Thermosphere Modelling Using Machine Learning

Using the Swarm satellite constellation as study case

J.M.H. Claes

MSc thesis Aerospace Engineering

Delft University of Technology

TU Delft

Cover Image:
One of the Swarm trio orbiting over Italy.
Source online accessed on 13-March-2019:
https://www.rocketstem.org/2014/01/15/
esa-swarm-trio-monitoring-planets-magnetic-shield/

AE5810 - Final Thesis
Thermosphere Modelling using Machine Learning

Artificial Neural Networks to forecast thermospheric densities and generalise beyond the properties of an acceleration data set using the Swarm satellites as study case.

by

J. M. H. Claes

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on Monday 1, July 2019 at 14:30.

Student number: 4208048
Project duration: June 20, 2018 - July 1, 2019

Supervisor: Dr. Ir. J.A.A. (Jose) van den IJssel
Dr. Ir. E.N. (Eelco) Doornbos

Thesis committee: Prof. Dr. Ir. P.N.A.M. (Pieter) Visser
Dr. Ir. J.A.A. (Jose) van den IJssel
Dr. Ir. R. (Róbert) Fónod

An electronic version of this thesis is available at: https://repository.tudelft.nl/

To my family and dearest friends,
who kept me motivated throughout my years at university,
and who provided continuous encouragement,
that resulted in reaching the end of this research project.
Thank you.

Jochem Claes
Delft, 2019

Preface

This report covers the thesis research project as part of my MSc Aerospace Engineering degree at the Delft University of Technology, the Netherlands. It took place at the Faculty of Aerospace Engineering at the Delft University of Technology in Delft, the Netherlands, academic year 2018-2019. This thesis study report will contribute to the intellectual legacy of the Space Engineering department of the faculty.

I fell in love with the interactions taking place between the Earth's atmosphere and the space debris objects tumbling through space during my two internships at the Space Debris Departments of the European Space Agency and the Japanese Aerospace Exploration Agency. For this reason, I started looking for a supervisor who could provide me with a graduation topic in this field. It was therefore very exciting that my thesis project was going to be devoted to understanding and forecasting space weather and other interactions of solar processes with the Earth environment. My supervisor challenged me with the goal to assess the ability of the current empirical thermosphere models to improve the forecast performance of the thermospheric neutral density. This density directly affects the drag force working on a satellite body flying through the thermosphere. Since novel methods in the machine learning and data science space are becoming more popular and effective, due to the increase in data and computational power, I challenged myself to see what opportunities could be found in this field.

Throughout this research project, I got the opportunity to familiarise myself with two rather new and challenging topics: the thermosphere and its complex interactions with the Sun, as well as the machine learning algorithms needed to solve regression problems. The latter one definitely pushed me outside my comfort zone, since this field of expertise is most often studied at the Faculty of Computer Science and is not part of the Space Flight curriculum. However, I do not regret to have worked on this topic, given the attention it receives from large research institutes and corporations.

I would never have succeeded without the help of my two daily supervisors, Dr. Ir. Eelco Doornbos and Dr. Ir. Jose van den Ijssel. I would like to thank both supervisors for the great teamwork, interesting discussions and progress made throughout each phase of my thesis project. They combined both a supportive, as well as a critical attitude on the topics that challenged me the most. This eventually led to the results written in this report. A piece of work which I will look back at and feel proud off in the coming few decades. I look forward to tackling the upcoming challenges in my life as an engineer who graduated from the Delft University of Technology.

Jochem Claes
Delft, 2019

Executive Summary

The Low Earth Orbit (LEO) region has been attractive to many space agencies and organisations because of its ease of access and the ideal opportunity for remote sensing due to the low altitudes in combination with short orbital periods. As a logical consequence, the LEO region is the most crowded near-Earth space region with an increase in the amount of space debris. This has the risk that active missions are more often brought in a state of alert when they are on a potential crash course. Having the ability to now- and forecast the orbital state of a satellite to a high degree of accuracy could potentially avoid in-orbit collisions from happening. Due to the low altitudes, a satellite's orbital state is highly affected by the atmospheric drag force acting on the satellite's body. The drag force is considered the limiting factor in the accuracy of orbit determination and prediction of LEO satellites. The largest variation in this drag force is caused by the changes in thermospheric density due to the complex interactions of the Sun with the Earth's thermosphere. Thus, in order to properly forecast the orbital state of a LEO satellite, the thermospheric densities need to be predicted as accurately as possible.

The thermospheric density values can be estimated using empirical atmospheric density models, such as the DTM2013 (Drag Temperature Model). The DTM2013's output is based on climatologies derived from observations, by trying to fit parametric data to those observations. Additionally, it requires inputs that give information on the orbital state as well as solar and geomagnetic indices, which are indicators of atmospheric density. If one wants to forecast the density values that a LEO satellite encounters along its orbit then those input values need to be forecasted as well, introducing additional error sources to the model. During this thesis study it has been investigated whether the highly researched field of machine learning models could be used to develop a predictor for the along-track density values for the Swarm satellite constellation. This constellation has an abundant amount of trajectory-based time series of thermospheric density values from Precise Orbit Determination (POD) data.

This research has focused on the development of Multi-layer Perceptron (MLP) models which are a type of Feed-Forward Neural Network. These MLP models have been trained and tested on past acceleration and solar activity data sets provided by the Swarm satellite mission and space weather observatories, respectively. The POD-derived density values are seen as the target values during training, while the input of the univariate MLP models is a vector of past DTM2013 density data. For the multivariate MLP models, additional indices that capture part of the solar and geomagnetic activity are added to the input vector. The architecture of the MLP models has been set-up by using the Swarm-A density data exclusively. The density data of Swarm-B and Swarm-C is finally used to examine if the MLP models were able to generalise beyond the properties of the training data set of Swarm-A for different solar and geomagnetic activity. The performance of these MLP models was then compared to two baseline models, namely a Calibrated Persistence Model (CPM) and the density values modelled by DTM2013.

The results in this research have demonstrated that an MLP model with three hidden layers does perform best when compared to other MLP architectures. The first hidden layer of this specific model incorporates eight neurons, while both the second and third hidden layer have six neurons each. Moreover, the accuracy for daily forecasts of the thermospheric density values was found optimal for a sampling period of 120 s and ten orbital periods worth of delay variables in the input vector. This three-layer MLP model performed best when it was trained on data of the same spacecraft like the one it was supposed to perform along-track density forecasts for. The forecasting accuracy increased the most when the model was trained on long periods of training data characterised by high solar and low geomagnetic activity. When trained on these data sets, the MLP model has shown to outperform the baseline models when making predictions up until two days into the future during periods of high solar activity. However, the MLP models were not able to perform well for the test periods with low solar activity. If one would still like to use MLP models during these periods of low solar activity it is recommended to add the 15 cm solar radio flux to the input vector. This additional information helps to make the MLP model more robust to varying levels of solar activity.

The DTM2013 seems the best option to forecast density values during low solar activity. This conclusion was reached by introducing the known and observed input variables (eg. orbital state, and solar and geomagnetic indices) as input variables to the DTM2013, hence making the DTM2013 values used in this report the most optimistic DTM2013 forecasts. Note that in reality, one would need to forecast the orbital position and indices first before being able to make a forecast with this empirical density model. As an alternative, the CPM seems a suitable model when one needs to quickly implement a forecasting model with decent performance irrespective from the presence of geomagnetic storms. This thesis has shown the promising potential of using machine learning techniques to forecast the along-track density values of a LEO satellite. Machine learning is currently one of the state-of-the-art solutions for many problems in the science and engineering space. Future development of these techniques is suggested with the main goal to first improve the reliability and robustness of the neural network model before it is introduced to a new set of promising applications.

Contents

List of Figures

List of Tables

List of Acronyms

Acronym	Description
ACF	Autocorrelation Function
ADAM	Adaptive Moment Estimation
ADF	Augmented Dickey-Fuller
AI	Artificial Intelligence
ANN	Artificial Neural Network
AR	Auto Regression
ARIMA	Auto Regressive Integrated Moving Average
ATMOP	Advanced Thermosphere Modelling for Orbit Prediction
CCMC	Community Coordinated Modeling Center
CET	Central Eastern Time
CLS	Collecte Localisation Satellites
C(D)NN	Convolutional (Deep) Neural Networks
CPM	Calibrated Persistence Model
DTM	Drag Temperature Model
ESA	European Space Agency
EU	European Union
EUV	Extreme Ultra-Violet
LEO	Low Earth Orbit
LT	Long Term
LSTM	Long Short-Term Model
MA	Moving Average
ML	Machine Learning
MLP	Multilayer Perceptron
MTD	Modified Taylor Diagram
NASA	National Aeronautics and Space Administration
NN	Neural Network
NOAA	National Oceanic and Atmospheric Administration
POD	Precise Orbit Determination
PDF	Probability Density Function
POE	Precision Orbit Ephemeris
RELU	Rectified Linear Unit
RMSE	Root Mean Square Error
S/C	Spacecraft
ST	Short Term
SW	Space Weather
TS	Thermosphere

List of Symbols

Greek Symbols	**Description**	**Units**
ψ	Nonlinear activation function for neural network models	-
ρ	Density; Swarm along-track Precise Orbit Determination derived density	kg/m^3
$\hat{\rho}$	Predicted thermospheric model density	kg/m^3
$\rho_{DTM2013}$	DTM2013 thermospheric model density	kg/m^3
Roman Symbols	**Description**	**Units**
A	Satellite's cross-sectional area perpendicular to atmosphere's relative velocity	m^2
a	Scale factor in scaled persistence model	-
a_{drag}	Acceleration imparted on satellite from atmospheric drag	m/s^2
a_p	Planetary equivalent amplitude	-
b	Bias in scaled persistence model	kg/m^3
v_{rel}	Velocity of satellite relative to moving atmosphere	m/s
C_D	Drag coefficient	-
D_{st}	Geomagnetic equatorial index	nT
h	Horizontal displacement in scaled persistence model	s
J	Cost function	-
m	Spacecraft mass	kg
T_∞	Exospheric temperature	K
t_s	Sampling period	s
K_p	Planetary K-index	-
F	Solar radio flux at specified wavelength	sfu
R	Correlation	-
W_p	Prediction window	s

Chapter 1

Introduction

As a result of historical space operations and in-orbit collisions, and despite the adoption of several mitigation measures, it is likely that the population of orbiting objects has reached a critical point whereby growth will continue even in the absence of future space launches [*Lewis et al.*, 2016]. Over the years, the risk of space debris is increasing. Active missions are more often brought in a state of alert when they are on a potential crash course. This study case, among other applications such as re-entry procedures, mission manoeuvring, space object surveillance and tracking all require a high degree of accuracy in now- and forecast capabilities of the orbital state. Precise satellite trajectory prediction capabilities need to be enhanced to keep up with the more stringent requirements concerning localisation techniques both in space and on the ground. Orbit determination and prediction is a complex field in which a multitude of assumptions, measurement data and models come together to make an estimation of the current and a prediction of the future satellite position. In the ideal case of a perfect mathematical model, the nonzero difference between the predicted values and true measurements should be due to instrument noise. Since the real-life satellite force environment needs to be approximated, additional error sources will be caught in the estimated values.
One of the main contributors in the error budget during orbit determination for Low Earth Orbit (LEO) satellites is the modelling of the drag force, due to the interaction between the satellite and the medium it is flying through, the thermosphere [*Storz et al.*, 2005]. The thermosphere is a layer of the Earth's atmosphere and is both influenced by the solar activity as the interaction with its adjacent atmospheric layers, the ionosphere and mesosphere [*Weimer et al.*, 2016].

This introductory chapter will provide a short discussion on the thermosphere and its relevant characteristics. Section 1.1 will illustrate the context needed to kick-off the more technical details discussed in this report. Section 1.2 will then elaborate on the motivation, scope and research questions that are going to be answered in this research study. The final section, Section 1.3, will provide an outline of the structure of this report.

1.1 Problem Statement

In LEO the drag force is the second most dominant force after gravity [*Pérez and Bevilacqua*, 2015]. In this research study the drag force will be examined as being uncoupled from the other external forces. The effect of the drag force on a satellite body is due to two main working principles. The first one being the properties of the atmosphere it is flying through. The second being the interaction between the atmosphere and the satellite body. However, in the study by *Storz et al.* [2005] it can be read that re-

quirements for satellite prediction accuracy are not consistently met due to the lacking performance of the atmospheric density models, causing errors of 15-20%. Herein, the mass density of Earth's thermosphere is a critical parameter for LEO prediction because of the atmospheric drag on satellites in this region [*Emmert*, 2009]. As a consequence, techniques will be studied that help to improve the atmospheric state prediction capabilities of state-of-the-art thermospheric density models. In this section the thermosphere, forecasting techniques and density measurements will be shortly discussed.

1.1.1 Thermosphere Modelling

The atmosphere around our planet is a complex-to-model collection of different gasses, kept in place by the gravity of the planet. The variability of the atmospheric temperature causes the density to change accordingly. At heights above about 200 km, the temperature asymptotically approaches a limiting value. This limiting temperature value is often called the exospheric temperature, T_∞ [*Weimer et al.*, 2016]. The mass density becomes smaller than a billionth of the air density at sea level at an altitude of 200 km. The atmospheric layer of interest will be the thermosphere, which forms the upper part of Earth's atmosphere (~ 90-600 km altitude) and directly affects the satellites flying in a LEO. The state of the thermosphere on a certain time and location is defined by the temperature, density, wind and its composition. In terms of composition, the thermosphere consists of partially ionised gas constituents consisting primarily of N_2, O_2, O, He and H. How the state of the thermosphere varies temporally and spatially depends on the interaction of the thermosphere with external energy sources. The most important sources of heat are solar EUV radiation, related to solar activity, and energetic charged particles, related to geomagnetic activity caused by this continuous stream of particles emitted by the Sun [*Doornbos*, 2012]. Other sources of heating or cooling and transport mechanisms are outside the scope of this report. In spite of the low density values, a drag force can still be experienced due to the high orbital velocities of the satellites flying through this thermospheric layer. The main focus of this research study will be on the upper part of the thermosphere (~ 200-600 km altitude). From this point onwards, the upper thermosphere is referred to as 'thermosphere'.

Atmospheric density models are required in the computation of the state of the thermosphere. The Community Coordinated Modeling Center (CCMC) is a multi-agency partnership which provides a gateway to modern space science simulations, to the international research community. What is of interest is the support they provide for the transition to space weather operations of modern space research models [*CCMC*, 2018]. Looking at the ionosphere-thermosphere models they present, a direct classification between physical and empirical models is defined. The differences between both type of models are explained next and do have an influence on the type of improvements that can be applied to those models.

Empirical models of density are based on climatologies derived from observations, by trying to fit parametric equations to this data [*Kodikara et al.*, 2018]. Physical models on the other hand numerically solve the fluid equations to derive the evolution of parameters such as thermospheric density, temperature, wind and composition [*Qian and Solomon*, 2012]. However, the distinction between physical and empirical models is not absolute. For example, empirical models will use physical constraints in their models while physical models also use observation data to set boundary conditions. For the further extent of the report, both are considered different. Semi-empirical models will be named empirical models from this point onward. The type of model to use is application dependent. Empirical models are most widely used in operational environments for orbit tracking and short-term forecasting. Physical models are not often used for short-term forecasting due to the computationally expensive calculations. However, due to the increase in computational power and new studies, the physical models are more and more getting developed for operational applications. An overview of and discussion on the different physical and

empirical models can be found in the literature study report by *Claes* [2018], which preceded this thesis study. Since orbital state nowcasting and forecasting is the main goal of this thesis project, improving the empirical models will be the main focus of this report. This will also allow for more efficient prototyping when using a personal computer.

The model used in this report is a semi-empirical thermosphere model, called the Drag Temperature Model (DTM2013), and will be put to use throughout the further extent of this report. This is to model and predict the along-track thermospheric density [*Bruinsma*, 2015]. DTM2013 is an end product of the Horizon 2020 programme, which is the biggest EU Research and Innovation programme to keep improving the European space weather forecasting abilities. Within the ATMOP [1] (Advanced Thermosphere Modelling for Orbit Prediction) project the existing DTM models were improved by assimilating GOCE (Gravity field and steady-state Ocean Circulation Explorer) satellite data, as an example. The DTM2013 was picked because of its performance, the assimilation of space-based observational data and the availability at the Faculty of Aerospace Engineering. From this point onwards, the neutral mass density of the thermosphere will be abbreviated to density.

The two main variables that make developments in both empirical and physical models possible are the availability of (observational) data and the computational power of electronic devices. With the increase of the quality and amount of observational data, the resolution of the empirical models is increased [*Kodikara et al.*, 2018]. *Pérez et al.* [2014] indicate that few new models are created, since this is time-consuming and complex, but improvements are performed on previously developed models. An overview of the different improvement techniques is summarised in the literature study report. This report concluded that advanced forecasting techniques based on machine learning models are promising due to the combination of the increasing accessibility of space-based data and the proven track record of its efficient networks for many applications. *Camporeale et al.* [2018] state that the purpose of machine learning is to build predictive models. This in comparison to information science and data mining, where the former deals with understanding causality between observations and the latter is effective in discovering nontrivial patterns and correlations in large datasets.
Machine Learning (ML) and Neural Network (NN) techniques in specific have shown to be promising strategies in the field of finance (eg. *Borovykh et al.* [2017]), weather forecasting (eg. *Tang et al.* [2001]) and oceanography (eg. *Hsieh and Tang* [1998]). This thesis study will elaborate on the potential of these advanced techniques in solving Space Weather-related problems. *Yu et al.* [2012] and other scientists started to use Artificial Neural Networks (ANN) to characterise complex space weather phenomena. The nonlinearities in the neurons of such a network could potentially help in forecasting the nonlinear behaviour encaptured in the solar-terrestrial system.

1.1.2 Atmospheric Density

Pérez et al. [2014] indicated that the density is the parameter with the largest variations in modelling the drag force. Hence it is beneficial to improve the density output of those atmospheric models. In contrast to modelling the thermospheric density, as discussed in the previous section, the density can also be measured. The mass density can be measured via trajectory observations or via on-board instruments such as accelerometers. The former is used to present the acceleration imparted on the satellite from atmospheric drag as presented in Equation 1.1 [*McClain and Vallado*, 2001].

$$\bar{a}_{drag} = -\frac{1}{2}\rho v_{rel}^2 \frac{\bar{v}_{rel}}{|\bar{v}_{rel}|}\frac{C_D A}{m} \tag{1.1}$$

[1] More information on: `www.atmop.eu`

Here ρ is the atmospheric density, $\bar{v}_{rel}$ the velocity of the satellite relative to the moving atmosphere, m is the spacecraft mass. Finally, A is the satellite's cross-sectional area perpendicular to the atmosphere's relative velocity and C_D the coefficient of drag.

In this study the Swarm satellite constellation is used as test and validation data set due to the abundant availability of trajectory-based time series of neutral thermospheric density from precise orbit determination data. The data sets are publicly available and are the result of the postprocessing efforts of the TU Delft under contract of the European Space Agency (ESA) [*Olsen et al.*, 2013]. The Swarm satellite mission, launched on 22 November 2013, is the ESA's first satellite constellation for Earth observation. The three Swarm satellites are measuring the magnetic signals that stem from Earth's core, mantle, crust and oceans, as well as from the ionosphere and magnetosphere. This state-of-the-art mission will lead to new insight into many natural processes, from those occurring deep inside the planet to weather in space caused by solar activity [2]. All three satellites are identical and were put into orbit at the same time. Swarm-A and -C form the lower pair of satellites flying side-by-side (1.4° separation in longitude at the equator) at an altitude of 462 km (initial altitude) and at 87.35° inclination angle, whereas Swarm-B is cruising at higher orbit of 511 km (initial altitude) and at 87.75° inclination angle [*Stanica et al.*, 2018].

1.2 Research Objective, Scope and Questions

The aim of this thesis was formulated based on the information gathered in Section 1.1 and the literature study report by *Claes* [2018], written prior to this thesis study. The aim is to enhance the capabilities of an established, empirical thermosphere model to increase the accuracy of thermospheric density predictions. To effectively forecast density values with the current state-of-the-art empirical models, a forecast of its input values needs to be achieved first after which the model can be run to compute the future thermospheric state. These input values include orbital state information, as well as solar and geomagnetic indices. However, in this thesis project the goal is set to create a tool to improve forecasting capabilities using incorporated machine learning models. The research objective is to determine if machine learning predictors are a feasible option to improve thermospheric density forecasts to such a degree that they can be used in future space missions and research efforts.

The size of this project would be immense if a scope would not be defined. The size is confined by four main aspects, which are discussed next. Any other assumptions are minor and will be mentioned in the report itself. Firstly, the main focus will be to add value to the academic world in terms of improving the accuracy of the thermospheric density forecasting techniques from a scientific point of view. Thus, not on the analysis of how these new density predictions impact the orbit determination performance directly. Secondly, only past and current acceleration and solar activity data sets provided by the Swarm satellite mission and space weather observatories, respectively, are used to make an offline prediction in the future. Part of the forecasted solar and geomagnetic indices are made available and are often provided by their corresponding observatory, but are not taken into consideration in this study. Thirdly, the models will only be tested on one satellite mission, consisting out of three, identical Swarm satellites. Consequently, the machine learning models can be tested on robustness when the solar activity or/and orbital elements change while keeping the form factor constant. Finally, the machine learning model will need data to learn specific relations. The scope of this research excludes the interpretation of how these relations are learnt during the training of these machine learning models.

