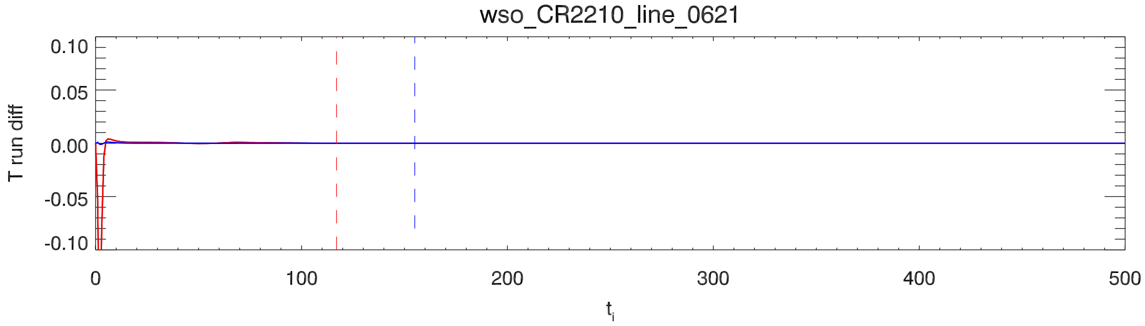

wso_CR2210_line_0214

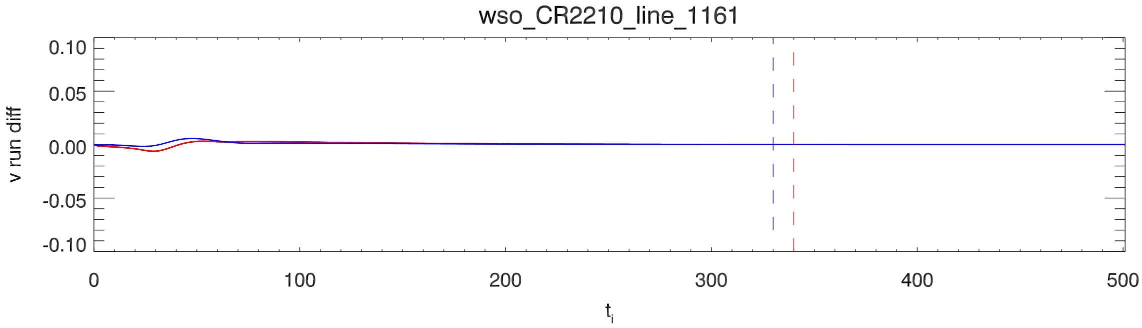
wso_CR2210_line_0214

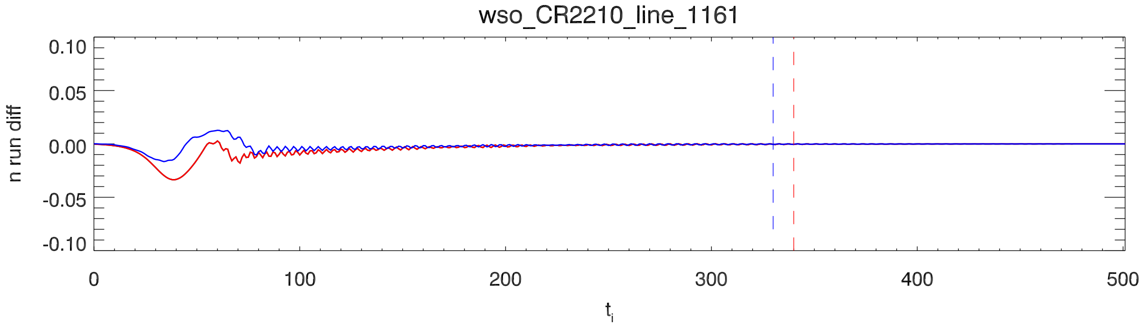
wso_CR2210_line_0214

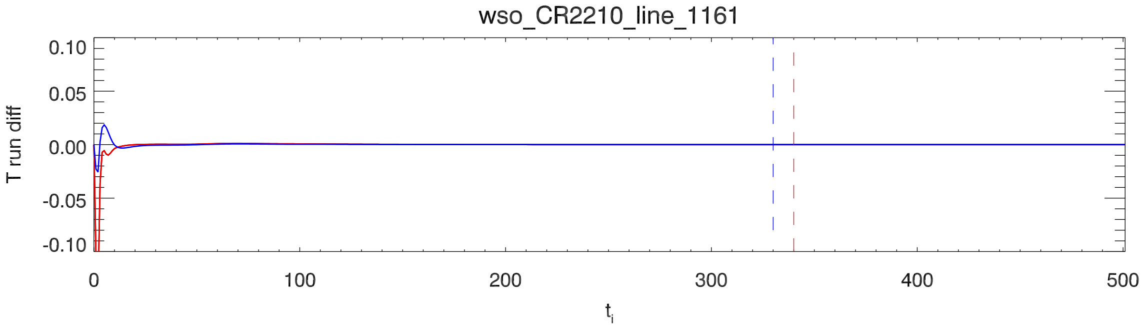

wso_CR2210_line_0621
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

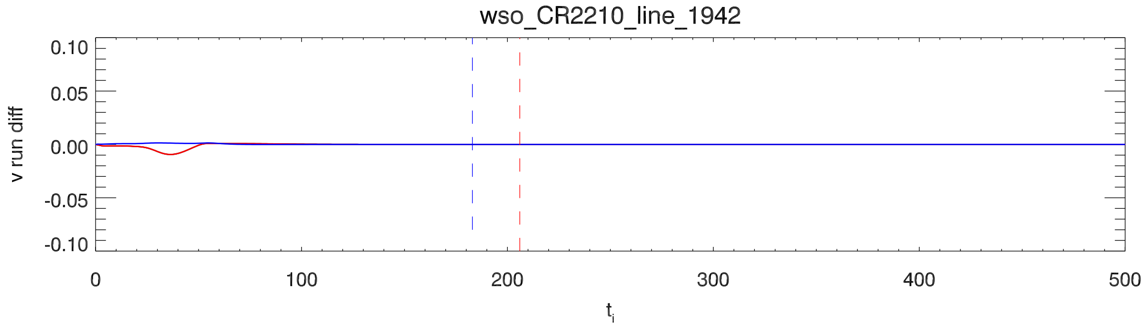

wso_CR2210_line_0621
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