[2] More information on: `www.esa.int/Our_Activities/Observing_the_Earth/Swarm/Introducing_Swarm`

To achieve this goal, one main research question is set-up that needs to be answered to check if this goal has been met. The main research question is stated as follows:

Can Artificial Neural Networks (ANNs) forecast thermospheric densities and generalise beyond the properties of an acceleration data set using the Swarm satellites as study case?

This research question is then broken into two sub-questions. When all sub-questions are answered, a conclusive answer can be formed on the research question. For each sub-question, a set of related questions is listed to provide guidance during the analyses presented in this report. The sub-questions are shown below:

1. *What is the overall forecasting performance of an ANN model when predicting the along-track density values of the Swarm satellites?*
    - How do the forecasted density values compare with the POD-derived density values of the Swarm satellites when additional layers and/or neurons per layer are added to the ANN model?
    - What is the change in forecasting performance when the ANN model needs to forecast density values up until three days into the future?
    - How does the performance of an ANN model change if extra input data is added to the input vector; i.e. more past model data, solar EUV proxies or geomagnetic indices?
    - What is the forecasting performance of the ANN model with respect to other baseline models; i.e. persistence and empirical, thermospheric density models?
2. *Can ANNs generalise beyond the properties of a given acceleration training data set for satellites with the same form factor?*
    - Can ANN models generalise beyond the training data in terms of temporal variation (high or low solar activity)?
    - Can ANN models predict the along-track density values of the Swarm satellites during periods of high geomagnetic activity?
    - Can ANN models generalise beyond the training data in terms of spatial variation (orbital elements)?

1.3 Thesis Outline

This thesis report encompasses the process that was undertaken in order to answer the proposed research questions. Chapter 2 discusses all data sources that are required as input for the DTM2013 that will be used during this research process. In Chapter 3, the theory behind all time series forecasting models is elaborated upon. Chapter 4 depicts the steps that need to be taken to preprocess the data sets before they can be inserted in the machine learning models. Chapter 5 will present and discuss the outcome of forecasting the thermospheric density values. The conclusion and the additional recommendations will be listed in Chapter 6.

Chapter 2

Thermosphere Model

In this chapter the chosen atmospheric density model will be explained in detail. The reason for this extensive discussion on the thermospheric model and its input variables is that the quality will highly reflect in the forecasting accuracy of the machine learning models. Section 2.1 depicts the software interactions of the thermosphere model with its inputs and will then zoom in on the internals of the model. In Section 2.2 all the required inputs for the model are listed and elaborated upon. Finally, the model's output will be benchmarked against a validation data set in Section 2.3. This validation part will provide the necessary confidence to use this model in future research.

2.1 Empirical Thermosphere Model

As explained in Section 1.1, only established, empirical thermosphere models are considered as plausible candidates for this research project due to their feasibility in operational applications. However, many different empirical models are currently in use to compute the atmospheric drag force in Precise satellite Orbit Determination (POD) and prediction cases. An overview of all the relevant, state-of-the-art models can be found in the Literature Study report [*Claes*, 2018]. In short, *Bruinsma et al.* [2017] studied and compared the performance of three global models - DTM2013, NRLMSISE-00, JB2008 - for different solar activity. The aforementioned models are most often used for orbit tracking and short-term forecasting. They are global models in a sense that they are able to predict the density of atmospheric gasses that vary with location (longitude, latitude and altitude), time and space weather conditions. Based on this comparison and the research problem at hand, it is concluded that the DTM2013 (Drag Temperature Model) will be the main model used in this research project for the reasons mentioned next.

Firstly, *Bruinsma* [2015] concluded that, overall, the DTM2013 performs best due to the assimilation of more space-based data sets. Data assimilation is a model calibration technique which is used to efficiently incorporate density observations from satellite dynamics or accelerometer measurements in an established thermosphere model. By doing so the model is calibrated internally with the goal to minimise the error between the modelled density values and the observed ones. A set of correction parameters is then used in the internal source code of the old model to directly influence the output of the model. As an example, newly available GOCE data was used to create the DTM2013 as an update for the previously used DTM2009 [*Bruinsma*, 2015]. The DTM2013 required a new inversion to estimate all the parameters internal to this model, to incorporate all the data, including the old data as well as the new large batch of GOCE data. Table 2.1 lists all data sets used in the DTM2013. The second reason is a consequence of the difference in origin of the data sets that are assimilated in this model. The Precise Orbit Determination-

Table 2.1: Data used in the construction of DTM2013 [*Bruinsma*, 2015].

Source	**Measurement**	**Period**
CHAMP	ρ	05/2001-08/2010
GRACE	ρ	01/2003-12/2011
GOCE	ρ	11/2009-05/2012
Starlette and Stella	ρ	01/1994-12/2010
Deimos-1	ρ	03/2010-09/2011
CACTUS	ρ	07/1975-01/1979
OGO6	T	06/1969-08/1975
DE-2	T, He, O, N_2	08/1981-02/1983
AE-C	N_2	01/1974-04/1977
AE-E	T, He, O	12/1975-05/1981

and accelerometer-derived density measurements of the Swarm satellite constellation are not yet incorporated in the model. As a consequence, the predictions - that are performed in this research study - of the along-track density values of the Swarm constellation are not yet biased by the model itself. This is an additional benefit of working with the DTM2013. Finally, the third reason why the DTM2013 is preferred is due to its availability at the Faculty of Aerospace Engineering at the TU Delft.

The flowchart in Figure 2.1 presents the flow of data starting from the raw data files, their interaction with the DTM2013 and the format of the database in which the DTM2013's output is stored. The database is then used as a starting point for the machine learning models discussed in Chapter 3. This complete software interaction is written in the Python programming language. The decision to use this programming language is mainly driven by the availability of open-source machine learning libraries. More information on the specifics of the libraries which are used can be found in Section 3.1. As a consequence, a Python wrapper is written around the DTM2013, of which the source code is originally programmed in the FORTRAN programming language. All the different steps in this process are discussed next.

First, all data files are pulled from their corresponding external source. The different organisations providing these data files are listed in the next section. Next, the timestamps and the along-track density values are extracted from the Swarm satellite constellation database. These timestamps serve as the driver for two other input elements. The first one being the orbital state (longitude, latitude and altitude) of the Swarm satellite at this timestamp, which is taken from the Swarm database. The second element is the combination of solar and geomagnetic indices which are processed such that they comply with the input requirements of the DTM2013. If all data is extracted, the input variables are preprocessed such that one vector of values can be fed into the DTM2013 via the aforementioned Python wrapper. The internal algorithm of the DTM2013 has not been modified throughout this research project. A small discussion on this algorithm can be found in Appendix A, which is outside the scope of this report. Finally, the DTM2013 outputs the temperature, composition and density at a specified position and time. Also, the exospheric temperature above that position is calculated. This process is repeated for all timestamps of all Swarm satellites, such that an inventory can be stored. All values in this inventory are then be ready for the data preparation phase which will be discussed in detail in Chapter 4.

Figure 2.2 shows an example of the thermospheric density and temperature which are computed using the DTM2013 model for a fixed timestamp and varying altitude. All values below 120 km are indicated in black and are computed using the International Standard Atmosphere conventions, which will not be touched upon in this report but can be found in Appendix A. The maximum temperature, also called the exospheric temperature, is asymptotically reached in the topmost layer, the thermosphere. The level of

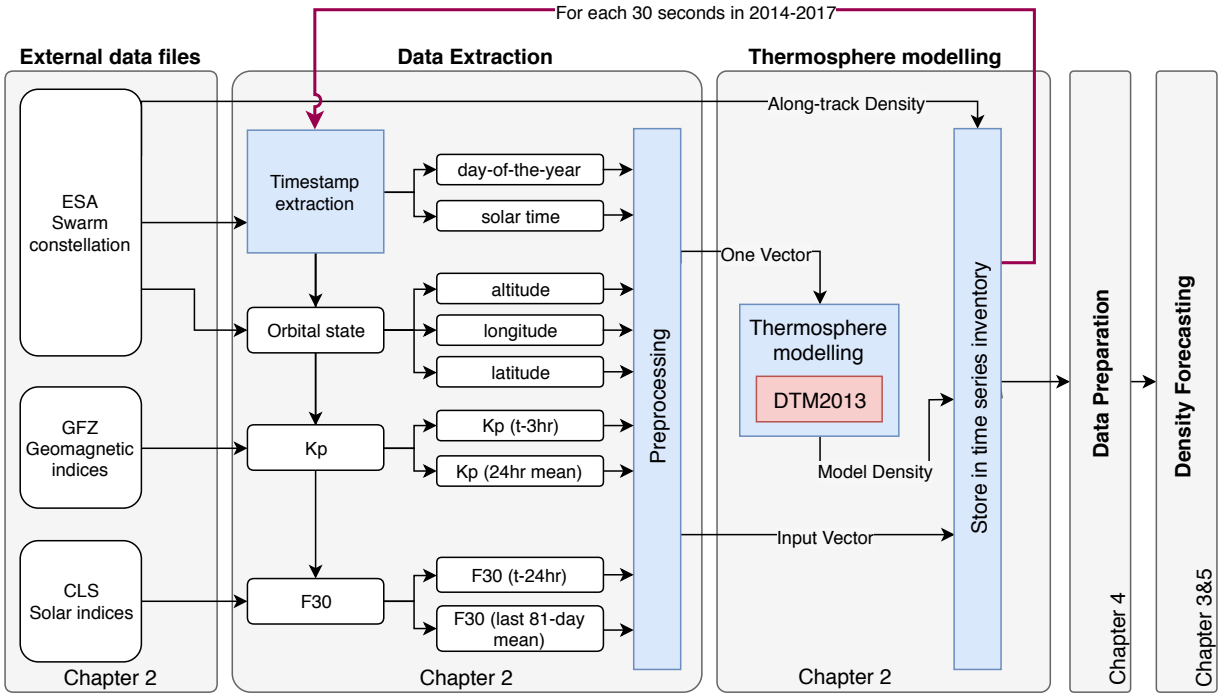


Figure 2.1: High-level code overview of the inventory set-up starting from external data files.

the temperature in this region is highly dependent on the solar activity at that point in time. The variations in temperature lead to variations in density, which can span several orders of magnitude [*Doornbos*, 2012].

2.2 Model Input Sources

In this section the different, required input data sets are listed and discussed. This selection is driven by two main components: the required input for the DTM2013 and the need for additional data sources in the machine learning model. Only the external data sources are discussed here. The preprocessing of the external data files will be left for discussion in Chapter 4.

2.2.1 Swarm Satellite Data

An introduction to the European Space Agency's (ESA) Swarm satellite constellation has been presented in Section 1.1.2. The constellation consists of three identical satellites, of which two of the satellites - Swarm-A and Swarm-C - fly at a lower orbital altitude than their third counterpart, Swarm-B. In Figure 2.3 their orbital altitudes, as well as the POD-derived density values, are shown in the time frame from January 2014 till December 2017. In this figure the influence of the solar activity on the thermospheric density becomes clear as well. If one would make a conclusion based on the DTM2013's output that is presented in Figure 2.2, one would expect the density to increase with decreasing orbital altitude. Although this altitude-density relation still holds for the same point in time, the variation in thermospheric density is primarily driven by the change in solar activity. Hence the lower experienced along-track density values at the end of this time frame are due to the lower solar activity, as presented in the *bottom* graph of Figure 2.4.

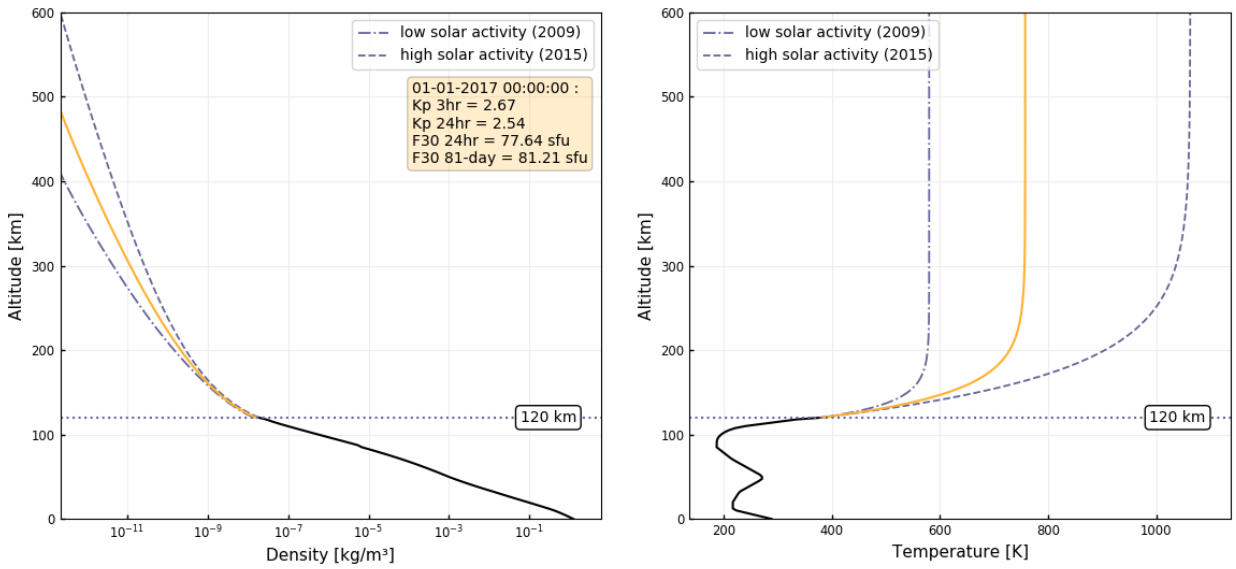


Figure 2.2: Altitude profiles of atmospheric temperature *(left)* and density *(right)*. Yellow lines present the DTM2013's modelled temperature and density profiles at 01-01-2017 00:00:00 at Delft, the Netherlands.

Table 2.2: Altitude and inclination ranges of the Swarm satellite constellation during the periods covered by the data used.

	Swarm-A	**Swarm-B**	**Swarm-C**
Time interval	01/01/2014-31/12/2017	01/01/2014-31/12/2017	01/01/2014-31/12/2017
Number of data points	4,082,424	3,931,846	4,128,001
Sampling frequency (s)	30	30	30
Altitude (km)	439-523	492-545	439-531
Colour DTM2013 data	Orange	Blue	Red
Colour POD data	Black	Black	Black

A summary of the properties of the complete data set of all three Swarm satellites is presented in Table 2.2. These data sets are initially build for each time step at which a POD-derived density value is available. All three satellites do not have the same number of data points which can also be seen in the multitude of empty timestamps in Figure 2.3 - primarily the ones earlier in the mission lifetime - at which no data is available. To get more insight into the mission envelope, the manoeuvres performed and mission anomalies there is taken advantage of the *Swarm Spacecraft Anomalies and Manoeuvres History (till 31 March 2018)* document and can be found on the Swarm platform [1]. Not all data can thus be used since its quality does affect the outcome of the forecasting models. Consequently, specific data sets are defined based on a set of properties which will be discussed in Chapter 4. The data generated by the Swarm satellite constellation serves multiple purposes of which the three main side notes are discussed next.

Firstly, while the DTM2013 dictates the required input variables for the model to work, the Swarm satellite constellation drives the position, altitude and timestamp at which these input variables need to be processed. In accordance with the ESA Earth Observation Data Policy, all Swarm data and thermosphere products are freely accessible to all users via anonymous access. The data files are downloaded via an ftp

[1] More info on: `https://earth.esa.int/web/guest/missions/esa-operational-eo-missions/swarm`

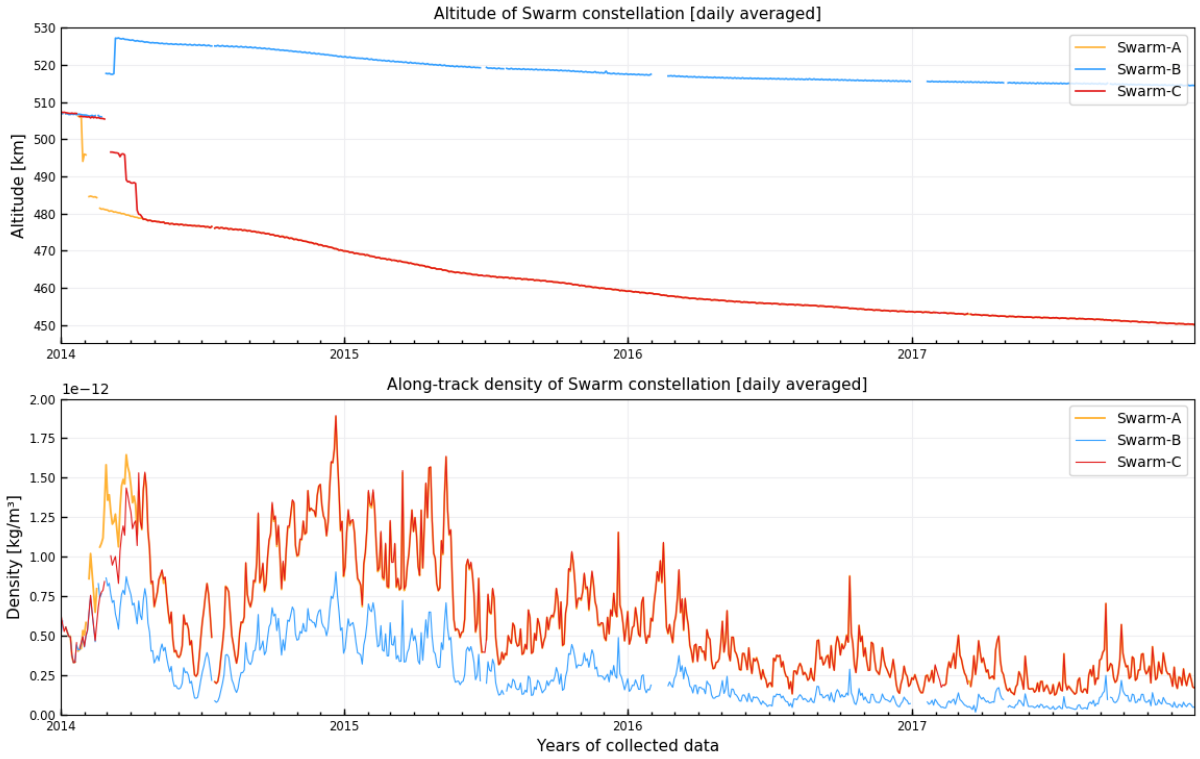


Figure 2.3: Orbital altitude of the Swarm satellite constellation (*top*), and POD-derived, along-track density measurements (*bottom*) over the timeframe of 2014-2017.

client [2]. For this research project, only the validated scientific data (Level 2) files are used, which means that the data of all three satellites has been combined with complex algorithms to generate the final thermospheric density products.

Secondly, only the time series of the neutral thermospheric densities that are derived from POD data are provided for the Swarm constellation. These values are provided for all three satellites at a sampling frequency of 30 s and are used as the benchmark density values for the DTM2013. ESA also offers the accelerometer-derived density values, but these are only available for a limited timeframe of the Swarm-C satellite. This is due to the fact that the Swarm satellites' accelerometer data are subjected to several disturbances [*Siemes et al.*, 2016]. From all three satellites, Swarm-C is the least affected by these disturbances [*Kodikara et al.*, 2018]. The longer the satellites are in orbit, the more non-gravitational accelerations will be experienced that may help improve the calibration of those accelerometer-derived densities. For this reason the focus will only be on the POD-derived density values. This also makes it a more robust approach since accelerometers are not carried by every satellites orbiting Earth.

Finally, an additional strength of using the Swarm constellation is that although it is considered to be one mission, the drifting orbits double the instantaneous local solar time coverage. The orbits of Swarm-A and Swarm-C are co-planar, which has the advantage to study relatively short spatial scale features. On the other hand, Swarm-B orbits the Earth at another altitude but still has an identical form factor as the two other satellites. This is advantageous when testing a proven or trained forecasting model on different orbital regimes. All three satellites orbit the Earth around 15 times a day.

[2] Anonymous access at `ftp://swarm-diss.eo.esa.int`

2.2.2 Solar Indices

The Sun emits electromagnetic radiation at different wavelengths. The radiation with the shortest wavelengths, called Extreme Ultra-Violet (EUV) radiation, reaching the thermosphere, is the main cause of heating up the thermosphere. The variability of emitted solar radiation is related to the level of activity of the Sun's magnetic field and the interaction of this magnetic field with gases on the solar surface and in its atmosphere [*Doornbos*, 2012]. The amount of EUV radiation absorbed by the thermosphere depends on two factors: the relative position and orientation of the Earth-Sun system, such as day- or night-side, seasonal variation, and the temporal variability of the solar activity. Direct measurement of EUV needs to be done in space. These space-based measurements are unfavourable due to the eroding effect of solar radiation on EUV measuring instruments, making instrument calibration difficult. Solar EUV flux is impossible to measure from ground due to the fact that most of the radiation will be absorbed by the atmosphere before it even reaches the ground instruments [*Zhang and Paxton*, 2018]. Nevertheless, ground solar radio instruments have the advantage to continuously observe specific radio wavelengths that can serve as a good proxy for EUV variability.

In this subsection, all solar indices that are implemented in this research study are discussed. The scaled F_{30} solar index is required as input in the DTM2013 and will be discussed first. This specific solar index is most commonly used to capture the solar activity [*Henney et al.*, 2012], thus generally very accessible. Other solar indices are discussed thereafter and are used as additional inputs in the machine learning algorithms. The selection of the best index is inconclusive and is case-dependent. It depends on the time scale of the forecast, the solar activity at that point in time and the thermosphere model in use.

Solar index, F_{30} scaled to $F_{10.7}$

The predecessor of the DTM2013, named DTM2009, uses the $F_{10.7}$ index as input. However, based on the findings by *Dudok De Wit and Bruinsma* [2017], the 30 cm solar radio flux (F_{30}) is used as solar proxy in the DTM2013. The paper concludes that the F_{30} proxy is more representative of the thermospheric solar energetic input in the ultraviolet than the $F_{10.7}$ for altitudes lower than 500 km, in which the Swarm satellites fly. After testing the different proxies on the DTM models, the F_{30} indices drove the reconstructed density values to higher fidelity. Since the internal algorithm is not changed in terms of input format, the F_{30} solar proxy needs to be scaled to $F_{10.7}$. This is done by implementing Equation 2.1 as provided by *Yaya et al.* [2017]:

$$\mathrm{F_{30scaled}} = 1.553755 \cdot \mathrm{F_{30}} - 1.5998 \tag{2.1}$$

The daily F_{30} indices are obtained from the servers of the Space Weather Services at the Collecte Localisation Satellites (CLS) [3]. This proxy is measured in Japan at the Toyokawa and Nobeyama observatories since 1957. The archived, absolute values, which are not corrected for solar flares and data gaps, are used for this research project. Absolute in the sense that the indices are not corrected for the eccentricity of the orbit of the Earth around the Sun since this better captures the effect of a solar radio flux at a specific point in time. The F_{30} value is provided on a daily basis and therefore kept constant throughout the whole day starting at 00:00:00 and ending on 23:59:50 of that same day. In the *top* graph of Figure 2.4 the $F_{10.7}$ and the scaled F_{30} index are shown for 2014-2017. Two insights can be generated. Firstly, the solar activity is much higher in the first half of the plotted time frame. This results in much higher experienced along-track density values due to the thermosphere puffing up. Secondly, the variability in solar activity is less in the second half of the time-frame.

[3] Anonymous access at `ftp://ftpsedr.cls.fr/pub/previsol/solarflux/forecast/absolute/`

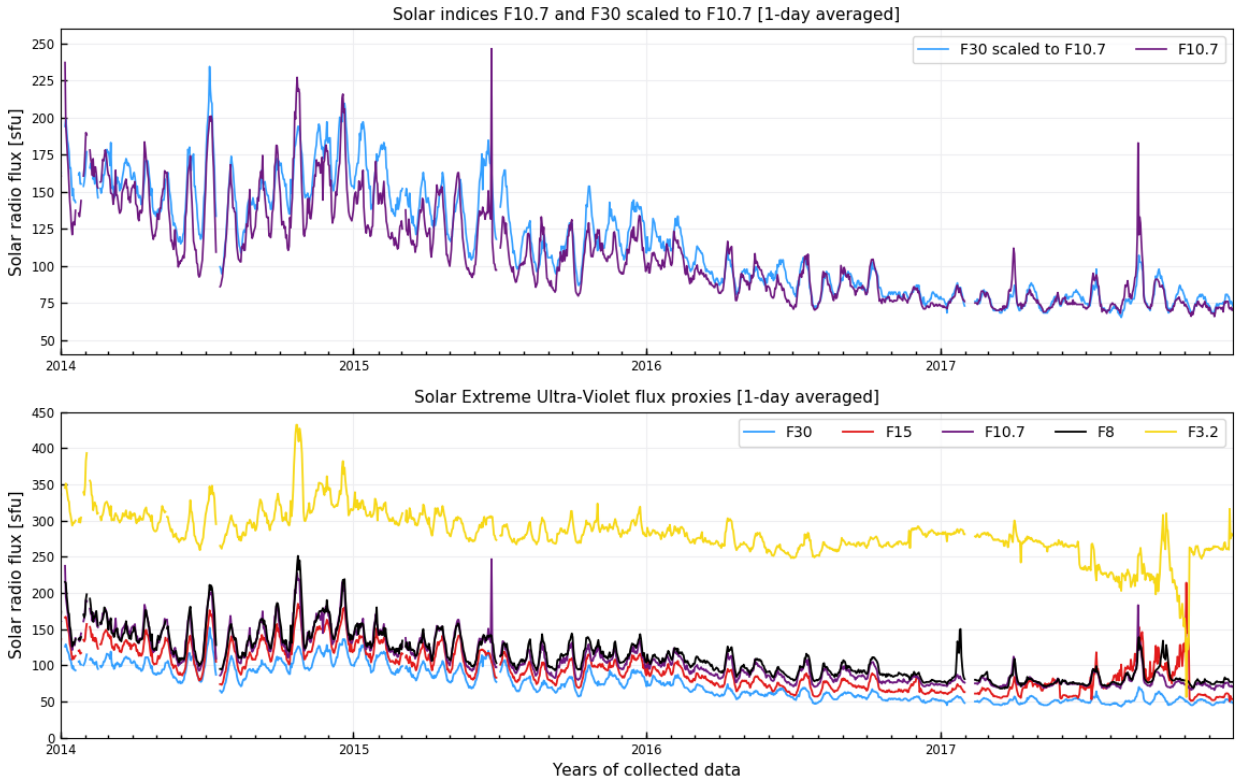


Figure 2.4: 1-day averaged scaled F_{30} and $F_{10.7}$ (*top*), and collection of solar indices at different wavelengths (*bottom*) over the timeframe of 2014-2017.

Other solar indices

In terms of solar indices, there is another approach to approximate the EUV flux by combining different proxies in one model. *Bowman et al.* [2008] saw an opportunity in the unmodelled error sources that show a periodicity of around one solar cycle. They therefore opted for a combination of daily proxies, namely S_{10}, M_{10} and Y_{10}. *Zhang and Paxton* [2018] also try to create a solar EUV flux proxy by combining solar radio fluxes at six different frequencies using neural networks. This study indicates that a combination of solar fluxes may improve predictions of solar EUV flux. The latter study is the most feasible in this context due to the availability of these solar indices and the ease with which they can be extracted from the external files. Additionally to the F_{30} solar radio flux, the F_{15}, $F_{10.7}$, F_8 and $F_{3.2}$ fluxes are also extracted from the same external data files obtained from the server of the Space Weather Services at the Collecte Localisation Satellites (CLS). Their variation over the timeframe of 2014-2017 is presented in the *bottom* graph of Figure 2.4. In general they do follow the same trend as the scaled F_{30} indices, indicating their common correlation with the solar activity. The extent to which these EUV proxies are correlated to the variability in density measurements will be discussed in Section 4.2. Sensors which are able to monitor the EUV flux do exist, but are excluded from this study. This is due to the fact that it is hard to obtain a publicly available database with those EUV measurements for extended periods of time at the sampling rate of the POD-derived density values of the Swarm satellite constellation.

2.2.3 Geomagnetic Activity

The Sun constantly emits a stream of particles out into space, called the solar wind. The solar wind consists of charged particles which shape the magnetic field around the Sun. This solar wind then interacts with the Earth's geomagnetic field which creates the Earth's magnetosphere. These terrestrial magnetic field lines shield the Earth from the charged particles emitted by the Sun. A complex system originates from this interaction since the moving charged particles induce currents and electric fields that will interact with the Earth's magnetic field [*Doornbos*, 2012]. When the solar flares and coronal mass ejections are of extraordinary magnitude, so-called geomagnetic storms perturb the Earth's magnetic field, which can last several hours or even days [*Echer et al.*, 2005]. In conclusion, the two most important geomagnetic processes that affect the thermosphere are the interaction of the energetic particles from the magnetosphere with the thermosphere and the energy deposited from the ionosphere electrical currents via Joule heating [*Wilson et al.*, 2006]. *Donner et al.* [2018] state that the magnetosphere is continuously far from equilibrium and undergoes complex variations at a broad range of temporal and spatial scales. This results in a response of the magnetosphere which commonly changes abruptly rather than gradually. Thus, expectations are that predicting geomagnetic storms will be a challenge, despite the extra geomagnetic indices that are included in the DTM2013 and machine learning models.

In this subsection, all the geomagnetic indices that are implemented in this research study are discussed. The K_p index is a required input for the DTM2013 and will be discussed first. This specific geomagnetic index is most commonly used to capture the geomagnetic activity [*Henney et al.*, 2012]. Other geomagnetic indices are discussed thereafter and are used as additional inputs in the machine learning algorithms.

Geomagnetic index, K_p

Due to the influence of the solar and ionospheric activity, amongst others, trying to find one parameter which captures the complete geomagnetic activity is almost impossible. Just like $F_{10.7}$ is used to represent the solar EUV energy input, geomagnetic indices come into use to characterise the geomagnetic activity. The K_p index is an often used proxy to represent thermospheric energy input due to geomagnetic activity. K_p is derived and standardised from measurements made at 11 geomagnetic observatories on a daily basis [*Doornbos*, 2012]. K_p is measured on a 3-hourly interval by the German national research centre for Earth sciences named *The GFZ German Research Centre for Geosciences*. These historical 3-hourly values are anonymously accessible on their server platform [4]. A closely related index, a_p, is the planetary equivalent amplitude. This value is extracted from the server as a raw value and then conversed in the corresponding numeric value of K_p using the conversion table in Table B.1 of Appendix B. This conversion is required, since K_p needs to be used as input for the DTM2013. K_p indices range in 28 steps from 0 (quiet storm) to 9 (greatly disturbed storm) with fractional parts expressed in thirds of a unit.

K indices isolate solar particle effects on the earth's magnetic field; over a 3-hour period, they classify into disturbance levels the range of variation of the more unsettled horizontal field component [*Johnson and Wing*, 2005]. Each activity level relates almost logarithmically to its corresponding disturbance amplitude. The daily average of K_p for 2014-2017 is shown in Figure 2.5. However, the K_p indices are stored as such that they are kept constant over the whole 3-hour interval.

[4] Anonymous access at `ftp://ftp.gfz-potsdam.de/pub/home/obs/kp-ap/`

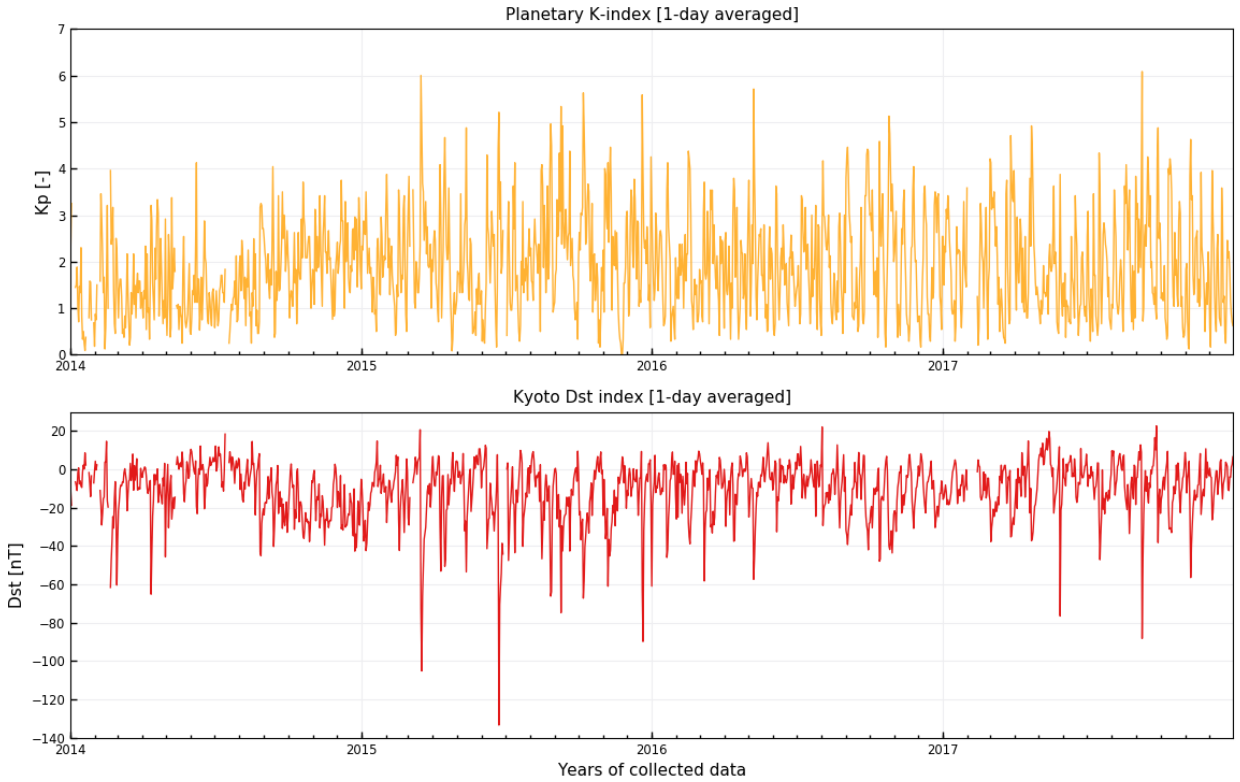


Figure 2.5: 7-day averaged planetary K-index obtained from the GZF German Research Centre for Geosciences over the timeframe of 2014-2017.

Kyoto D_{st} index

Other geomagnetic indices are not required as input for the DTM2013, but can possibly feed relevant information into the machine learning networks. For this reason, a selection of additional indices is discussed in this paragraph. The external data files, data extraction and inventory set-up to add these indices are similar to the flowchart presented in Figure 2.1.

In terms of geomagnetic indices, *Bowman et al.* [2008] implemented the D_{st} index instead of K_p to incorporate the effect of the geomagnetic storms phenomena on the thermosphere. D_{st} provides more detail, because of its hourly rate, but it only represents the geomagnetic disturbances in the equatorial region. This study also concluded that when D_{st} was introduced the overall model density errors were reduced especially during geomagnetic storms. *Lazzús et al.* [2017] developed new forecasting methods to predict the future values of this geomagnetic index. For this reason, D_{st} is added as an additional geomagnetic index. Its values are obtained from the *World Data Center for Geomagnetism* located in Kyoto and operated by the Data Analysis Center for Geomagnetism and Space Magnetism of Kyoto University. They do not provide server access but are accessed by their Digital Data Archive [5]. The variation of D_{st} over the full timeframe is shown in the *bottom* part of Figure 2.5. Similar to the solar indices, its correlation to the POD-derived density values will be discussed in Section 3.4.3.

[5] Obtained from `http://wdc.kugi.kyoto-u.ac.jp/dstae/index.html`

2.3 Model Verification and Validation

The output of the DTM2013 will form the backbone of the rest of the research discussed in this report. It is therefore relevant to verify and validate its workings given the software integration discussed in Figure 2.1 and the multitude of different external data files that are used. Verification is to check if the software integration and the models work as expected. Validation is to make sure the model is actually calculating the right thing, in this case the thermospheric density at a specific time and position.

The verification of the output is done for the Swarm-A satellite over the complete day of 1 August 2014. The verification data was provided by my supervisor Dr. Ir. Eelco Doornbos from the Faculty of Aerospace Engineering and is assumed to be the correct DTM2013 input and output for that day. The Swarm POD-derived density values are used as validation data for the model. Furthermore, the European Space Agency offers a quality report of the Swarm satellite constellation for each day. Additionally, this quality report is used both to verify and validate the results. However the model (NRLMSISE-00) used in this report is different, it is still used as an indicator for the model's output. Although the internal algorithm of the DTM2013 is not changed, the model is tested if, after having implemented the Python wrapper, still outputs the correct values for a specific input vector. This is done for one day of the verification data set provided by my supervisor and can be considered verified. The verification of the model as part of the software integration is split up in three part: the verification of the solar indices, magnetic indices and the model density values. This model density also encaptures the influence of the time and location inputs of the model, which are not checked separately. Figure 2.6 shows three plots that correspond to all three verification parts.

The *top* graph shows the input values for the geomagnetic index, K_p, of that day. The geomagnetic indices are made available every three hours and are kept constant in this report. The verification dataset linearly interpolates the 3-hourly values. Both methods are correct and depend on the interpretation of the user. The values are similar when looking at the start and end points of each interval, and are thus considered verified. The mean of the last 24-hour values of K_p is similar in both the model input and the verification dataset. The *middle* graph shows the input values for the solar index, F_{30} scaled to $F_{10.7}$, which is made available on a daily basis. Although the instantaneous flux appears similar in the plot, there is a small difference due to the interpolation scheme similar to the geomagnetic index. This reflects in the small offset of less than 1% of the model's input values when taking the 81-day average of the solar indices. This input is verified given the small difference in input values. The *bottom* graph finally plots both the POD-derived density values of Swarm-A, as well as the DTM2013's output for both the model set-up discussed in this chapter as the density values of the verification data set. The modelled densities do correspond largely and only deviate in absolute values when the density values reach a peak or low. This small offset is explained by the small differences in input values as explained before. When comparing the modelled density values with the along-track density values, it is concluded that the values are in phase but do differ in magnitude. This can be explained by a multitude of reasons ranging from the limitation in the use of solar and geomagnetic proxies to the assumptions encapsulated in the DTM2013. A more elaborate discussion on these differences will be given in Chapter 5. When taking a look at the report of the ESA of 1 August 2014 in Figure 2.7 one can draw the same conclusions as discussed earlier.

In conclusion, the DTM2013 and the software infrastructure can be considered verified and validated for future use. Hence, all input and output values stored in the database can be used as input for the machine learning models. To verify if the correct indices were stored, a set of random test dates were chosen on which the raw data files and the inventory output were compared. All values extracted from the final database were the exact same values as presented in the raw data files and are thus considered verified.

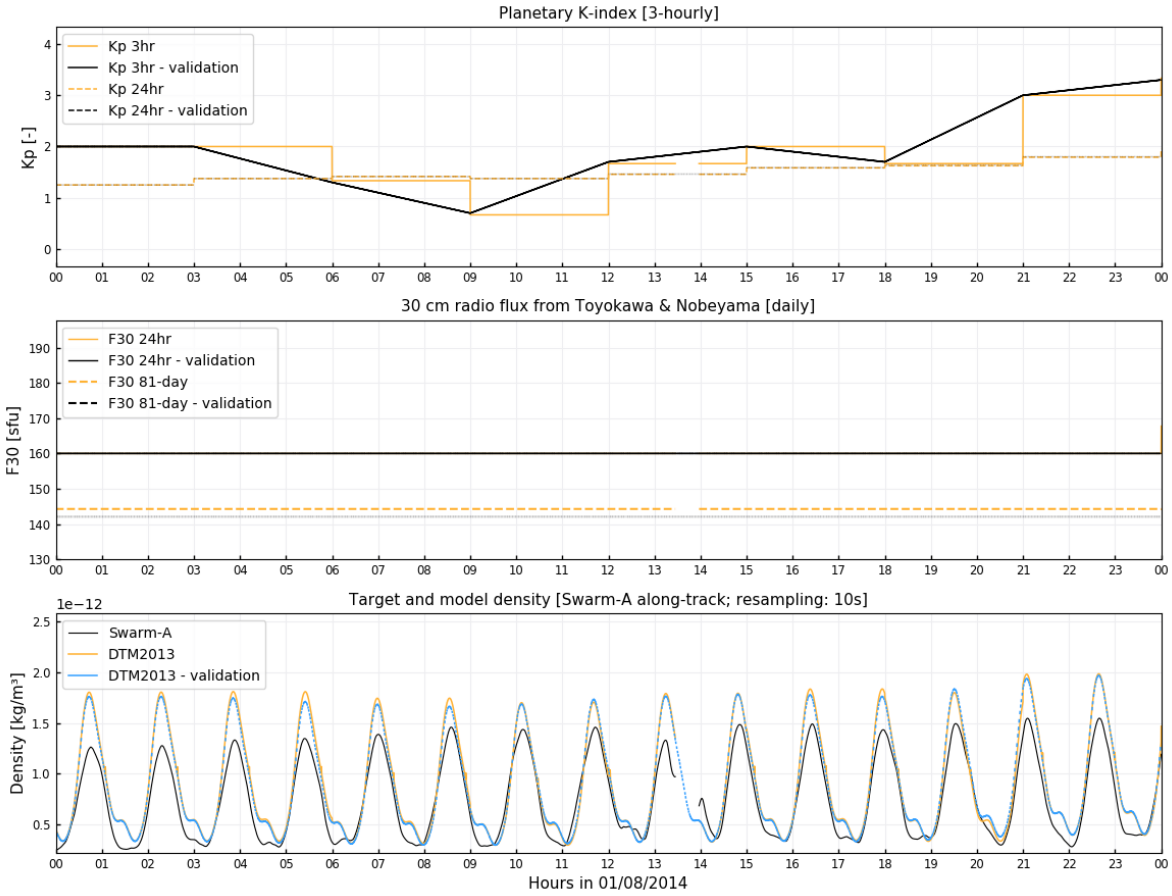


Figure 2.6: Input geomagnetic indices (*top*) and solar proxies (*middle*). Swarm-A along-track density values for 1 August 2014 using the DTM2013 empirical model (*bottom*). For verification and validation purposes.