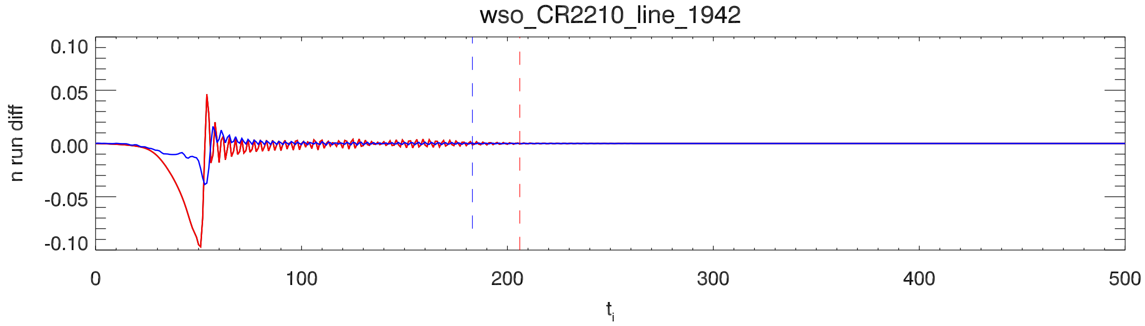

wso_CR2210_line_0621
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2210_line_1161
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2210_line_1161
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2210_line_1161
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2210_line_1942
v run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i