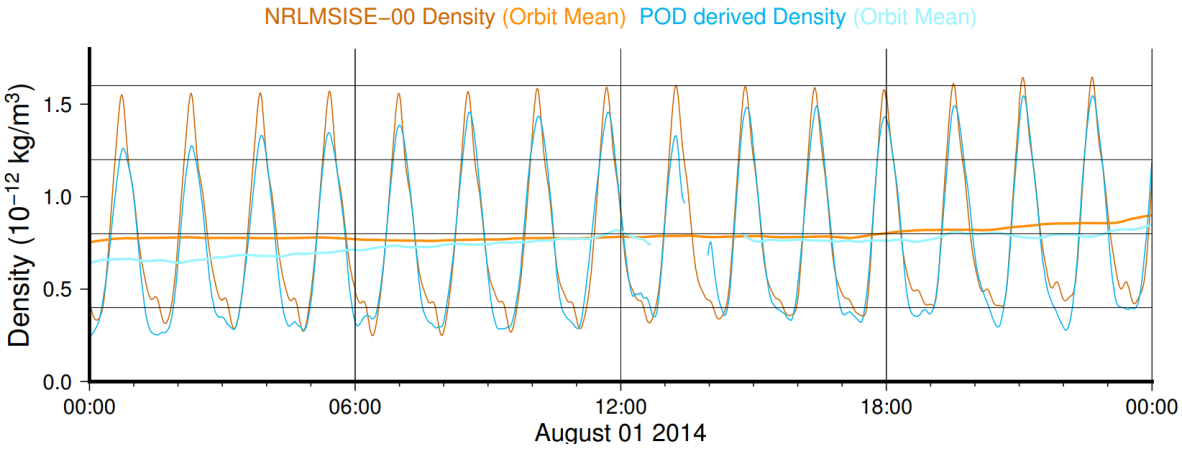


Figure 2.7: ESA validation file for Swarm-A along-track POD-derived density values for 1 August 2014 using the NRLMSISE-00 empirical model.

Chapter 3

Forecasting Methods

This chapter elaborates on both the methodology that is used throughout this thesis project, as well as the knowledge required to study the problem at hand. In Section 3.1 the architecture of the implemented forecasting techniques is explained. Section 3.2 highlights the time-dependency when forecasting time series. Next, Section 3.3 lists the baseline models which are used to compare the performance of the Neural Network (NN) models with. These latter ones are discussed in Section 3.4 and form the backbone of the analysis. The performance of those models is compared by using performance metrics. These metrics are depicted in Section 3.5.

3.1 Improvement Technique

This section lays out the methodology used to enhance the established thermosphere model's output by incorporating an additional NN model. Machine Learning (ML) is first and foremost known for its classification and regression predictive modelling capabilities within the computer vision space. The temporal structure inherently linked to a time series adds an additional order to the observations itself. As a consequence, a different methodology needs to be applied, which will be discussed in the coming two subsections. Before diving into the methodology itself, some specific naming conventions are defined in Table 3.1.

Table 3.1: Naming conventions used to define the different density values and models.

Naming Convention	**Symbol**	**Meaning**
Established model	-	DTM2013
Baseline models	-	DTM2013 or Calibrated Persistence Model for forecasting
Improved model	-	Neural Network model
Model density	$\hat{\rho}_{DTM2013}$, $\hat{\rho}_{DTM}$	DTM2013 output density
Target density	ρ	Swarm along-track POD-derived density
Predicted density	$\hat{\rho}$	NN model output density
Prediction window	W_p	Prediction window of the NN model
Cost function	J	Internal NN model performance metric
Performance metrics	RMSE, R, Bias...	NN model output accuracy performance metrics (see Section 3.5)

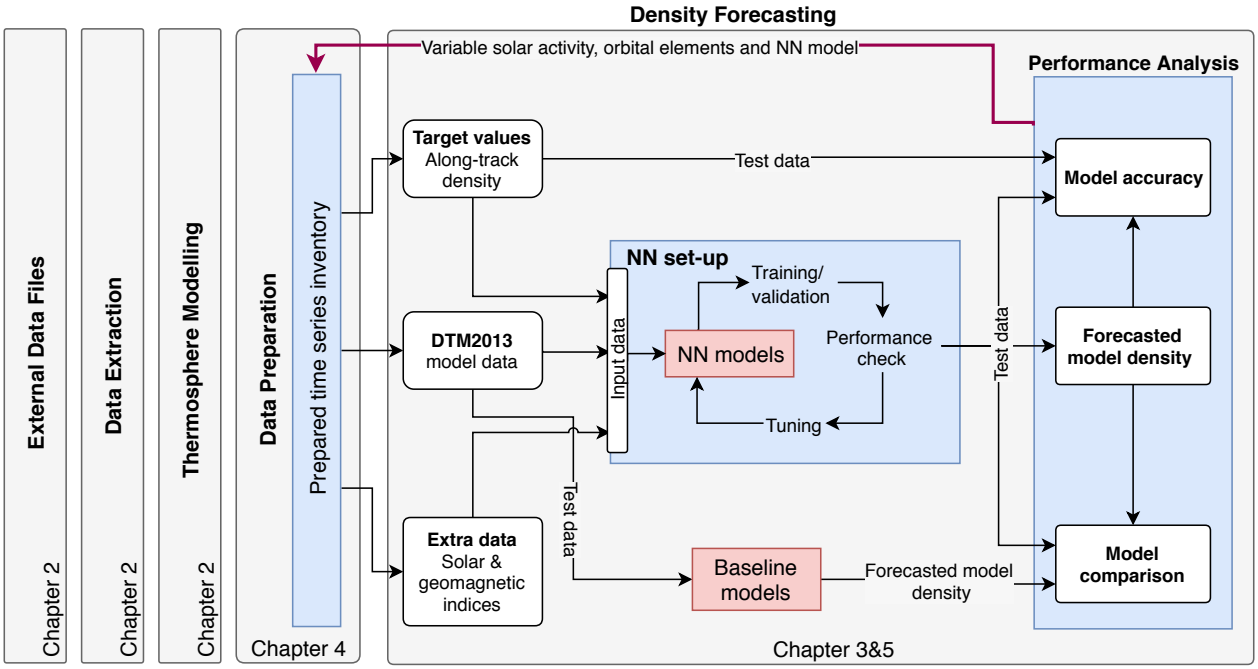


Figure 3.1: High-level code overview of the integrated forecasting models.

3.1.1 Methodology

In this subsection the software integration methodology is discussed. Figure 3.1 presents a flow chart of the data interacting with the NN model to improve the time series forecasting capabilities of the DTM2013. The goal is twofold; firstly, to develop a model which uses only past DTM2013 and Swarm satellite density values, as well as solar and geomagnetic indices to predict model densities in the near future, and secondly, to create a model which minimises the offset between empirical model and POD-derived density values. The NN model development starts at the point where the data preparation ends. This latter phase and the exact operations that fall under it are discussed in the next chapter. Only past model and target data can be used and both data sets are split into different training and test data sets. In conclusion, the improved forecasting model set-up can be divided into two segments, namely training (eg. developing) the NN model and testing the NN model. Both segments are discussed next.

First, the training phase is elaborated upon. This phase is indicated by the blue box in Figure 3.1 stating *NN set-up*. Training data sets, and not test data sets, are used to train/develop both the baseline and NN models. Only DTM2013 density values are used as input to train the different models. However, NN models have the capability to incorporate additional data sources, such as solar and geomagnetic indices. During the training phase of a model, the internal parameters of the model are tweaked until the model's output has reached the desired accuracy. This accuracy gives an idea of how well the model is able to transform its input such that it approximates the target values. The target values in this research study are the POD-derived along-track density values and they are not used as input for the baseline or NN models. Afterwards, these internal parameters are saved and used to test the model's performance on the test data sets.

The reason for which the POD-derived density values cannot be used as the input values for the NN model is that these density values are not publicly available on a daily basis with high accuracy. The goal

of this research is to provide a tool which can be put in practise from day one. Therefore, only solar and geomagnetic indices, and thus also DTM2013 density values can be used as input values since they can be obtained on a daily basis.

Secondly, the testing phase is discussed. After the model is trained, it is used to forecast the thermospheric density values on a specific test day. These predicted density values are then compared to two different values using the performance metrics that will be discussed in Section 3.5. The first one being the predicted density values generated by the baseline models. The second one being the target values on that specific test day to check two aspects. Firstly, how well the model can learn the relationship between the DTM2013 input values and Swarm density values. Secondly, how well the network is able to forecast along-track density values given its input.

One could argue that predicted model densities shall be compared to a third parameter, namely the predicted DTM2013 density values. If one wants to forecast future density values using an empirical model such as DTM2013, the input variables (solar EUV proxies and geomagnetic indices) need to be forecasted first. The better the quality of such a forecast, the more accurate the final density estimate will be. Sadly enough, the forecasted solar and geomagnetic indices at a point in the past are not meticulously stored by the different observatories. Hence a hypothetical density forecast cannot easily be constructed. The reader is encouraged to take a look at three input forecasting methods which are discussed in literature. A linear input forecasting model by *Warren et al.* [2017], an autoregression (AR) and neural networks (NN) model by *Yaya et al.* [2017]. It is safe to assume that the DTM2013's density values with predicted input variables will always be inferior to the true DTM2013 density values computed using up-to-date input values. For this reason, the predicted model densities of the NN models are not compared to these DTM2013 density values with predicted input variables within the scope of this research.

To answer all research questions, the steps presented in Figure 3.1 shall be repeated using different NN models, as well as different combinations of training and test datasets to assess the models' forecasting abilities in terms of variability in solar activity and orbital elements.

3.1.2 Software Description

The software tools used to set-up a variety of machine learning models are discussed now. The preferred programming language and software modules do affect the way the models are constructed and tested. Throughout the whole thesis project the Python (version 3.6) programming language is used. This decision is based on personal evaluation of the different available options (eg. MATLAB, FORTRAN and C++). The main drivers for this decision are my personal experience with Python and the available libraries for database operations and machine learning. The biggest benefit of working with this open-source programming language is that a large community and ample documentation are within easy reach online. The next four libraries are key in developing time series forecasting models and frameworks and are considered validated libraries used at a multitude of Fortune 500 companies:

1. *Pandas*: software library written for the Python programming language for data manipulation and analysis.
2. *Statsmodels*: a Python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration.
3. *TensorFlow*: free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.
4. *Keras*: open-source neural-network library written in Python.

3.2 Autocorrelation

In this section the definition and relevance of autocorrelation are discussed. Autocorrelation is a technique which is often used in time series forecasting methods such as Autoregressive (AR) and Autoregressive Integrated Moving Averages (ARIMA) models. These models were considered as feasible forecasting models for the drag problem but were then neglected due to their limited learning capabilities. The aforementioned models use the autocorrelation function (acf) to define their internal parameters. As an example, the autocorrelation function is used in the model identification stage of the Box-Jenkins ARIMA time series model [*Enders*, 2014]. The acf is obtained from computing the linear correlation of each value at time t to the other values at time *t-1*, *t-2* and so on [*Flores et al.*, 2012]. This function thus shows the intensity of the temporal autocorrelation at a different number of previous time steps. These time steps are called lag variables.

Some studies, such as the one by *Guoqiang et al.* [1998], do not use autocorrelation functions in combination with NN models since these functions are based on linear relations. Interpreting the acf seems therefore useless since NN models try to capture the non-linearities within a time series. However, the acf is still helpful for a multitude of reasons. Firstly, forecasting the next time steps of a time series do best work when the series is considered stationary (see Section 4.2), meaning there is no presence of any trend and/or seasonality. The autocorrelation plot is a means to discover seasonality in a time series. If seasonality is present, it affects the time series forecasting model. It has an advantage over, for example, using box plots to discover seasonality if the period of seasonality is not known. It also helps to set an expectation of the likelihood that a certain prediction window is still going to provide acceptable results. Secondly, the acf is useful when constructing the baseline and NN models. It helps in figuring out the number of input data points that can be included in the NN model's input. Thirdly, other research papers do promote using the acf to improve neural network models. As an example, *Flores et al.* [2012] argue that the acf is beneficial when used to minimise the number of parameters in the input layer which need to be learned.

The correlation coefficients of the acf could be manually plotted for each lag variable. Luckily, the open-source Pandas library has a built-in function to plot these coefficients automatically. This is done for a month worth of DTM2013 density data and is shown in Figure 3.2. Each lag variable in this plot represents a shift of one day. In this figure the autocorrelation for 30 of such shifts is presented. The lag number is presented along the x-axis and the correlation coefficient value on the y-axis, ranging from -1 to 1 on the y-axis. The plot also includes solid and dashed confidence bands which indicate how relevant the correlation at that specific lag number is. If the autocorrelation curve falls in between these confidence bands, the correlation is considered not to be present. The pattern of this autocorrelation function is similar to a damped wave whose amplitude of oscillation decreases almost exponentially with time and eventually is going to zero. This sinusoidal behaviour indicates that there is some periodicity present in the modelled density data. However, this periodicity is not strong enough for the time series to become non-stationary, as will be discussed in Section 4.2. Another insight is that the along-track density values on one day are most significantly correlated with density values that are up until three and a half days apart. This indicates that the most useful information seems to lie in this time interval. This information will finally help us in defining the maximum amount of input variables of past days that the neural network can have in its input vector. This analysis will be touched upon in Chapter 5.

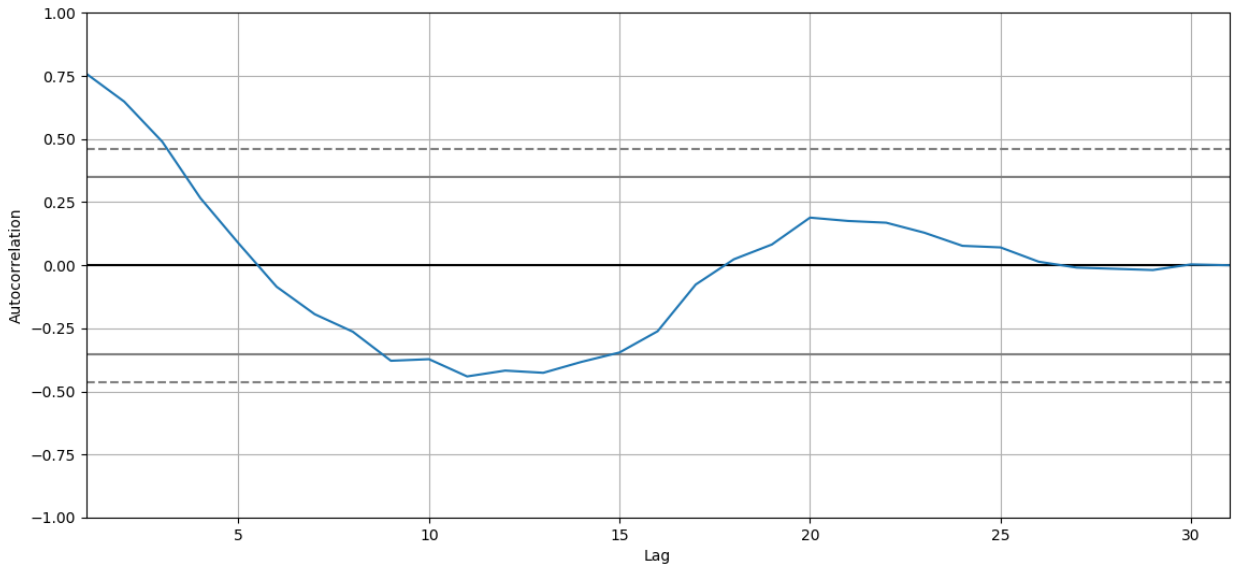


Figure 3.2: Autocorrelation for DTM2013 modelled, along-track density values for the Swarm-A satellite during one month (11/2014). One lag variable represents a single shift of one day.

3.3 Baseline Models

The baseline forecasting models are discussed in this section. A baseline in forecast performance provides a point of comparison for all other forecasting models described in this chapter. Two different baseline models are discussed in this section. The simplest model, which is the normal persistence model, and afterwards to calibrated persistence model. If any of the machine learning models achieve a similar of lesser prediction performance, the model should be tuned or replaced. Both models are univariate forecasting models, meaning they only incorporate a time series of the past DTM2013 density values.

3.3.1 Persistence Model

One of the simplest methods for predicting the future behaviour of a time series is the so-called persistence model [*Coimbra and Pedro*, 2013]. Persistence implies that future values of the time series are computed on the assumption that conditions remain unchanged between "current" timestamp t and future timestamp $t + W_p$, where W_p stands for the prediction window. Equation 3.1 shows the implementation of the persistence model in the thermospheric density forecasting case. Thus, a persistence model forecasting the density will predict the future density values to be the same as it in the present [*Pérez et al.*, 2014].

$$\hat{\rho}(t) = \rho_{DTM}(t - W_p) \tag{3.1}$$

The only requirement for the implementation of the persistence model as defined in the aforementioned equation is the stationarity of the time series. This means that only a time series can be used whose mean and variance do not change over time. The test to determine if a time series is stationary will be discussed in Section 4.2. In short, the density variations phenomena are clearly non-stationary because of the diurnal, seasonal and yearly cycles. Therefore this model is only limited to make predictions where the prediction window lies around the orbital period of the Swarm satellites. This means that if each predicted orbit has similar along-track density values as the ones measured in the preceding orbit, given

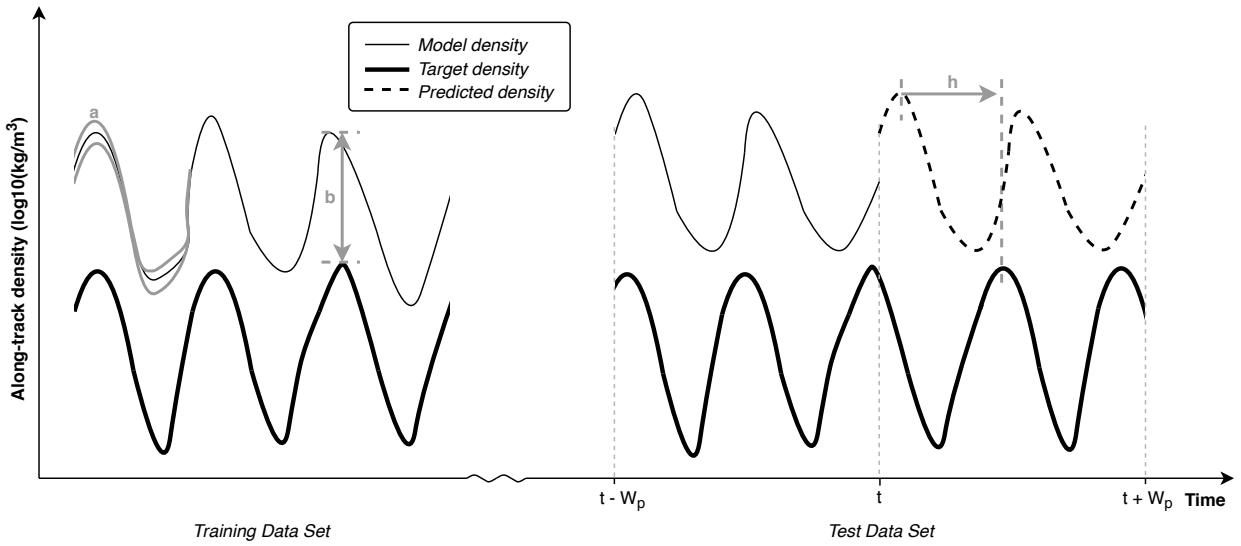


Figure 3.3: Schematic representation of the calibrated persistence model with the determination of scale factor and bias on the training data set and the horizontal displacement factor on the test data set. (W_p = Prediction Window)

a prediction window of one orbit. This conclusion is also reached when discussing autocorrelation in Section 3.2.

3.3.2 Calibrated Persistence Model

As concluded in the previous subsection, the persistence model has some setbacks if the prediction window is poorly defined. The along-track density values measured by a Swarm satellite fluctuate due to it moving through the day- and night-side of the Earth. The forecasted time series is out of phase with the true POD-derived density values if the prediction window is a value other than a multiple of the orbital period. Besides being out of phase, a model's forecast can also be subjected to errors caused by an offset in its amplitude or a vertical offset compared to the POD-derived density values. Figure 3.3 presents a schematic of how the normal persistence model scheme can be transformed into a forecasting method with superior performance.

Scaling the persistence model is therefore an effective and straightforward approach with the capability to outperform more advanced forecasting methods. Three operations are applied to the original persistence model to complete the transformation. As a consequence, Equation 3.1 is rewritten to Equation 3.2 such that all three operations are accommodated.

$$\hat{\rho}(t) = a \cdot \rho_{DTM}(t - W_p - h) + b \tag{3.2}$$

All operations are discussed below. First, a horizontal displacement needs to be performed such that the predicted density values are in phase with the true density values. This displacement is indicated by the variable h. Afterwards, the DTM2013 density values are vertically displaced by a bias b. Finally, the model is scaled by a scaling factor a. It is important to mention that all three variables can only be achieved by using past data.

Horizontal Displacement (h)

The horizontal displacement is required to put the forecasted DTM2013 values into phase with the target density values. Consequently, the point, at which the current DTM2013 density values are connected with the forecasted density values, becomes more continuous. The horizontal displacement is a time unit and is obtained by shifting the prediction window backwards in time until the Root Mean Squared Error (RMSE) value between the horizontally shifted persistence model and the POD-derived density values has reached a first local minimum. For this operation it is assumed that the POD-derived density values become available on a daily basis.

Scale Factor (a) and Bias (b)

Contrarily to the horizontal displacement which is computed by using the test data set, the scale factor and the bias are calculated based on the information available in the training data set. Both the model density time series and target density time series are used to compute the scale factor and bias. The method of least squares is used to determine which combination of bias and scale factor results in the lowest RMSE value between the calibrated persistence model and the POD-derived density values. Since the horizontal displacement is not applied on these training data sets, it is expected that the model and target density time series are in phase with each other.