wso_CR2210_line_1942
n run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

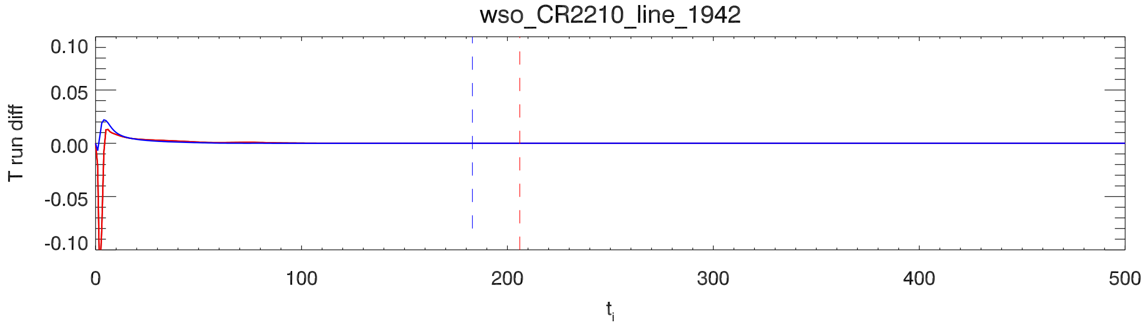

wso_CR2210_line_1942
T run diff
0.10
0.05
0.00
-0.05
-0.10
0
100
200
300
400
500
t_i

Table D.1: Analyzed run, the detected convergence times for each case, the respective number of iterations, and the speedup factor. This speedup factor was calculated by using the simple formula $\text{speed\_fac} = \frac{\text{standard\_number\_iterations}}{\text{predicted\_number\_iterations}}$.

Files	standard times	predicted times	standard iterations	predicted iterations	speedup
CR1992_line_0001	179	170	2706560	2537278	1.066718
CR1992_line_0704	244	246	2730976	2612230	1.045458
CR1992_line_1701	177	187	440005	459398	0.957786
CR1992_line_2441	155	77	859138	422883	2.031621
CR2056_line_1213	232	226	3258564	3153111	1.033444
CR2056_line_1689	201	171	1385252	1167697	1.186311
CR2056_line_1844	206	181	1598801	1391459	1.149011
CR2056_line_2079	163	111	791122	533737	1.482232
CR2125_line_0040	204	172	1566301	1308297	1.197206
CR2125_line_1033	337	334	4322533	4271646	1.011913
CR2125_line_2224	114	130	720955	817851	0.881524
CR2210_line_0214	166	150	1123877	1007777	1.115204
CR2210_line_0621	117	155	741121	977309	0.758328
CR2210_line_1161	340	330	771988	743205	1.038728
CR2210_line_1942	206	183	1388534	1221461	1.136781

References

[1] Eric Priest. *Magnetohydrodynamics of the Sun*. Cambridge University Press, 2014.

[2] Lawrence W Townsend, DL Stephens Jr, JL Hoff, EN Zapp, HM Moussa, TM Miller, CE Campbell, and TF Nichols. The carrington event: possible doses to crews in space from a comparable event. *Advances in Space Research*, 38(2):226–231, 2006.

[3] Rainer Schwenn. Space weather: The solar perspective. *Living Reviews in Solar Physics*, 3(1):2, 2006.

[4] Sami K Solanki, Bernd Inhester, and Manfred Schüssler. The solar magnetic field. *Reports on Progress in Physics*, 69(3):563, 2006.

[5] RF Pinto, AS Brun, and AP Rouillard. Flux-tube geometry and solar wind speed during an activity cycle. *Astronomy & Astrophysics*, 592:A65, 2016.

[6] Stephen Marsland. *Machine learning: an algorithmic perspective*. CRC press, 2015.

[7] Iqbal Muhammad and Zhu Yan. Supervised machine learning approaches: A survey. *ICTACT Journal on Soft Computing*, 5(3), 2015.

[8] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.

[9] Norman R Draper and Harry Smith. *Applied regression analysis*, volume 326. John Wiley & Sons, 1998.

[10] Tom M Mitchell, Jaime G Carbonell, and Ryszard S Michalski. *Machine learning: a guide to current research*, volume 12. Springer Science & Business Media, 1986.

[11] Rui Xu and Don Wunsch. *Clustering*, volume 10. John Wiley & Sons, 2008.

[12] Marcelo Finger and Glauber De Bona. A logic based algorithm for solving probabilistic satisfiability. In *Ibero-American Conference on Artificial Intelligence*, pages 453–462. Springer, 2010.

[13] Lior Rokach and Oded Z Maimon. *Data mining with decision trees: theory and applications*, volume 69. World scientific, 2008.

[14] Sotiris B Kotsiantis. Decision trees: a recent overview. *Artificial Intelligence Review*, 39(4):261–283, 2013.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[16] Bayya Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[17] Jason Weston, Sayan Mukherjee, Olivier Chapelle, Massimiliano Pontil, Tomaso Poggio, and Vladimir Vapnik. Feature selection for svms. In *Advances in neural information processing systems*, pages 668–674, 2001.

[18] Daniel Berrar. Cross-validation. *Encyclopedia of bioinformatics and computational biology*, 1:542–545, 2019.

[19] Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*, volume 26. Springer, 2013.

[20] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[21] MS Wheatland. A bayesian approach to solar flare prediction. *The Astrophysical Journal*, 609(2):1134, 2004.

[22] T Bai. Variability of the occurrence frequency of solar flares as a function of peak hard x-ray rate. *The Astrophysical Journal*, 404:805–809, 1993.

[23] John S Neal and Lawrence W Townsend. Predicting dose-time profiles of solar energetic particle events using bayesian forecasting methods. *IEEE transactions on nuclear science*, 48(6):2004–2009, 2001.

[24] Seth Jonas, Kassandra Fronczyk, and Lucas M Pratt. A framework to understand extreme space weather event probability. *Risk Analysis*, 38(8):1534–1540, 2018.

[25] JS Neal and LW Townsend. Multiple solar particle event dose time profile predictions using bayesian inference. *Radiation protection dosimetry*, 116(1-4):38–42, 2005.

[26] MS Wheatland. A statistical solar flare forecast method. *Space Weather*, 3(7):1–11, 2005.

[27] AS Parnowski. Regression modeling method of space weather prediction. *Astrophysics and Space Science*, 323(2):169–180, 2009.

[28] N. Srivastava. A logistic regression model for predicting the occurrence of intense geomagnetic storms. *Annales Geophysicae*, 23(9):2969–2974, 2005. URL: https://www.ann-geophys.net/23/2969/2005/, doi:10.5194/angeo-23-2969-2005.

[29] H-L Wei, SA Billings, A Surjalal Sharma, S Wing, RJ Boynton, and SN Walker. Forecasting relativistic electron flux using dynamic multiple regression models. In *Annales geophysicae*, page 415. Copernicus GmbH, 2011.

[30] IJ Leontaritis and Stephen A Billings. Input-output parametric models for non-linear systems part i: deterministic non-linear systems. *International journal of control*, 41(2):303–328, 1985.

[31] IJ Leontaritis and Steve A Billings. Input-output parametric models for non-linear systems part ii: stochastic non-linear systems. *International journal of control*, 41(2):329–344, 1985.

[32] SA Billings and QM Zhu. Model validation tests for multivariable nonlinear models including neural networks. *International Journal of Control*, 62(4):749–766, 1995.

[33] AY Ukhorskiy, MI Sitnov, AS Sharma, BJ Anderson, S Ohtani, and ATY Lui. Data-derived forecasting model for relativistic electron intensity at geosynchronous orbit. *Geophysical research letters*, 31(9), 2004.

[34] Rong Li, Huaning Wang, YanMei Cui, and Xin Huang. Solar flare forecasting using learning vector quantity and unsupervised clustering techniques. *Science China Physics, Mechanics and Astronomy*, 54(8):1546–1552, 2011.