3.4 Neural Network Models

Machine Learning (ML) is a subset of Artificial Intelligence (AI), that on its own, contains different subsets of more detailed methods [*Domingos*, 2012]. The main goal of ML is to create predictive models and learn 'new rules' from large data sets without explicitly being programmed to do so. This implicit learning makes NNs an attractive approach for the drag problem. In literature many of the physical reasoning for the variations and trends of the thermospheric density are already identified, but improvements are still possible. Conventional approaches restrict themselves in solving problems that are largely understood and which are known how to solve. Nonetheless, the sun-thermosphere interaction is characterised by delays in heating, unpredictability in solar events and other driving forces. The development of new NNs methods could unlock further improvements in understanding this complex system.

One could argue that calibrated persistence models can achieve decent forecasting performance with minimum effort. Nevertheless, other benefits do make it worthwhile to verify the capabilities of neural network models in forecasting along-track density values based on past model densities. Firstly, the solar-terrestrial system is complex and nonlinear in nature. The related ability to forecast the effect of this system on the Space Weather (SW) around Earth thus encourages the development of more advanced tools than the forecasting models that are built on linear relationships. Secondly, methods such as persistence and AR/ARIMA models can only incorporate a single input type. Contrarily, NN models have the ability to gather extra information encapsulated in additional data sources. Finally, NN models do have the power to train their networks to learn the relation between two inherently different time series, eg. model and POD-derived density data.

The studies by *Pérez et al.* [2014] and *Pérez and Bevilacqua* [2015] both use NN models to improve the density output of specific thermosphere models. Density forecast performance depends on the amount of training data and the correlation that the input time series have with the variable that needs to be estimated. Both studies concluded that using NNs is a promising strategy to improve density forecasts.

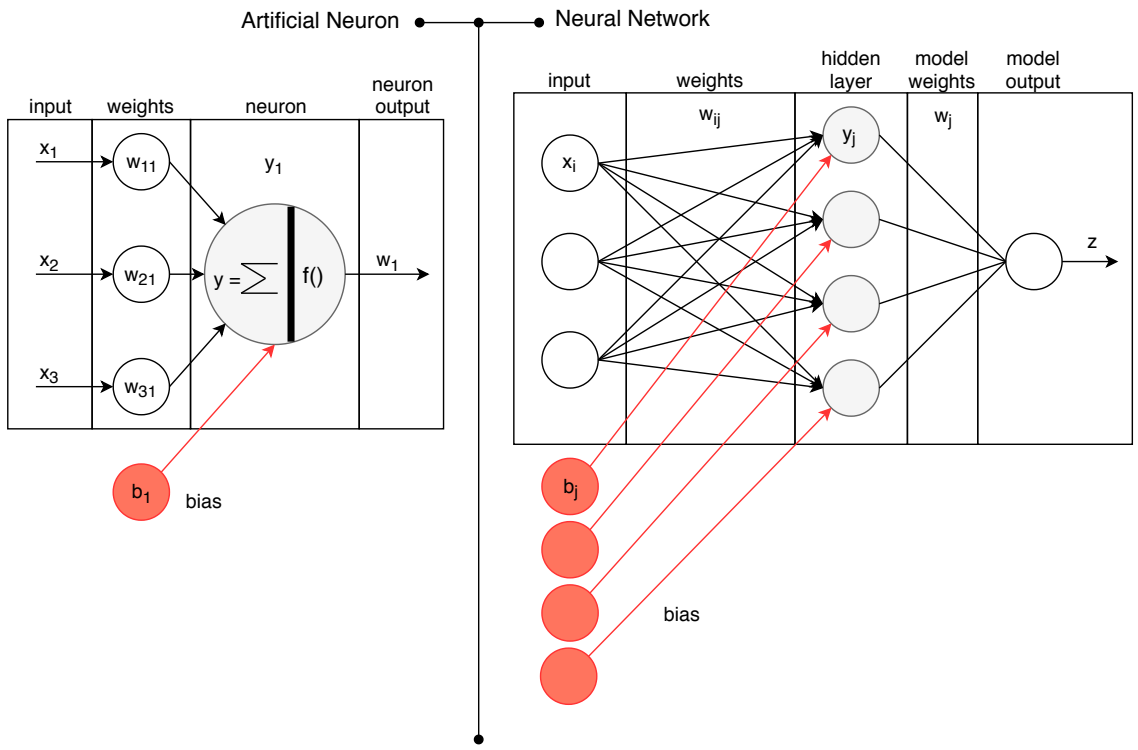


Figure 3.4: Artificial neuron diagram (*left*) and a possible Neural Network model with one hidden layer (*right*).

The main focus of this section is to provide the user with a more technical background in the working principles of NN models and the specific models that are designed for the research questions under investigation. Subsection 3.4.1 depicts the general theory of NN models. Next, Subsection 3.4.2 defines the forecasting approach that is undertaken. Subsection 3.4.3 lays out the specific Multilayer Perceptron (MLP) models that are created. The nested cross-validation approach is implemented to create those MLP models. The exact theory behind this approach is explained in Subsection 3.4.4.

3.4.1 Theory

The regular neural network models are an example of classifiers that will link specific behaviour in the input vector with a certain tendency of the output variables [*Haykin*, 2009]. Generally, NNs try to represent the inner workings of the biological brain in a mathematical manner. The brain consists of a large network of neurons, which form the key building blocks for NNs as well. They will be the units that capture the nonlinearities in a dataset and are a combination of inputs, outputs, weights, biases and nonlinear transfer functions or activation functions. The activation functions will give the network the ability to map nonlinearity in its data [*Pérez et al.*, 2014]. Figure 3.4 visualises the working of one neuron (*left*) and the connection of neurons in the neural network (*right*).

Between the input and output layer of the NN model, one or more layer(s) of neurons are inserted. The number of neurons in one layer will determine the complexity the neural network is able to learn, with the risk of having too few neurons (not being able to learn the full complexity hidden in the data set, *underfitting*) or too much neurons (the model has lost the ability to generalise among the training data set and thus underpeforms when new inputs are fed into the system, *overfitting*). The mathematical expression of the information that can be stored in a neuron is given by Equation 3.3:

$$y_j(t) = \psi\left(\sum_i w_{ij} x_i(t) + b_j\right) \tag{3.3}$$

This input-output (x_i-y_i) relation is defined by the type of activation function ψ, the biases b_j and weights w_{ij}. The network is getting 'smarter' by re-evaluating the cost function J (in this case the Mean Squared Error (MSE)) as explained at the beginning of this section. This cost function is a measure on how well the input-output relation of the training data set is being represented by the inner working of the NN. By using backpropagation, the weights and biases are adapted with the intent to increase overall performance and thus to minimise the overall cost function. As a remark, the cost function is an internal performance parameter only used during the training and validation phase of the NN. The performance metrics discussed in Section 3.5 are used to asses the trained NN's performance on a test data set. In conclusion, the training and validation data sets are used for creating the NN, while the testing data set is used for evaluating the performance. An example of a possible cost function is presented in Equation 3.4. z_{obs} is the observed output and z is the NN's model output which is presented by Equation 3.5.

$$J = \sum (z - z_{obs})^2 \tag{3.4}$$

$$z = \sum_j w_j y_j \tag{3.5}$$

Re-evaluation is necessary to optimise the overall outcome of the NN. The training process is stopped when one of the following conditions is satisfied [*Pérez et al.*, 2014]:

1. The cost function converges.
2. A maximum number of epochs is reached.
3. The cost function for a different dataset (eg. validation dataset) reaches a minimum.

The aforementioned discussion provides a first explanation of the basics of NNs. How to actually tweak these models and increase output performance is touched upon in the next subsections. Parameters such as the number of neurons, number of hidden layers, backpropagation method and hyper-parameters for internal NN tweaking will be part of the discussions in the next couple of subsections.

3.4.2 Forecasting Approach

No incoming data sources after the current time step t are allowed to be introduced into the NN model, as defined by the methodology of the drag problem in Section 3.1. However, the current form of the NN model, as presented in Equation 3.3, still requires that predicted density values are introduced into the model if one wants to make predictions for over one time step in the future, namely $t + 1$. This has as a consequence that if one tries to forecast density values up to a maximum of three days, the forecast error is propagated in time. Three alternative methods are discussed on how to solve the propagation of this forecast error, given the assumption that the sampling period, t_s, remains smaller than one orbital period. This is to allow the network to capture the orbital variations in the along-track density values.

Recursive Method

Equation 3.3 is only able to predict the value at $t + 1$, given a model input value at t. Three problems arise when using this definition of the drag problem. Firstly, the forecast horizon is limited by the sampling period of the time series, since it is only able to predict density values one time step ahead. Secondly, the model is not able to capture any valuable information in the past data of the time series such as *t-1*, *t-2*

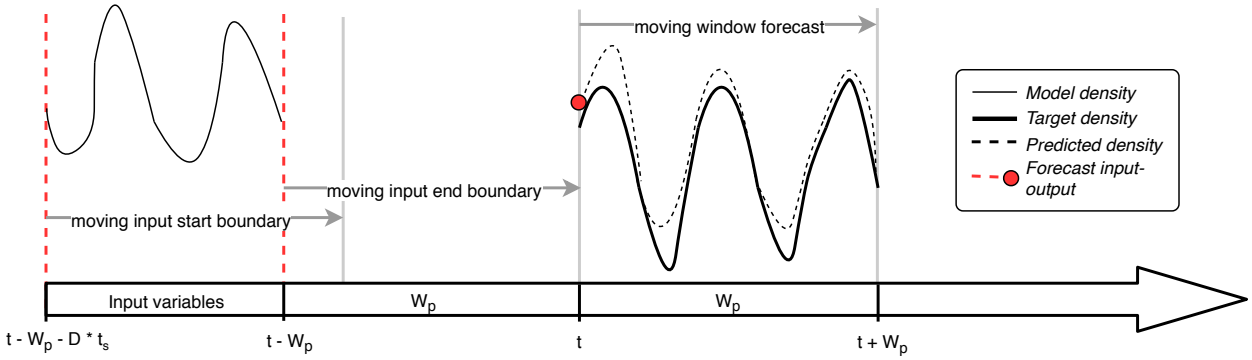


Figure 3.5: Schematic representation of the prediction problem at hand. Indicated are the time dependencies of the model, predicted and target densities. Red dot indicating predicted density value given the presented input variables. (W_p = Prediction Window, D = Input Delays, t_s = sample period)

and so on. Finally, as mentioned in the introduction of this subsection, an introduction of earlier forecasted values is required if one wants to predict density values at time steps *t+1* or later. In this case that means that a forecasted value at $t + 1$ serves as an input to the current model. This is not ideal since the model is trained to process the DTM2013 values as input. This problem is minor if the DTM2013 density and target values are similar. Since this is often not the case, an alternative method is preferred. An alternative method uses a sliding window in which a multitude of input values of the same time series are used in a time-delayed fashion to predict density values from t until $t + W_p$, where W_p is the prediction window. A schematic is presented in Figure 3.5. As a result, an alternative formulation of the drag problem is provided by Equation 3.6 and is adapted from the time-delay feed-forward neural network structure defined by *Pérez et al.* [2014].

$$y(t + W_p) = \psi\left(\sum_i w_i\Big(x_i(t), x_i(t - t_s), ...x_i(t - Dt_s)\Big) + b\right) \tag{3.6}$$

where the new element W_p is the prediction window, t_s is the sampling period of the data, and D is the number of delays in the input vector. The number of delays allows for the retention of the evolution of the inputs in time. In conclusion, the recursive method develops one NN model to make predictions from the current time stamp t up until $t + W_p$ given the input variables starting at [t - D * t_s - W_p , t - W_p] up until [t - D * t_s , t]. If one would like to increase the prediction window, the model is required to learn mathematical relations between its input and target values that are separated in time even more. This approach requires the least parameters to be learnt by the NN model. On the other hand, there can be an introduction of accumulated error sources for longer prediction windows. *Yaya et al.* [2017] did not encounter any degradation in performance for predictions of the solar indices below 30 days. It is therefore assumed to be a safe approach to solve the drag problem. Two alternative model approaches exist which also make use of the forecasting approach as defined by Equation 3.6.

Parallel Method

The parallel method proposes the development of an NN model with a multi-step output architecture. Thus, each output variable represents one of the model's forecast horizons. To develop such an NN model, a large network needs to be built with a large number of parameters that needs to be learnt. Although this method delivers a single network, the development and optimisation of such a network are not efficient in terms of CPU time and computational resources. The parallel method is therefore not implemented during this research study.

Successive Method

The successive method is similar to the recursive method in terms of the single output value it produces. The difference is that in the successive method a single NN model needs to be trained and developed to predict every time step after t. Thus, instead of using a single NN model that moves over all past input values, a network of models is trained on a data set in the past starting at [t - D * t_s , t] up until [t - W_p - D * t_s , t - W_p]. Although a single model in such a network is rather simple, one needs to train a large quantity of those NN models. As a conclusion, this is not preferred since a total number of W_p models needs to be developed accordingly.

In conclusion, the recursive method is implemented during all future analyses based on the arguments stated in this subsection. The recursive method, which is defined by Equation 3.6, allows the user to use one single NN model to make a prediction in the future as far as $t + W_p$. As a consequence, the only limitation would then be the size of the prediction window. In Chapter 5, a sensitivity analysis is performed on the optimal prediction window in terms of forecasting accuracy given a specific sampling period. If one wants to make a forecast beyond this prediction window, additional error sources are introduced.

3.4.3 Multilayer Perceptron Model

This section lays out the different Multilayer Perceptron Models (MLPs) that are effectively used in this research study. The models do use the theory and methodology which is discussed in the past two subsections. A univariate and multivariate MLP model are presented which have one and two or more input variables, respectively.

Univariate MLP Architecture

The univariate MLP model is a feed-forward neural network (FFNN), meaning that all information flows from the input vector to the output node. This input vector is solely constructed of past DTM2013 density values. The timeframe in which these input values are computed is defined by the specific problem definition. Figure 3.6 presents a schematic of a univariate MLP model with three hidden layers, each with a specific number of neurons and a bias added to each of those neurons. The model has one output to approximate the true along-track density values experience by the Swarm satellites. This output is constructed by a linear combination of its input values. In this report a specific model architecture is named an N1-N2-... MLP model (eg. 8-8-6 MLP model for an MLP model with eight neurons on its first and second hidden layer and six neurons on its third hidden layer).

Multivariate MLP Architecture

The multivariate MLP model is similar to the univariate MLP model since all information flows from the input vectors to the output node without information flowing back into the model. Despite this similarity, the multivariate model used in this report has the ability to process two input time series. The first input vector is constructed of past DTM2013 values, while the second input vector contains additional information in the format of solar or geomagnetic indices. Figure 3.7 presents the architecture of the multivariate MLP model used in this report. A multitude of options do exist when one wants to connect two time series to a model such as the one presented in Figure 3.6. For example, the Merge layer, as presented in Figure 3.7, could be omitted such that the two different time series are fully connected to the first hidden layer. During this thesis project different architectures were developed and tested. Finally, it was concluded that adding an additional layer of nodes (Merge layer) for each time series separately

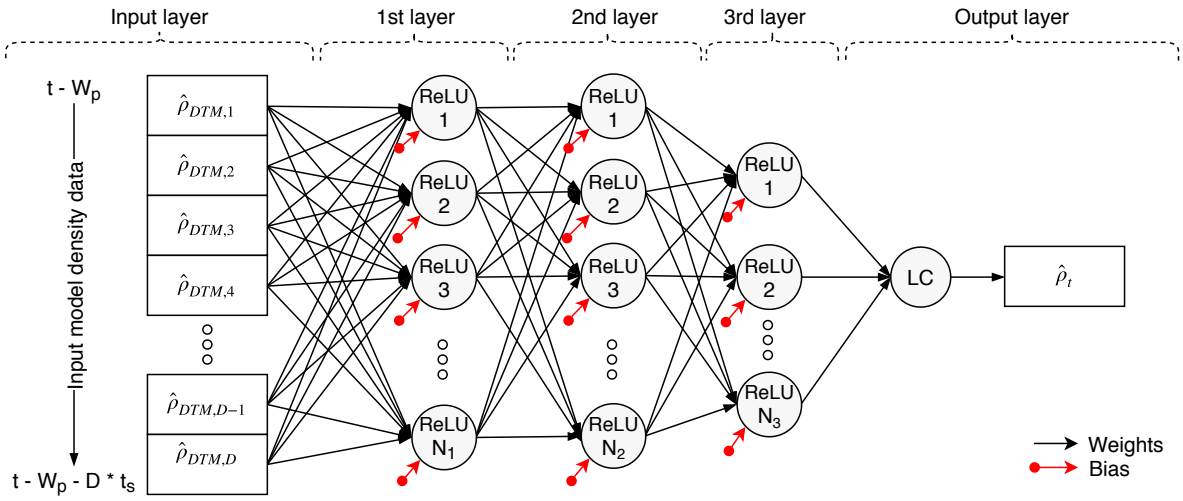


Figure 3.6: Univariate Multilayer Perceptron model with three hidden layers and one linear neuron in the output layer to forecast the along-track density values. Past DTM2013 density values serve as input values for the model. (LC = Linear Combination, N = Number of Neurons, ReLU = Rectified Linear Unit)

performed best in terms of forecasting capabilities. This merge layer adds two additional Input layers (I), while still relying on the properties of the tweaked univariate model. In this report a specific model architecture is named an I1I2-N1-N2-... MLP model (eg. 99-8-8-6 MLP model for an MLP model with two input layers with each nine neurons, eight neurons on its first and second hidden layer and six neurons on its third hidden layer). Two additional assumption are required before moving on with the discussion on the type of additional input variables can be used. First, the univariate model will be the model of which the hyperparameters are tweaked. It is assumed that these hyperparameter settings still hold for this multivariate model. Second, the number of neurons in both input layers are identical and are experimentally decided to be larger than the number of neurons in the first hidden layer. The exact model architecture is discussed in Chapter 5.

The multivariate model does not limit itself in the number of variables it processes as input. The hypothesis is that the additional information sources do uncover extra relations between the input and target density values with the goal to increase the forecasting capabilities. Table 3.2 presents the Pearson correlation (see Equation 3.12) factors between the POD-derived density values and the available solar and geomagnetic indices as a measure of the strength of the linear relationship between the two time series. Despite it not being the best approach to examine the relationship between these two variables, it gives a first indication of how well an additional index could potentially enhance the learning ability of the MLP model, and thus also the forecasting accuracy. Three insights are discussed next when looking at the numbers in this table. Firstly, the D_{st} index has a negative correlation factor since the value on its own is a negative value. Secondly, the low absolute value of the correlation factors is a result of the retarded heating of the thermospheric layer as a reaction to a change in the activity of the Sun. These correlation coefficients are thus best interpreted by comparing them relative to one another. Finally, it is not advised to re-use any of the indices that is also used as input for the DTM2013. This is to avoid to force the MLP model to learn the same relations. Therefore, K_p and F_{30} are neglected in this report. Hence, the D_{st} index is the only candidate in terms of additional, geomagnetic information. F_{15} seems to be the best candidate of all solar indices due to it having the highest correlation coefficient. Nevertheless, the performance of all available solar indices will be examined in Chapter 5.

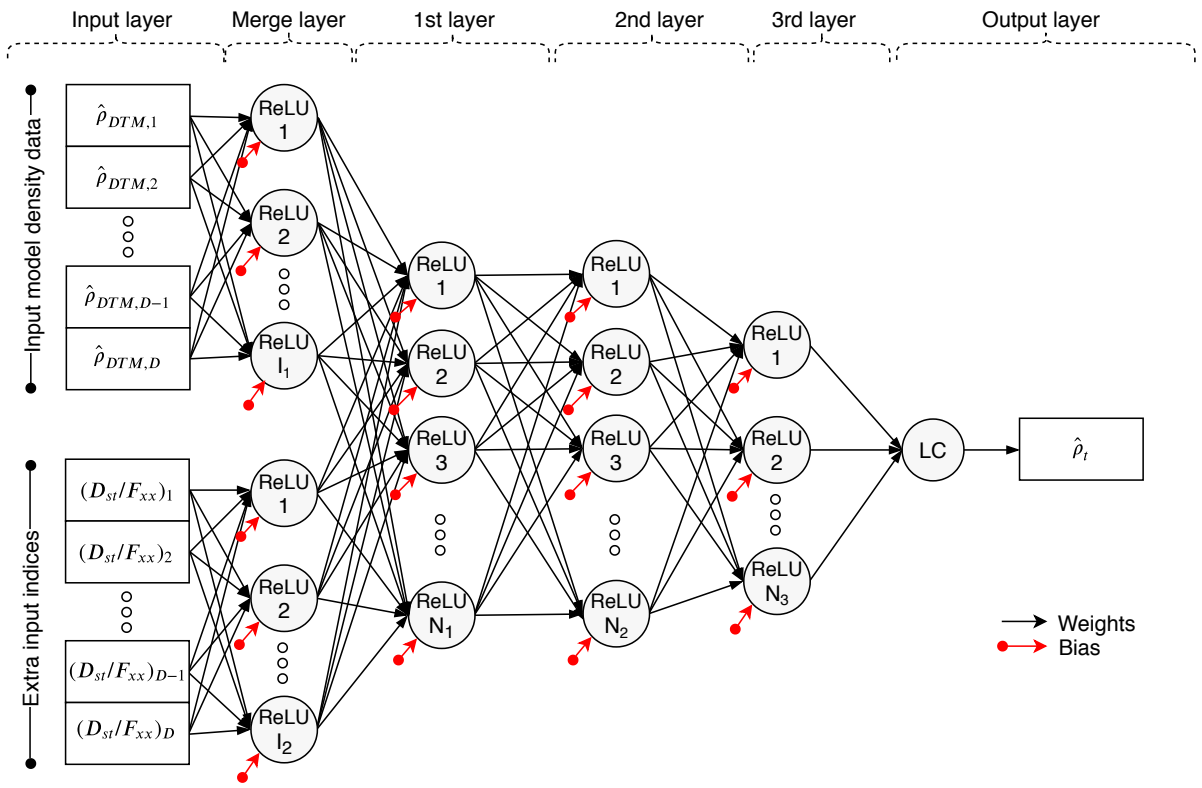


Figure 3.7: Multivariate Multilayer Perceptron model with one hidden merge layer, three hidden layers similar to the ones of 8-8-6 MLP and one linear neuron in the output layer to forecast the along-track density values. Past DTM2013 density values and additional solar or geomagnetic indices serve as input values for the model. (I = Input Neurons, LC = Linear Combination, N = Number of Neurons, ReLU = Rectified Linear Unit)

Hyperparameter Settings

Hyperparameters are the variables that define an NN model's architecture and determine how the network is eventually trained. The implemented set of hyperparameters is similar for both the univariate and multivariate NN models. The hyperparameters are kept the same throughout the whole research project and are decided on based on literature, best practices and quick software prototyping.