[35] Kevin R Moon, Véronique Delouille, Jimmy J Li, Ruben De Visscher, Fraser Watson, and Alfred O Hero. Image patch analysis of sunspots and active regions-ii. clustering via matrix factorization. *Journal of Space Weather and Space Climate*, 6:A3, 2016.

[36] AJ Engell, DA Falconer, M Schuh, J Loomis, and D Bissett. Sprints: A framework for solar-driven event forecasting and research. *Space Weather*, 15(10):1321–1346, 2017.

[37] Kangwoo Yi, Yong-Jae Moon, Gyungin Shin, and Daye Lim. Forecast of major solar x-ray flare flux profiles using novel deep learning models. *The Astrophysical Journal Letters*, 890(1):L5, 2020.

[38] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[39] Alberto Luceño and Daniel Peña. Autoregressive integrated moving average (arima) modeling. *Encyclopedia of Statistics in Quality and Reliability*, 2, 2008.

[40] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[41] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[42] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[43] F Valach, M Revallo, J Bochníček, and P Hejda. Solar energetic particle flux enhancement as a predictor of geomagnetic activity in a neural network-based model. *Space Weather*, 7(4), 2009.

[44] J Uwamahoro, LA McKinnell, and JB Habarulema. Estimating the geoeffectiveness of halo cmes from associated solar and ip parameters using neural networks. In *Annales Geophysicae*, page 963. Copernicus GmbH, 2012.

[45] Henrik Lundstedt. Neural networks and predictions of solar-terrestrial effects. *Planetary and Space Science*, 40:457–464, 1992.

[46] HN Wang, YM Cui, R Li, LY Zhang, and H Han. Solar flare forecasting model supported with artificial neural network techniques. *Advances in Space Research*, 42(9):1464–1468, 2008.

[47] Davor Sudar, Bojan Vršnak, and Mateja Dumbović. Predicting coronal mass ejections transit times to earth with neural network. *Monthly Notices of the Royal Astronomical Society*, 456(2):1542–1548, 2015.

[48] Jon Vandegriff, Kiri Wagstaff, George Ho, and Janice Plauger. Forecasting space weather: Predicting interplanetary shocks using neural networks. *Advances in Space Research*, 36(12):2323–2327, 2005.

[49] Fridrich Valach, Pavel Hejda, and Josef Bochníček. Geoeffectiveness of xra events associated with rsp ii and/or rsp iv estimated using the artificial neural network. *Studia Geophysica et Geodaetica*, 51(4):551–562, 2007.

[50] Tufan Colak and R Qahwaji. Automated solar activity prediction: a hybrid computer platform using machine learning and solar imaging for automated prediction of solar flares. *Space Weather*, 7(6), 2009.

[51] Jian-Guo Wu and Henrik Lundstedt. Geomagnetic storm predictions from solar wind data with the use of dynamic neural networks. *Journal of Geophysical Research: Space Physics*, 102(A7):14255–14268, 1997.

[52] Yi Yang, Fang Shen, Zicai Yang, and Xueshang Feng. Prediction of solar wind speed at 1 au using an artificial neural network. *Space Weather*, 16(9):1227–1244, 2018.

[53] DI Okoh, GK Seemala, AB Rabiu, J Uwamahoro, JB Habarulema, and M Aggarwal. A hybrid regression-neural network (hr-nn) method for forecasting the solar activity. *Space Weather*, 16(9):1424–1436, 2018.

[54] Andrew J Conway. Time series, neural networks and the future of the sun. *New Astronomy Reviews*, 42(5):343–394, 1998.

[55] Thomas Bayes. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, 53:370–418, 1763.

[56] Fadil Inceoglu, Jacob H Jeppesen, Peter Kongstad, Néstor J Hernández Marcano, Rune H Jacobsen, and Christoffer Karoff. Using machine learning methods to forecast if solar flares will be associated with cmes and seps. *The Astrophysical Journal*, 861(2):128, 2018.

[57] Jiajia Liu, Yudong Ye, Chenglong Shen, Yuming Wang, and Robert Erdélyi. A new tool for cme arrival time prediction using machine learning algorithms: Cat-puma. *The Astrophysical Journal*, 855(2):109, 2018.

[58] Yu Jiao, John J Hall, and Yu T Morton. Automatic equatorial gps amplitude scintillation detection using a machine learning algorithm. *IEEE Transactions on Aerospace and Electronic Systems*, 53(1):405–418, 2017.

[59] Dmitrii Kochkov, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21), 2021.

[60] BY Lattimer, JL Hodges, and AM Lattimer. Using machine learning in physics-based simulation of fire. *Fire Safety Journal*, 114:102991, 2020.

[61] Peter AG Watson. Applying machine learning to improve simulations of a chaotic dynamical system using empirical error correction. *Journal of Advances in Modeling Earth Systems*, 11(5):1402–1417, 2019.

[62] Hakan Tongal and Martijn J Booij. Simulation and forecasting of streamflows using machine learning models coupled with base flow separation. *Journal of hydrology*, 564:266–282, 2018.

[63] Oliver Hahn and Tom Abel. Multi-scale initial conditions for cosmological simulations. *Monthly Notices of the Royal Astronomical Society*, 415(3):2101–2121, 2011.

[64] Martín Crocce, Sebastián Pueblas, and Román Scoccimarro. Transients from initial conditions in cosmological simulations. *Monthly Notices of the Royal Astronomical Society*, 373(1):369–381, 2006.

[65] S Prunet, C Pichon, D Aubert, D Pogosyan, R Teyssier, and S Gottloeber. Initial conditions for large cosmological simulations. *The astrophysical journal supplement series*, 178(2):179, 2008.

[66] Gillen Brown and Oleg Y Gnedin. Improving performance of zoom-in cosmological simulations using initial conditions with customized grids. *New Astronomy*, 84:101501, 2021.

[67] Jens Jasche and Benjamin D Wandelt. Bayesian physical reconstruction of initial conditions from large-scale structure surveys. *Monthly Notices of the Royal Astronomical Society*, 432(2):894–913, 2013.

[68] Rui F Pinto and Alexis P Rouillard. A multiple flux-tube solar wind model. *The Astrophysical Journal*, 838(2):89, 2017.

[69] Marvin V Zelkowitz and Dolores R. Wallace. Experimental models for validating technology. *Computer*, 31(5):23–31, 1998.

[70] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.

[71] Richard Arnold Johnson, Dean W Wichern, et al. *Applied multivariate statistical analysis*. Prentice hall Upper Saddle River, NJ, 2002.

[72] Karl Pearson. Vii. note on regression and inheritance in the case of two parents. *proceedings of the royal society of London*, 58(347-352):240–242, 1895.

[73] Jorge Sola and Joaquin Sevilla. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on nuclear science*, 44(3):1464–1468, 1997.

[74] Keras Team. Keras documentation: The sequential model. URL: https://keras.io/guides/sequential_model/.

[75] D Stathakis. How many hidden layers and nodes? *International Journal of Remote Sensing*, 30(8):2133–2147, 2009.

[76] Russell Reed and Robert J MarksII. *Neural smithing: supervised learning in feedforward artificial neural networks*. Mit Press, 1999.

[77] Keras Team. Keras documentation: Kerastuner. URL: https://keras.io/keras_tuner/.

[78] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[79] Damir Kalpić, Nikica Hlupić, and Miodrag Lovrić. *Student's t-Tests*, pages 1559–1563. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. URL: https://doi.org/10.1007/978-3-642-04898-2_641, doi:10.1007/978-3-642-04898-2_641.

[80] Donald T Campbell and Thomas D Cook. Quasi-experimentation. *Chicago, IL: Rand McNally*, 1979.

[81] Donald T Campbell and Julian C Stanley. *Experimental and quasi-experimental designs for research*. Ravenio Books, 2015.

[82] Jake VanderPlas. *Python data science handbook: Essential tools for working with data*. " O'Reilly Media, Inc.", 2016.