First, the amount of *validation data* is elaborated on. The validation data set is a subset extracted from the training data set. Subsection 3.4.1 mentions three conditions for which the training process is stopped. One of those requirements states that the training can be stopped if the cost function for the validation data set reaches a minimum. Thus, a split of the training data set into the actual training data and validation data is required. There is no specific standard in what the ratio training-validation data needs to be. However, after having conducted some quick experiments on the STHighNS training data set (see Section 4.3), it is decided that 20% of the training set is used as validation data. As a consequence, 80% of the number of data points in Table 4.1 is eventually used as training data to tweak the internal parameters.

The aforementioned internal *cost function* is discussed next. This cost function is also called the loss function in the machine learning space. In this research study the Mean Squared Error (MSE) is used as the internal cost variable to minimise. The MSE is defined by the squared difference between the forecasted and target density values. It is presented by Equation 3.7 and basically represents the average

Table 3.2: Summary of the Pearson correlation coefficients between the DTM2013 along-track densities and potential solar and geomagnetic indices for the multivariate MLP model.

Variable	**Pearson Correlation Coefficient**
DTM2013	1.000
K_p	0.080
D_{st}	-0.178
$F_{3.2}$	0.291
F_8	0.309
$F_{10.7}$	0.285
F_{15}	0.330
F_{30}	0.330

of the errors. Since the errors are squared, the MSE does not care about the direction of the error, but it does penalise the larger errors more heavily. Many other, more advanced, loss functions exist and are included in the Keras library, but the MSE is preferred since it has nice mathematical properties which allow the model to efficiently compute gradients during the backpropagation process.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{\rho}_i - \rho_i)^2 \tag{3.7}$$

Next, the *backpropagation algorithm* is discussed. Backpropagation is a procedure first implemented by *Rumelhart et al.* [1986] to repeatedly adjust the weights of the connections in the network. This to minimise the error between the predicted and target values while training the NN model. Adjusting those weights can be done by using many different methods. However, *Ruder* [2017] discusses a method called *gradient descent*, which is a way to minimise the cost function J by updating the model's parameters in the opposite direction of the gradient of that cost function with respect to those parameters. The paper states that the *learning rate* defines the step size that needs to be taken to reach this minimum. Again, many different gradient decent procedures do exist. The one used in this research study is the Adaptive Moment Estimation (Adam) method. *Kingma and Ba* [2015] show empirically that Adam is a method in stochastic optimisation that requires little memory and is computationally efficient. Above all, they have empirically shown that the method works well in practise and compares well to other gradient descent methods with adaptive learning rates.

The neuron in an NN model has the capability to combine a weighted sum of its inputs and additional biases. Afterwards, it feeds this combination of information into a non-linear *activation function*. This activation function maps this input into a specific range of values. Similar to the different backpropagation methods, a multitude of different activation functions are developed as well. In this research study the Rectified Linear Unit (ReLU) function is used in each hidden layer of the NN models. *Glorot et al.* [2011] first highlighted the benefits of using this activation function in specific. The activation function is defined by Equation 3.8 and it basically implies that the activation function outputs the exact input value if this value is positive, otherwise it gives zero as an output.

$$\text{ReLu(x)} = \max(0, \text{x}) \tag{3.8}$$

The ReLu function is therefore straightforward to implement and computationally efficient since it basically only needs to compute the maximum of two values. Another benefit is that this function looks and acts like a linear activation function. *Glorot et al.* [2011] also argues that the gradients flow back well during the backpropagation phase during training due to this linearity. Another benefit is that the ouput

Table 3.3: Summary of the hyperparameter settings that are implemented in both the univariate and multivarite MLP models.

Hyperparameter	**Setting**
Validation data	20% of testing data set
Cost function	Mean square error (MSE)
Stochastic gradient descent	Adaptive Moment Estimation (Adam)
Activation function	Rectified Linear Unit (ReLu)
Bias	Uniform distribution [0,0.05]

of specific neurons can be set to true zeros, instead of very small numbers compared to other activation functions. This directly excludes a neuron from the training algorithm, which then speeds up the training time. However, a downside of using this method is that some neurons can completely disappear during the training process, also known as the vanishing gradient problem [*Sussillo and Abbott*, 2014].

A countermeasure to avoid this vanishing gradient problem is to add a small, positive *bias* to the input of each neuron at the start of the training phase. This additional bias is already indicated in Figure 3.4. The Keras library offers a multitude of different bias initialisers that determine the format in which such biases are generated. In this research report, a uniformly distributed bias scheme is applied with the biases values ranging from 0 to 0.05. All hyperparameter settings are summarised in Table 3.3.

3.4.4 Nested Cross-validation

This subsection presents a method which allows to compare different NN model architectures and is often used when tuning its internal hyperparameters. In this research study this method is used to test for the optimal number of hidden layers and the number of neurons in each of those hidden layers. The original method is called the K-fold cross-validation method and is one of the more popular methods in the machine learning space. This method requires a data set to be divided into K different groups of equal size. The first group is used as a test data set, while the model is trained on all other K-1 data sets. Afterwards, the complete data set is shuffled and the process is repeated k times [*Bergmeir and Benítez*, 2012].

However, this K-fold cross-validation method cannot be used in time series problems due to the properties inherently connected to time series. Namely, if a time series is split into different subgroups and shuffled afterwards, the temporal dependencies are broken. This means that a model does not have the ability to learn from the past to predict the future variables. A different approach, called the nested cross-validation approach, is presented in Figure 3.8 and described by *Varma and Simon* [2006]. In this method the complete training data set is split into six equally sized blocks of data. Afterwards, five different experiments are performed where for each experiment the amount of training data increases. The model is then trained on 80% of the this training data and uses the other 20% for validation purposes. The internal parameters are then tuned until the cost function converges to a minimum. After the model is trained, the model's performance is tested on the test data set of which the amount of data remains the same size for all five experiments. The overall performance of all five tests of this Nested Cross-Validation is the average of the performance metrics of every split in order to compute a robust estimate of the model error. The performance metric used in this report is the RMSE. This overall metric is presented in the following equation:

$$\mathrm{RMSE}_{\mathrm{average}} = \frac{\sum_{k=1}^{5} \mathrm{RMSE}_{\mathrm{k}}}{5} \tag{3.9}$$

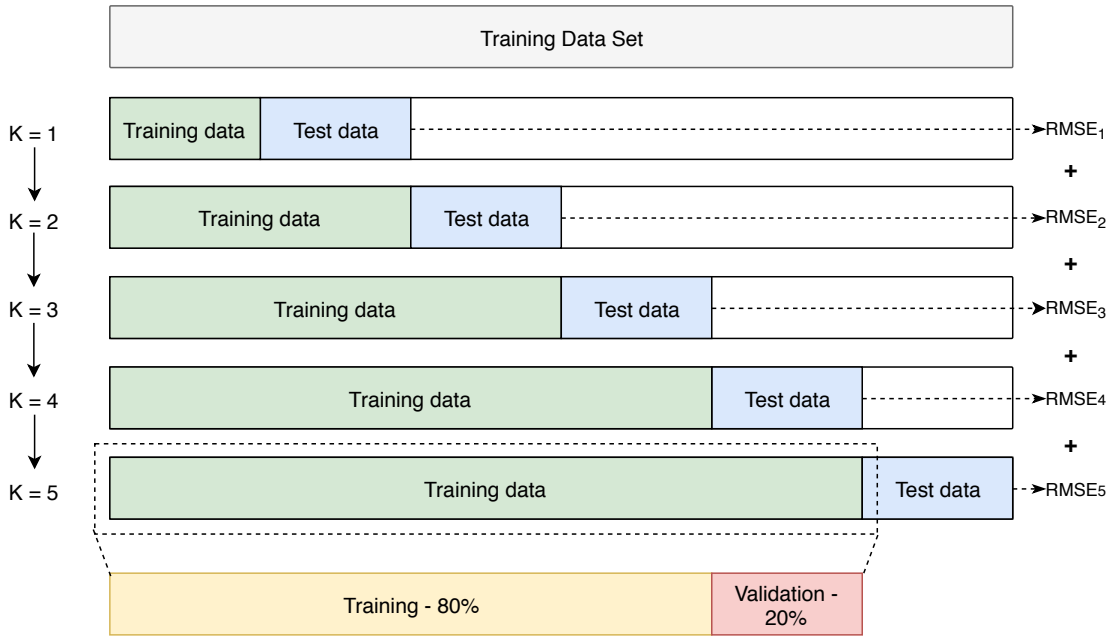


Figure 3.8: Schematic representation of the Nested Cross-validation for one specific Neural Network model with K the number of the experiment. (RMSE = Root Mean Squared Error)

The advantage of using this method is that it reduces the forecasting bias since five different models are trained on five different data sets with a different number of data points. Additionally, this can give an insight into how the developed MLPs behave with varying amounts of training data.

3.5 Performance Metrics

To assess the overall performance of a model's prediction capabilities, different metrics are used. These metrics are called *performance metrics* and need to be checked and validated before comparing models on a one-to-one basis since definitions could vary in literature. The performance metrics that will be discussed in this section and are represented in Equation 3.10, Equation 3.11 and Equation 3.12 are similar to the ones used by *Pérez et al.* [2014]. This is to be able to compare the outcome of the NN models and to validate their performance. Afterwards, they are directly applied to the verification data set used in Section 2.3. The target values are considered to be the POD-derived density values of Swarm-A on 1 August 2014, while the DTM2013 verification data set is used as the model output. The density is a function of different factors as expressed in Equation 2.1. Part of these factors needs to be forecasted which means that the prediction of the density values is affected by a multiplication of different error sources. Hence, besides the errors introduced due to the variations in time, a model's output is also affected by model errors. To reduce these model errors during final analyses the *natural logarithm* is taken of the density values. This helps in stabilising the variance of the density values.

The *Root Mean Square Error* (RMSE) is considered to be the performance metric to model the accuracy of a certain forecast. Equation 3.10 presents the definition of the metric used in this report, where $\hat{\rho}$ and ρ are the modelled and target density value, respectively. The RMSE combines the magnitude of the errors in the predictions for various timestamp into a single metric. A value of 0 would mean a perfect prediction, which is rarely ever achieved. In general, a lower RMSE is better than a higher one. The benefit of RMSE is that it penalises large errors and the scores are in the same units as the forecast values.

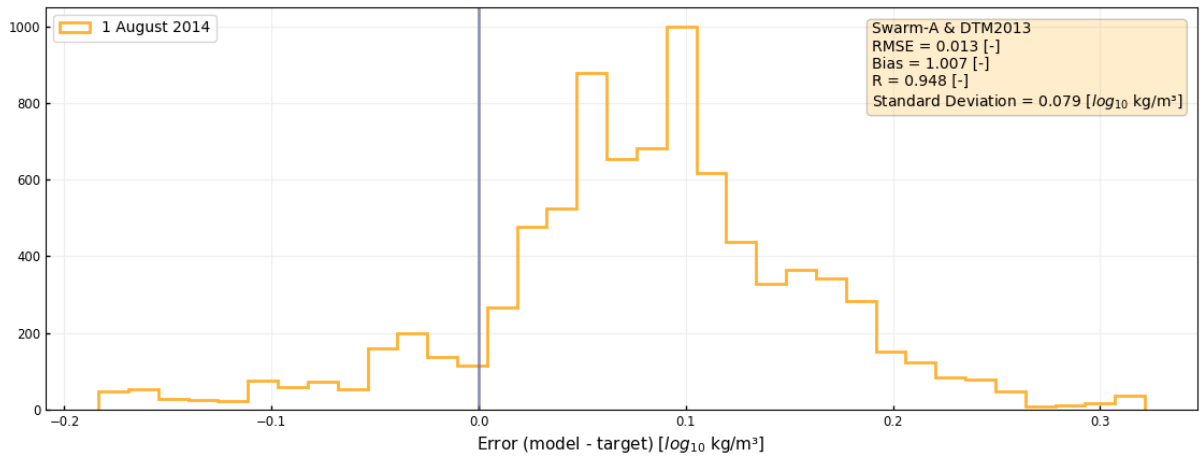


Figure 3.9: Histogram of the residuals for the Swarm-A POD-derived density data set and the DTM2013 model output on 1 August 2014. This day has also been used to validate the DTM2013 in Section 2.3.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{\rho}_i - \rho_i)^2} \tag{3.10}$$

Pérez and Bevilacqua [2015] define a metric to measure the *bias* (B) that occurs when there are consistent differences between the target values and the forecasts of those values. The definition used in this report is the mean of the ratios of the targets to the forecast values, as presented in Equation 3.11. A good forecast shall be unbiased and to be valid for this expression it shall be as close as possible to 1.

$$\text{B} = \frac{\sum_{i=1}^{n}\frac{\rho_i}{\hat{\rho}_i}}{n} \tag{3.11}$$

The *Pearson Correlation Coefficient* (R) is used to quantify the correlation or linear dependency between the forecast and the past observed values at the same time instance. This coefficient also helps in identifying if certain solar and geomagnetic indices have an impact on density fluctuations, as presented in Table 3.2. The Pearson correlation coefficient is described in Equation 3.12 [*Pérez and Bevilacqua*, 2015]. Where $s_{\hat{\rho}}$ and s_{ρ} are the standard deviations of the model outputs and targets, respectively.

$$\text{R} = \frac{\frac{1}{(n-1)}\sum_{i=1}^{n}(\rho_i - \bar{\rho}_i)(\hat{\rho}_i - \bar{\hat{\rho}}_i)}{s_{\hat{\rho}}\, s_{\rho}} \tag{3.12}$$

To study the errors in the forecasts more systematically and to identify trends, *histograms of the residuals* are used to present the variation in errors [*Yaya et al.*, 2017]. The width of the histogram shows the uncertainty of the forecast. These performance metrics are all presented in Figure 3.9 and are based on the aforementioned validation data set. One can conclude that the DTM2013 consistently predicts a higher density value in comparison to the POD-derived density values. However, due to the low RMSE and correlation, one can conclude that the DTM2013 does capture the variability in the data.

Elvidge et al. [2016] created the *Modified Taylor Diagram* (MTD) which is a tool in which the complexity of different models can be captured and compared in a rather organised graph. The Taylor Diagram was first developed by *Taylor* [2001], after which *Elvidge et al.* [2014] made a modified version. Another benefit is that the software used to develop this diagram is freely available on GitHub [1]. However, the

[1] Adapted from `https://github.com/seanelvidge/python_general/blob/master/modifiedtaylordiagram.py`

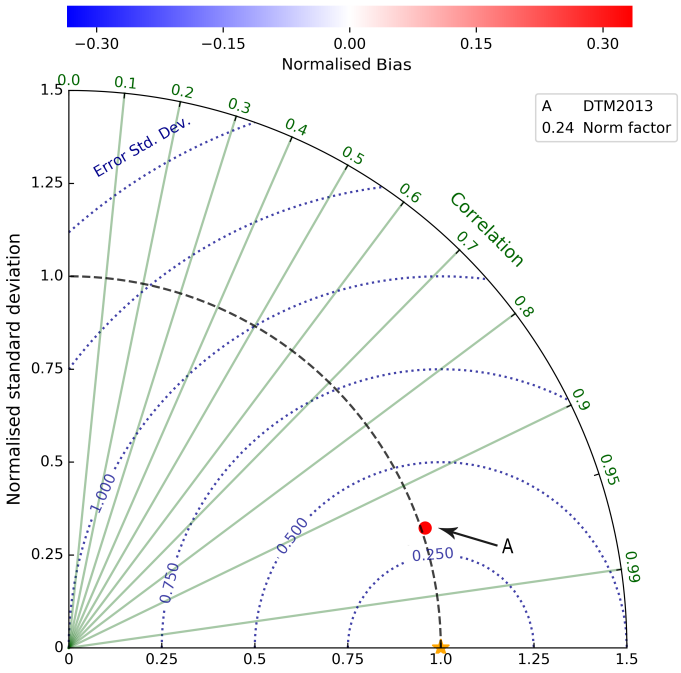


Figure 3.10: Modified Taylor Diagram for the Swarm-A validation data set and the DTM2013 model output on 1 August 2014.

definitions for some of the metrics used in this MTD are different from the ones discussed earlier in this section. This does not form any problem, since the graphical representation of the MTD will primarily be used to compare the forecast performance of different models. In a perfect forecast, the model output shall approximate the target values as well as possible. In the MTD this means the model indicator shall be as light coloured as possible and shall be located near the yellow star, which is characterised by a normalised standard deviation and correlation of 1 and an error standard deviation of 0.

The azimuth angle in the MTD represents the correlation between the target and modelled density values. This metric is similar to Equation 3.12. The *Norm Factor* is the standard deviation of the POD-derived density time series. This factor is used to normalise both the bias as the standard deviation of the modelled density values. The bias is presented by the colour scale and is defined differently when compared to Equation 3.11. The definition of the bias is presented by Equation 3.13. The *model error standard deviation* (E) is defined by Equation 3.14 and is a measure for the model variation. It is especially useful in comparing distributions that are usually strongly correlated. If two distributions are identical, the model error standard deviation will be zero. An example of such an MTD for the validation data set of Section 2.3 is shown in Figure 3.10.

$$\mathrm{B}_{\mathrm{MTD}} = \hat{\rho}_i - \rho_i \tag{3.13}$$

$$\mathrm{E}_{\mathrm{MTD}} = \sqrt{s_{\hat{\rho}}^2 + s_{\rho}^2 - 2 s_{\hat{\rho}} s_{\rho} R} \tag{3.14}$$

Chapter 4

Data Preparation

This chapter elaborates on all the data preparation-related procedures that need to be performed before the time series data sets can be used by the forecasting models. The flowchart in Figure 4.1 shows the different steps that are taken to prepare the training and test data sets for their implementation in the Neural Network (NN) models. First, the training and test data sets are defined. The training data sets will only be used to train and validate, thus develop, the NN models. These trained models will then be tested on a set of test data sets. Next, the availability of all density values and input indices for all the pre-specified data sets are checked. Availability in the sense that they need to be published by their corresponding observatory or agency. Afterwards, these data sets are checked on anomalous and missing values. Different methods to cope with such events do exist. Moreover, the temporal structure embedded in the time series creates an extra order to the observations. Consequently, extra assumptions need to be checked for such that the forecasting models can still be used. One of those assumptions is that a set of density observations needs to be stationary, meaning that the summary statistics of these observations needs to remain constant. The observations in a stationary time series are thus not dependent on time. Based on the outcome of those stationarity tests, extra data transformations are required to guarantee this consistency over time. Finally, the time series are resampled based on the best NN model performance.

All different steps will be discussed more thoroughly in the consecutive sections of this chapter. Section 4.1 elaborates on the necessary procedures to deal with missing and erroneous data. The second section of this chapter, Section 4.2, depicts the set of tests needed to prove if a time series is stationary. Finally, in Section 4.3, the different data sets of training/validation and test data are listed.

4.1 Data Preparation

In this section the procedures to prepare the raw time series inventory are discussed. All different steps are performed on each training and test data set. The Pandas library is used to implement these different procedures. This Pandas library - as part of the Python ecosystem - offers the flexibility in the manipulation of databases, time-series handling and missing data handling. The library also helps to extract statistical data from a time series data set and visualise time-indexed databases.

4.1.1 Timing

The relevance and effect of the time at which the data is made publicly available are elaborated on. As input for the DTM2013, four data sources are required. These are the location and time of the Swarm

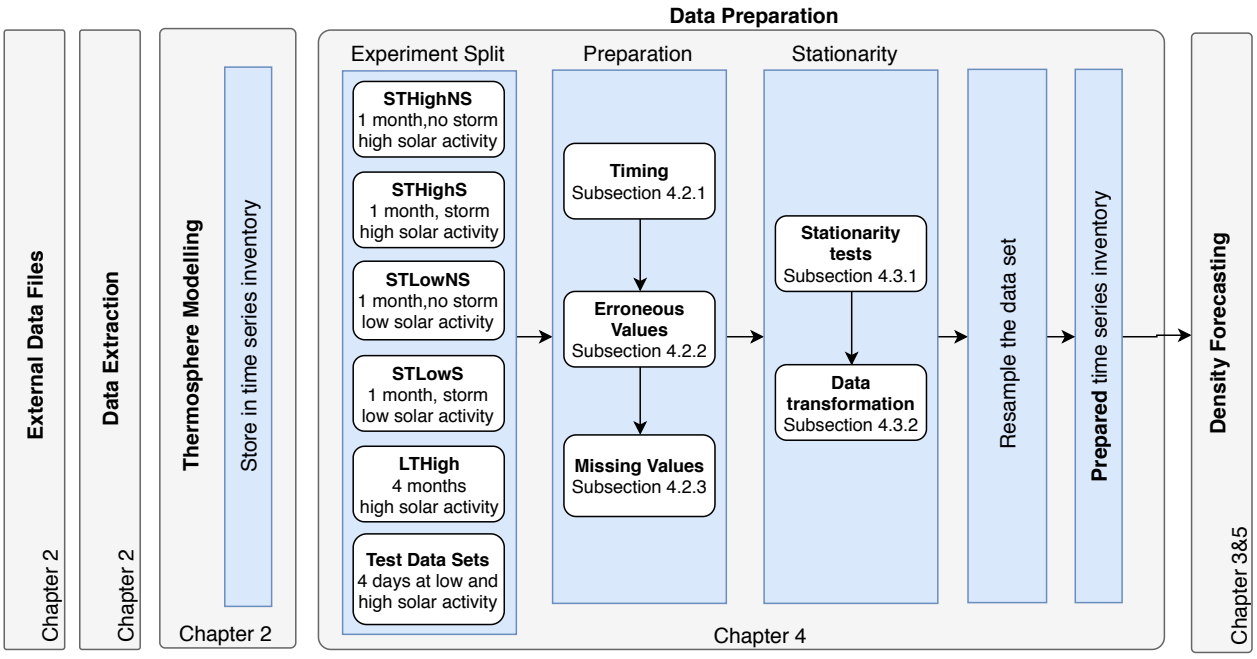


Figure 4.1: High-level code overview of the data preparation procedures.

satellites, as well as the daily solar indices and 3-hourly geomagnetic indices. Because this study is only focused on training the models on past data and testing them on days in the past, it is not required to incorporate up-to-date data sets in order to form a conclusive answer on the research question. The data files were all extracted from their corresponding servers at the beginning of this thesis project. If any of those files would have no significant version updates by the publishing organisation, they are considered minor and are not corrected for. However, it is relevant to discuss this topic shortly with an eye on future developments and the operational relevance of this work.

The provisional values of the solar indices at different wavelengths are published by Toyokawa & Nobeyama (F_{30}, F_{15}, F_8 and $F_{3.2}$) and Penticon ($F_{10.7}$) on a daily basis around 12:00 CET and 5:00 CET, respectively. The definite values with minor adjustments are published once per month, with a latency of 15–40 days. As mentioned before, those minor adjustments are not further taken into account. The solar indices are thus available in a timely manner for daily operational use. However, for high accuracy forecasts the timegap of 17 hours between the measurements of Penticon and Toyokawa & Nobeyama could form a problem and could ask for a resampling technique to be implemented. The GFZ German Research Centre For Geosciences located in Potsdam supports and operates 13 geomagnetic observatories worldwide to finally publish a list of K_p values on a monthly basis. Since the geomagnetic indices provided by these observatories are not published on a daily basis, in contrast to the solar indices, they are not suited for operational use. Alternatively, forecasted values can be used instead. These could be forecasted using one's own propagation software, but are often offered by the publishing organisations themselves for a multitude of future time steps.

4.1.2 Anomalous Values

Occasionally, anomalous values are present in the observed solar and geomagnetic indices are caused by a number of different effects. Some of the outliers could be detected by eye and have an instrumental or human origin, while others could be the result of non-thermal emissions that originate from solar flares.

One can improve the detection of these anomalous values by comparing the results of different observatories. *Yaya et al.* [2017] have identified that in the case of the solar indices the number of average outliers is smaller than two days per year. On top of this, it is difficult to categorise the different outliers as being caused by solar activity or by human/instrument interaction. The latter one even has a limited impact on the thermospheric density measurements. It is concluded that the outliers in the time series of solar and geomagnetic indices are not significant enough to take into account during this research project, since past, processed data is used and both the solar and geomagnetic indices are the product of a multitude of observatories.

On the other hand, the main outliers that are taken into account are the ones present in the Swarm satellite constellation data. The largest part of these outliers are flagged by processing efforts and are most often related to anomalies and manoeuvres. For information purposes one could use the Swarm documentation on the anomalies and manoeuvres history. The flagged density data are removed from the time series, thus generating missing values. The method to cope with those missing values is discussed in the next subsection.

4.1.3 Missing Values

Both the Swarm data sets, as well as the observed solar and geomagnetic indices occasionally have missing values for three reasons. The first reason is related to the removal of the anomalous values, as discussed in the previous subsection. Secondly, a data gap of the observed values could be caused by an interruption in measuring that specific value. This type of missing data points is not that common since the observatories often publish their own reconstructed values instead. The third reason that causes missing values in the inventory is due to actions taken in the post-processing phase of the entities that provide those data sets. This is primarily visible in the case of the Swarm satellite constellation, in which multiple days worth of data are not available online. However, the majority of time gaps in the training and test data sets are a result of the removal of the flagged timestamps of the Swarm satellites' operations. A multitude of training data sets has been identified in which a marginally low number of missing and/or flagged data is present. Consequently, this has the benefit that those time gaps are not required to be filled with interpolated values, which can cause extra bias in the final results. If time gaps smaller than ten minutes do occur the short timespan of the time gap is filled with linearly interpolated values. This technique has shown to be a feasible approach for missing density values in the research of *Pérez et al.* [2014]. Other clever techniques such as masking the time gaps with extreme values, such that the machine learning algorithm is learnt to identify those, are not part of this thesis report.

4.2 Stationary Time Series

Stationary time series are characterised by having no trend or seasonal effects. The observations in a stationary time series are therefore not dependent on time [*Wayne*, 1995]. When a time series is stationary, it can be easier to model, since the model has the capabilities to be used on all data points of that same time series. The majority of statistical modelling methods are only effective if the time series under consideration are stationary. *Qi and Zhang* [2008] conclude that stationarity is not a hard requirement, but beneficial, for NN models since they are nonlinear non-parametric adaptive models. However, the stationarity tests for the DTM2013 density time series are still used as an additional check. Subsection 4.2.1 will discuss two methods which are used to test for stationarity, while Subsection 4.2.2 will give an overview of the transformation methods used to transform the time series in this analysis.

4.2.1 Stationarity Tests

The stationarity of a time series can be tested by using a number of different methods. Two methods are discussed in this section. The first method focuses on providing a summary of statistical parameters for different, random partitions of the data series. The second method uses a statistical test called the Augmented Dickey-Fuller test to check if the stationarity conditions are met or not.

The first method is discussed next. In the case of the thermospheric density forecasting problem, the time series are considered non-stationary if their mean and variance vary over time. This causes a drift in the different concepts a model tries to capture. To test if such a drift is present, a time series is partitioned into different, small subsets for which the mean and variance are calculated. An additional histogram helps in visualising these parameters better. If these statistical properties are significantly statistically different, the density time series are considered non-stationary.

The second method, the Augmented Dickey-Fuller (ADF) test, is discussed now. The ADF test is a type of statistical test called a unit root test [*Wayne*, 1995]. A unit root test is often used as a tool to determine how strongly a time series is defined by a trend. The ADF test uses an autoregressive model and optimises an information criterion across multiple different lag values. The ADF test, like many other hypothesis testing methods, is specified in term of a test statistic. *Berger and Casella* [2001] describe this test statistic as a numerical summary of the data sets which is used to form a conclusive answer on the hypothesis test. If the ADF statistic is negative, the time series is considered stationary. The ADF test is incorporated in the Statsmodel module of the Python programming language.

The Null Hypothesis, H0, of the ADF test states that the time series can be represented by a unit root. Consequently, the time series is not stationary. The Alternate Hypothesis, H1, means the null hypothesis is rejected and the time series can be considered stationary. When the null hypothesis is rejected the time series does not have a time-dependent structure. One way to interpret the significance of the result of a hypothesis test is by using the p-value from that test. This p-value defines the probability that the null hypothesis is rejected if this p-value is below a pre-specified threshold value. In this report threshold values of 10%, 5% and 1% are chosen. Contrarily, the null hypothesis is accepted if the p-value has a value above those threshold values. As a summary the different conditions for stationarity are listed below:

1. *Null Hypothesis*, H0: If failed to reject, the time series has a unit root, meaning it is non-stationary. It has some time dependent structure. The null hypothesis cannot be rejected if the p-value is greater than 0.05.
2. *Alternate Hypothesis*, H1: If H0 is rejected, the time series does not have a unit root, meaning it is stationary.

Statistical tests such as the ADF test can only be used as an indicator to which degree a null hypothesis can be accepted or otherwise rejected. The reason is that statistical tests do come forth from a set of assumptions about the time series and are thus unique for each given problem. Nonetheless, the ADF test can be used as a quick check to determine if the density time series is stationary or not. This test will be performed on the training data sets presented in Section 4.3.

4.2.2 Data Transformation

The data sets are transformed first because stable, stationary data sets are needed before they are fed to the NN model. A method which is often used is to take the *natural logarithm* of those density values. This helps in stabilising the variance of the density values. This method is extensively discussed by

Doornbos [2012]. Another possible method is to rewrite the thermospheric density to *exospheric temperature* values which by definition are more stable than the density values. This allows to better compare density values at different time steps and locations [*Choury et al.*, 2013]. Lastly, *density ratios* can be used to normalise the observed density values with the model density to suppress the range to a smaller scale. In this thesis study the method using the natural logarithm is implemented due to its simplicity and interpretability.

Differencing a time series is another approach which is often used to make a time series stationary. When differencing a time series a new time series is created by literally taking the difference between its value and their preceding values. This technique is often implemented in forecasting models used in the financial world, with the goal to stabilise the mean of a time series and to uncover any extra feature hidden in the original time series. However, the literature review presented in the literature study report by *Claes* [2018] concludes that the differencing method is not often used in machine learning models which try to forecast space weather phenomena. Consequently, this gives a first indication that this method might not help in improving the forecasting accuracy. Nevertheless, this is checked for the current data sets that are described in Section 4.3. In conclusion, a larger relative change in variance and the mean across the different parts of a random time series data set has been identified in comparison to the change when not differencing the modelled density data set. Thus, it is concluded that differencing is not that effective in this particular case of space weather forecasting and is not touched upon in rest of this research project.

Some NN models have problems with vanishing gradients [*Domingos*, 2012]. The weights of such an NN model are updated each iteration to enhance the final performance of the network. When those weights change, a gradient is computed to measure how effectively the network is improving. This vanishing gradients problem occurs when the weights need to pass through multiple layers of multiplication, until they reach a point where the weight changes are so small that no gradient can be computed anymore (eg. gradient explodes or vanishes). Another benefit of taking the natural logarithm of the density values is that it helps in addressing this issue [*Pérez and Bevilacqua*, 2015].

4.3 Training and Test Data

In this section the training and test data sets are presented and discussed. The selection of the ideal training data sets is based on a couple of different aspects. Firstly, a set of training data sets needs to be defined such that the research questions can be answered accordingly. This thus means that the data sets need to differ in terms of solar and geomagnetic activity, as well as the number of data points. Secondly, the amount of flagged and/or missing data points shall be reduced to a minimum since interpolation techniques are based on assumptions which can never represent the physical relations embedded in the DTM2013 model. Finally, the transformed model density time series should ideally be defined as a stationary time series. This to make the prediction capabilities of the ML model as robust as possible if used as a forecasting method for a random test day. The stationarity principle of the chosen training data sets is discussed in this section.

Appendix C depicts the average solar and geomagnetic activity for each month of the available database in which no missing Swarm data is present. The number of flagged time steps are indicated as well. These flagged data points are removed in the final density time series and thus cause additional missing time steps. The monthly data sets with a minimum of flagged timestamps are thus preferred. The

Table 4.1: Final training data sets including the characterisation of the solar and geomagnetic acitivy, number of samples at a resampling rate of 30s and operations performed on the time series. ADF-test results are added to check for stationarity. (*ST* = Short Term, *LT* = Long Term, *S* = Storm, *NS* = No Storm).

ID training data set	**STHighNS**	**STHighS**	**STLowNS**	**STLowS**	**LTHigh**
Begin data	01/12/2014	01/09/2014	01/10/2017	01/10/2016	01/09/2014
End data	31/12/2014	30/09/2014	31/10/2017	31/10/2016	31/12/2014
Average $F_{10.7}$ [sfu]	159.1	147.71	76.42	86.11	154.29
Average K_p [-]	2.29	2.03	1.83	2.45	2.07
Maximum K_p [-]	4.67	6.33	5.67	6.33	6.33
Resampling rate [s]	30	30	30	30	30
Nb of values	89278	86400	89280	89280	350401
Missing values approach	linear interpolation	N.A.	N.A.	N.A.	linear interpolation
Data preparation approach	log_{10}	log_{10}	log_{10}	log_{10}	log_{10}
ADF test results:					
- ADF statistic	-49.325	-31.893	-18.53	-26.571	-42.589
- p-value	0.0	0.0	0.0	0.0	0.0
Stationarity based on ADF	yes	yes	yes	yes	yes

approximately 30-day periods are based on the months in a calendar year for two reasons. Firstly, the conventionally accepted months are more convenient to refer to throughout every phase of this research report. Secondly, most solar and geomagnetic indices, as well as measured density values, are available on a monthly basis which makes it more convenient for the reproducibility of the results presented in this report. Two monthly data sets are defined for both high and low solar activity. One including geomagnetic storms, while geomagnetic storms are absent in the other one. The data set with high solar activity and low geomagnetic activity is used as the core training data set, since forecasting the effect of geomagnetic storms is an ambitious challenge with the current state-of-the-art methods and is not the focus of this report.

Finally, four short term (1 month) and one long term (4 months) training data set are selected from all possible candidates. The properties of all different data sets are presented in Table 4.1. They include the average solar and geomagnetic indices used as input for DTM2013, the final number of data points at a specific resampling rate and the results of the Augmented Dickey-Fuller test for the DTM2013 data. The theory behind all properties has been discussed in the previous sections. A graphical representation of these training data sets can be found in Section C.2.

Based on the negative value of the ADF statistic, one can conclude that all five data sets pass the rejection of the Null Hypothesis, and are thus considered stationary. Additionally, each short term training data set is divided into three equal parts. As a redundant tool to test for stationarity, the mean and variance are computed for each part and are graphically presented via histograms. Those histograms are presented in Figure 4.2, Figure 4.3, Figure 4.4 and Figure 4.5 for the monthly training data sets. In order for the training data sets to be stationary, the change in mean and variance across all three parts is required to be constant. For all four short term training data sets the mean values remain constant. Consequently, this indicates that on a 1-month timespan the trend present in the density values is rather absent. On the other hand, the change is variance is more significant across all three parts, meaning that the seasonality within the density values is still present. Countermeasures to get rid of this phenomenon will be discussed later on in the report if the seasonality seems to be a significant factor in training the NN model. This difference in variance becomes more clear in the training data sets where a storm is present when

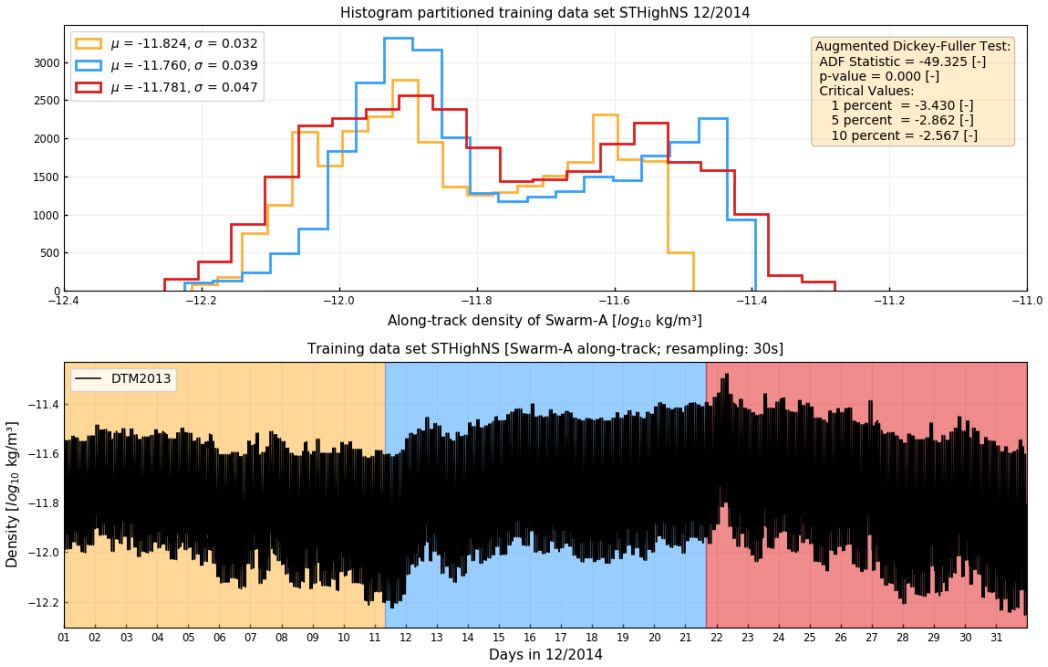


Figure 4.2: Stationarity tests for training data set with identifier STHighNS. Histogram for all three parts of the Swarm-A along-track density values. Additional data presenting the change in statistical properites and ADF-test results (*top*). Density values for December 2014 using the DTM2013 empirical model (*bottom*).

compared to the monthly data sets that do not include significant geomagnetic storms. This indicates that these phenomena are more difficult to predict.

Finally, the different test days are presented in Table 4.2. The test data is chosen to be independent of the training data sets, to guarantee that the NN model has never seen and did never have the opportunity to learn from the same data points. Again, the main difference between the different test data sets is the solar and geomagnetic activity present on each specific test day. In future analyses, cases do exist where an NN model is used to predict values in the past, while the model is trained with future values. As an example, when data set *STHighNS* (12/2014) is used to train an NN model to predict values of *TestHighNS* (11/2014). This approach does not affect the training capabilities of the NN model by any means. However, from an operational point of view this would not be possible since these future data sets would not be available yet. Since all data sets used in this research project are based on past data, this is outside the scope of this study.

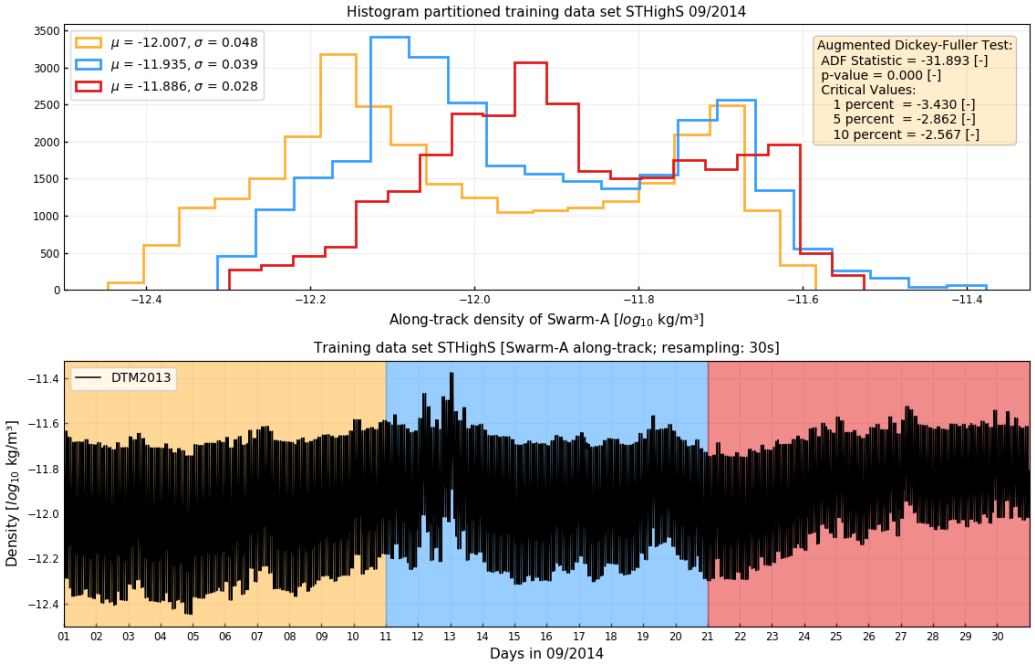


Figure 4.3: Stationarity tests for training data set with identifier STHighS. Histogram for all three parts of the Swarm-A along-track density values. Additional data presenting the change in statistical properties and ADF-test results (*top*). Density values for September 2014 using the DTM2013 empirical model (*bottom*). Geomagnetic storm present at 12/09 and 13/09.

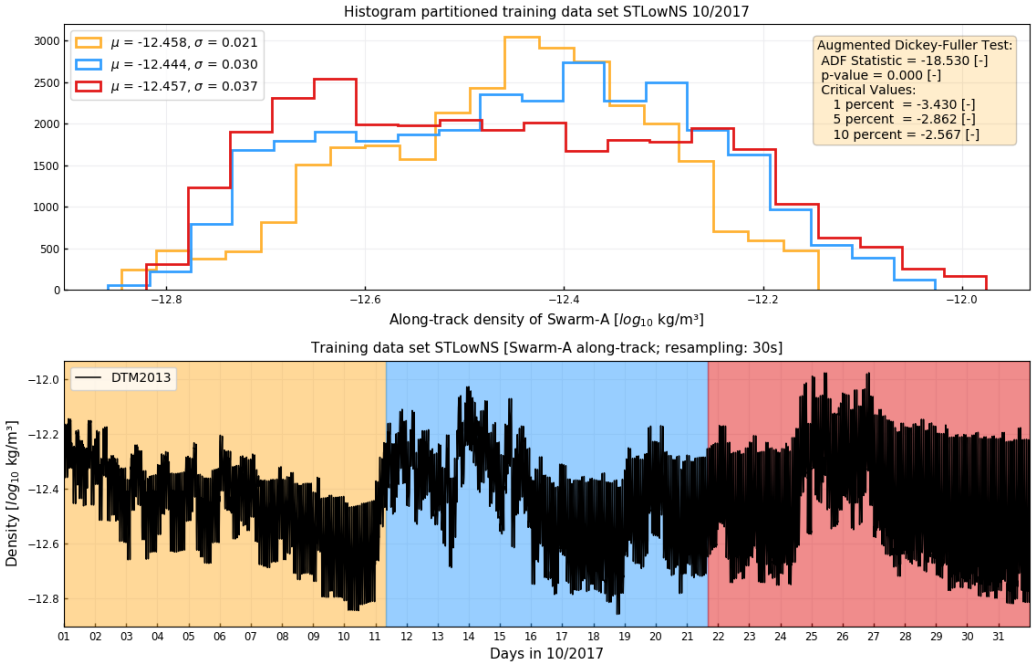


Figure 4.4: Stationarity tests for training data set with identifier STLowNS. Histogram for all three parts of the Swarm-A along-track density values. Additional data presenting the change in statistical properties and ADF-test results (*top*). Density values for October 2017 using the DTM2013 empirical model (*bottom*).

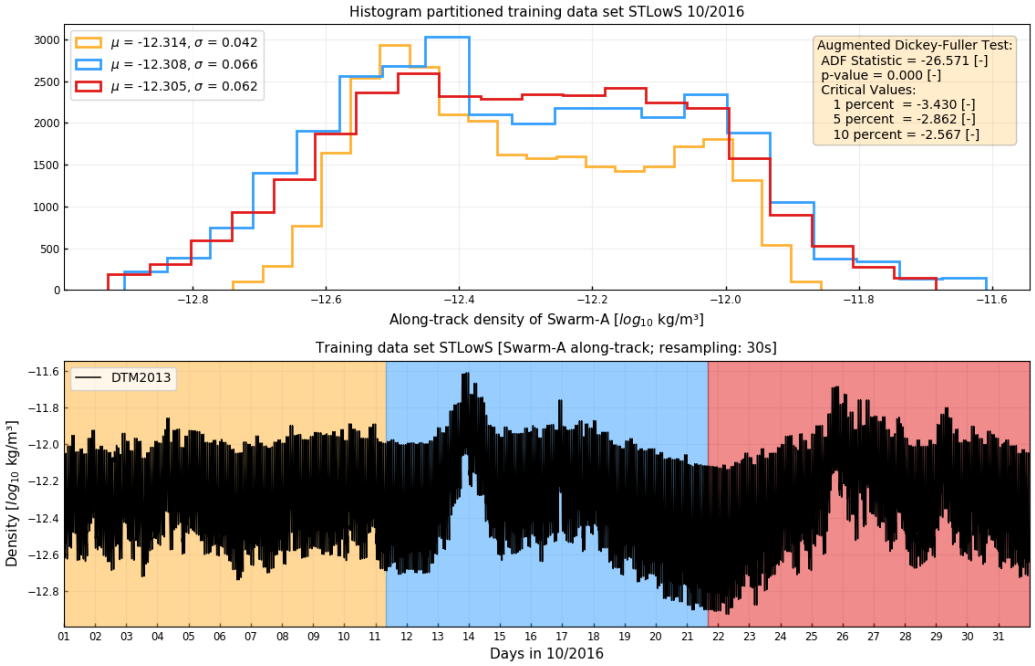


Figure 4.5: Stationarity tests for training data set with identifier STLowS. Histogram for all three parts of the Swarm-A along-track density values. Additional data presenting the change in statistical properties and ADF-test results (*top*). Density values for October 2016 using the DTM2013 empirical model (*bottom*). Geomagnetic storm present at 14/10 and 26/10.

Table 4.2: Test data sets covering a forecasting window of three days. All characterised by a unique solar and geomagnetic activity. (*S* = Storm, *NS* = No Storm).

ID test data set	**TestHighNS**	**TestHighS**	**TestLowNS**	**TestLowS**
Begin data	20/11/2014	19/09/2015	15/05/2016	06/09/2017
End data	22/11/2014	21/09/2015	17/05/2016	08/09/2017
Average $F_{30scaled}$ [sfu]	192.09	112.34	97.83	104.05
Average K_p [-]	2.24	3.03	2.62	3.71
Maximum K_p [-]	3.66	7.0	3.66	8.33
Resampling rate [s]	30	30	30	30
Nb of values	8640	8640	8640	8640
Missing values approach	N.A.	N.A.	N.A.	N.A.
Data preparation approach	log_{10}	log_{10}	log_{10}	log_{10}

Chapter 5

Density Forecasting

This chapter will present all the results that are required to answer the relevant research questions. These results are the output from the models and methodologies that are discussed in the preceding chapters. The different steps to develop the architecture and to tweak the hyperparameters of the final Multilayer Perceptron (MLP) model are discussed in Section 5.1. Section 5.2 discusses the forecasting performance of the best performing MLP models. Finally, Section 5.3 depicts the ability of the model to generalise beyond its given training data in terms of solar and geomagnetic activity, as well as a satellite's orbital elements. In this chapter, the performance metrics are most often listed in a set of tables per test case or model architecture. In those tables, the best performing metrics are indicated in bold, while the worst performing metrics are indicated in italic.

5.1 Model Set-up

This section elaborates on the different hyperparameters that need to be tweaked to reach the best forecasting performance. In Subsection 5.1.1 different model architectures are examined in terms of the number of hidden layers and the number of neurons per layer. Afterwards, the sample period of the training data set is varied in Subsection 5.1.2. Finally, in Subsection 5.1.3, the model's performance is evaluated for a variety of prediction windows and number of delay variables. The best performing MLP model of one subsection is used as the starting model in the next subsection.

In this section, only the STHighNS training data set (see Table 4.1) is used for training purposes. A short term (ST) data set allows to quickly iterate over a set of models with different hyperparameters. A specific set of hyperparameters are kept constant throughout this whole section, as discussed in Subsection 3.4.3. In this section's final subsection, the baseline and MLP models are validated by comparing their performance metrics with those presented by *Pérez et al.* [2014]. In this research paper, the authors focus on predicting the along-track density values for the CHAMP satellite. This reference is one of the few available sources available to validate the MLP models with, despite the dissimilarity in satellite test case.

5.1.1 Model Width and Depth

In this subsection the depth (number of layers) and width (number of neurons per layers) are decided on. More layers and neurons allow the network to capture more nonlinearities that are present in the model-target interaction. The STHighNS training data set and the five-step nested cross-validation method are used in this subsection. The nested cross-validation method is explained in Subsection 3.4.4. For this

nested cross-validation method, the STHighNS training data set is split into five training blocks of respectively 5, 10, 15, 20 and 25 days. This specific problem is sketched in Figure 3.8. The goal of this subsection is to investigate what kind of architectural complexity might affect the forecasting performance of the NN model.

Some hyperparameters such as the sampling period (t_s), number of time-delayed input variables (D) and the prediction window (W_p) are not yet examined and are decided on in the two consecutive subsections. Therefore, a set of first estimates is based on the work performed by *Stastny et al.* [2009] and *Pérez et al.* [2014]. A sampling period of 80 s is used in this subsection since *Stastny et al.* [2009] identified that this period delivered great performance. *Pérez et al.* [2014] claim that the ideal delay of input variables lies between one-fourth and one full orbital period. In their paper, they also conclude that a prediction window of one orbit delivers the ideal performance. Based on these findings, one orbit worth of time-delayed input variables and a prediction window of one orbit are used in this subsection.

To judge which model architectures are considered to perform best, two additional factors are taken into account besides the Root-Mean-Squared Error (RMSE) performance metric. The first one is the time it takes to train a model, while the second one is the ability of a model to take extra input values into account. All three factors are evaluated to finally come up with a set of four best candidate models.

The number of hidden layers is gradually increased such that the MLP models become deeper. In this case the number of hidden layers is interpreted as the number of layers not including the hidden output layer. Within a specific layer, the number of neurons is gradually increased as well. The exact number of neurons per layer is chosen based on the findings of the preceding MLP experiments. Furthermore, each K^{th} step in the nested cross-validation approach is ended if one of the next two conditions is met. Firstly, if the maximum number of 50 epochs is met. This condition has been experimentally identified as the number of epochs the validation loss function needs to converge. Secondly, early stopping is implemented to avoid overfitting when training an MLP model with an iterative method. The training of the model is stopped when it cannot lower the validation loss for more than ten epochs. Finally, seeding is used to feed the pseudo-random number generator with a value such that all randomly generated numbers are identical at epoch one. By doing so, the experiments performed in this subsection can be repeated and checked again. Table 5.1 presents the results for the nested cross-validation experiment for a different number of model architectures.

The performance increase for deeper MLP models is best witnessed when comparing the average RMSE values of the models with one hidden and two hidden layers. Adding one hidden layer to all the best performing one-layer MLP models increases the forecasting performance for almost all models. The only exception here is the 8-6 MLP model. A potential reason could be that this specific number of neurons and the architecture in which they are connected is not able to capture the variation in the thermospheric density well enough. This model is therefore discarded. The forecasting performance remains the same or only slightly decreases when another extra hidden layer is added to the best performing two-layer MLP models. However, these best performing three-layer MLP models are not yet discarded since they still have the potential to capture more complex relations when extra input data is added. The addition of extra input vectors will be discussed in Section 5.3.1.

Only three cases are considered for the 4-layer MLP models since its performance in terms of the average RMSE values degraded significantly in comparison to the models with less hidden layers, while the time to train the model is longer as well. Therefore, the four-layer model is left out of future analysis. A potential reason is that such models contain too many neurons such that the model overfits on the available

Table 5.1: Nested cross-validation test to identify performance capabilities of univariate MLP models with varying number of layers and neurons on training data set STHighNS. Metric used is the RMSE and averaged RMSE for all 5 steps in the nested cross-validation test. (t_s = 80 s, D = 1 orbital period, W_p = 1 orbital period)

# Hidden Layers	**Properties**	**Case 1**	**Case 2**	**Case 3**	**Case 4**	**Case 5**
1 Hidden Layer	# neurons layer 1	1	5	8	10	50
	$RMSE_1$	5.618	0.062	0.071	0.089	0.142
	$RMSE_2$	0.195	0.048	0.045	0.046	0.162
	$RMSE_3$	0.215	0.038	0.050	0.010	0.093
	$RMSE_4$	0.205	0.067	0.091	0.086	0.078
	$RMSE_5$	0.224	0.076	0.069	0.061	0.104
	$RMSE_{av}$	*1.291*	**0.058**	0.065	0.077	0.116
2 Hidden Layers	# neurons layer 1	5	7	8	8	9
	# neurons layer 2	4	5	6	8	8
	$RMSE_1$	0.080	0.061	0.196	0.066	0.067
	$RMSE_2$	0.049	0.045	0.195	0.052	0.048
	$RMSE_3$	0.034	0.037	0.216	0.030	0.066
	$RMSE_4$	0.058	0.059	0.196	0.057	0.055
	$RMSE_5$	0.058	0.066	0.219	0.092	0.059
	$RMSE_{av}$	0.056	**0.054**	*0.204*	0.059	0.059
3 Hidden Layers	# neurons layer 1	5	7	7	8	9
	# neurons layer 2	4	5	5	8	8
	# neurons layer 3	2	3	5	6	6
	$RMSE_1$	0.098	5.593	0.072	0.104	0.111
	$RMSE_2$	0.042	0.195	0.039	0.051	0.039
	$RMSE_3$	0.050	0.215	0.041	0.032	0.038
	$RMSE_4$	0.055	0.205	0.056	0.056	0.055
	$RMSE_5$	0.090	0.224	0.073	0.055	0.055
	$RMSE_{av}$	0.067	*1.286*	**0.056**	0.059	0.060
4 Hidden Layers	# neurons layer 1	7	8	9		
	# neurons layer 2	5	8	8		
	# neurons layer 3	5	6	6		
	# neurons layer 4	3	4	4		
	$RMSE_1$	0.116	0.196	0.197		
	$RMSE_2$	0.039	0.193	0.193		
	$RMSE_3$	0.033	0.231	0.216		
	$RMSE_4$	0.057	0.208	0.207		
	$RMSE_5$	0.058	0.222	0.224		
	$RMSE_{av}$	**0.061**	*0.210*	0.208		

training data sets. This problem of overfitting during the training phase is also present when increasing the number of neurons per layer. This is best seen in Case 5 of the MLP model with one hidden layer. Here it is concluded that if a hidden layer has too many neurons, in this case from ten neurons onward, the accuracy worsens with 100% when compared to the best performing MLP models.

It is also interesting to take a look at the RMSE values that are computed for each step in the nested cross-validation method. The RMSE values are computed on the final five days of the STHighNS data set after the model is being trained on the first 5, 10, 15, 20 and 25 days of this STHighNS data set. The RMSE values are indicated as $RMSE_1$, $RMSE_2$, $RMSE_3$, $RMSE_4$ and $RMSE_5$, respectively. From the results in Table 5.1 it can be concluded that the two-layer MLP models perform best when trained on a data set of

five days, while the three-layer MLP models generally perform best when trained on data sets that span more than five days. This indicates that the three-layer MLP models have the advantage to learn more relations between the input and output variables when longer training data sets are available. *Domingos* [2012] explicitly states that adding data is one of the best ways to develop a more robust neural network. Therefore, monthly data sets are used as the short term training data sets as discussed in Chapter 4.

In conclusion, the four best performing two-layer MLP models (5-4 and 7-5) and three-layer MLP (7-5-5 and 8-8-6) models are used as a starting point in the next subsection. One still wants to keep the options open in terms of MLP model architecture since picking one specific model does not guarantee that the model is going to perform the best when exposed to training data sets with different solar and geomagnetic activity.

5.1.2 Sampling Period

The goal of this subsection is to identify the model that is most robust to a variation in sampling period and to pick a sampling period for which this model performs best. The four best performing two-layer (5-4 and 7-5) and three-layer (7-5-5 and 8-8-6) MLP models of the previous subsection are used as a starting point. The sampling period, t_s, is varied from 30 s until 300 s to test the robustness of the models. *Pérez et al.* [2014] already concluded that forecasting accuracy diminished for sampling periods longer than 180 s. However, a sensitivity analysis is still recommended given they implemented a neural network model with a different model architecture. Similar to Subsection 5.1.1, the number of time-delayed input variables (D) and the prediction window (W_p) are kept constant throughout this subsection and are set equal to one orbital period. The nested cross-validation method is used on the STHighNS data set (see Table 4.1) to evaluate the relative forecasting performance of the set of MLP models. The same early stopping procedure is used as is discussed in the previous subsection. Another seeding variable is set at the start of this experiment, such that all results presented in this subsection can be compared. The averaged RMSE values of the nested cross-validation experiment are presented in Table 5.2.

The best performing MLP model is the model which has the lowest averaged RMSE value overall and does perform well when other sampling periods are implemented. Beyond 180 s the forecasting performance significantly decreases for all four models. However, within the boundaries of 30 s to 120 s the performance varies irregularly for all four models. All four models also perform best on different sampling periods. Solely based on the lowest RMSE values, one should go for the three-layer 8-8-6 MLP model at a sampling period of 120 s. Two other consequences of having a larger sampling period other then the original 30 s need to be considered as well. Firstly, the disadvantage of having a larger sampling period is that the smaller variations in the density signal could disappear. However, from an operational point of view, it is not required to achieve a density forecasting resolution every 30 s. Secondly, the training data sets are fixed between a start and end date. Consequently, the number of data points to train the model with decreases when the sampling period is increased. On the other hand, this has as advantage that the time it takes to train the model reduces. In conclusion, the preferred sampling period is chosen to be 120 s. The 8-8-6 MLP model will be used in all coming model developments and performance tests. If in future research studies a smaller sampling period is required, the other two-layer MLP models are good candidates as well since they require less time to train.

Table 5.2: Nested cross-validation test to identify the performance capabilities of predefined two- and three-layer MLP models with varying sampling period (t_s) on training data set STHighNS. Metric used is the RMSE averaged over all 5 steps in the nested cross-validation test. (D = 1 orbital period, W_p = 1 orbital period)

Sampling Period t_s	**MLP 5-4**	**MLP 7-5**	**MLP 7-5-5**	**MLP 8-8-6**
30 s	0.063	0.068	0.076	0.058
50 s	**0.056**	0.062	0.060	0.064
80 s	0.057	**0.059**	**0.056**	0.067
100 s	0.061	0.067	0.208	0.061
120 s	0.060	0.069	0.206	**0.055**
180 s	0.071	0.068	0.070	0.077
300 s	*0.102*	*0.119*	*2.093*	*0.106*

5.1.3 Input Delays and Prediction Window

The 8-8-6 MLP model is identified as the MLP model with the most ideal architecture given the current drag problem, based on the findings discussed in the previous two subsections. This model in combination with a sampling period of 120 s is used to figure out the best relation between the size of the input vector (D) and the prediction window (W_p). One would expect the number of delay variables to increase when the prediction window increases since the relation between in the input variables and the values to predict become less apparent the further they are apart. The relation between D and W_p is examined in this subsection. The nested cross-validation method is not used, since the relative performance of different model architectures does not need to be compared. The 8-8-6 MLP model is trained on the STHighNS data set (see Table 4.1) and tested on the TestHighNS test data set (see Table 4.2). Every forecast starts at the begin date of the test data set, as noted in Table 4.2.

A discrete set of variables is chosen for the variation of D and W_p. This discretisation is based on the orbital period of the Swarm-A satellite which is assumed to perform 15 orbital periods in one day. The number of delay variables is varied from one to 45 orbital periods worth of data points. The prediction window is varied from one to 15 orbital periods worth of data. A well-performing MLP model that can predict the along-track density values over a period of one day is generally desired since this is the time it takes for new solar indices to become available. Forecasts over periods longer than one day are discussed in Section 5.2. Table 5.3 summarises the quantitative results of this experiment. The performance metrics which are used in this table are the performance metrics discussed in Section 3.5.

For comparison, the performance metrics of two additional baseline models are added to Table 5.3 as well. The first baseline model is the Calibrated Persistence Model (CPM) which is described in Subsection 3.3.2. Its performance is listed for every W_p. To transform the regular Persistence Model (PM) to the CPM a set of three transformation parameters are required. All three transformation parameters are listed in the table as well. The scale factor, a, and the bias, b, are the variables trained on the training data set, while the horizontal displacement, h, is determined by the specific prediction window and test data set. Secondly, the DTM2013 model values are added for every variation in prediction window size. Note that, these DTM2013 predictions are in fact the modelled density values given the known and observed input values (orbital state and solar and geomagnetic indices) on that given point in time. This thus means that those input values are not forecasted by any means. Next to these two baseline models the PM model developed by *Pérez et al.* [2014] is added to validate the outcome of the 8-8-6 MLP model. Their model is tested on the CHAMP satellite at times with high solar and low geomagnetic activity (D_{st} = -16 nT, a_p = 10 and $F_{10.7}$ = 190.4 sfu) with a prediction window of both half and one orbital period and

Table 5.3: Univariate 8-8-6 MLP model performance analysis for different sets of number of time delays (D) and prediction windows (W_p). Trained on the STHighNS training data set and tested on the TestHighNS data set with a sampling period (t_s) of 120 s. The PM, CPM and DTM2013 serve as baseline models to compare the MLP's performance with. (PM = Persistence Model; CPM = Calibrated Persistence Model with a = 0.94 and b = -0.82)

Prediction Window, W_p	**Delay Variables,** D	**RMSE**	**B**	**R**	**Settings**
1 orbit	CPM	0.106	1.007	0.923	h=5640s
	Persistence model	(0.152)	(1.000)	(0.969)	[*Pérez et al.*, 2014]
	DTM2013	0.182	1.015	0.945	
	1/2 orbit	*0.238*	*1.019*	*0.894*	
		(0.118)	(1.001)	(0.982)	[*Pérez et al.*, 2014]
	1 orbit	0.134	1.010	0.978	
		(0.122)	(1.001)	(0.981)	[*Pérez et al.*, 2014]
	5 orbits	0.098	1.007	**0.983**	
	10 orbits	0.103	1.007	0.979	
	15 orbits (1 day)	0.130	1.010	0.977	
	30 orbits (2 days)	**0.067**	**0.999**	0.973	
	45 orbits (3 days)	0.084	0.996	0.969	
5 orbits	CPM	0.065	1.004	0.939	h=5040s
	DTM2013	*0.187*	*1.015*	0.927	
	1/2 orbit	0.178	1.013	*0.697*	
	1 orbit	0.071	1.005	0.956	
	5 orbits	0.076	1.006	**0.967**	
	10 orbits	0.096	1.007	0.961	
	15 orbits (1 day)	**0.059**	1.004	0.963	
	30 orbits (2 days)	0.165	1.013	0.959	
	45 orbits (3 days)	0.062	**1.003**	0.954	
15 orbits	CPM	0.061	1.003	0.899	h=3840s
	DTM2013	*0.179*	*1.015*	0.897	
	1/2 orbit	0.131	1.006	*0.455*	
	1 orbit	0.061	1.002	0.898	
	5 orbits	**0.050**	**0.999**	0.923	
	10 orbits	0.071	0.996	**0.925**	
	15 orbits (1 day)	0.059	1.002	0.920	
	30 orbits (2 days)	0.074	1.004	0.924	
	45 orbits (3 days)	0.118	1.009	0.900	

a sampling period of 80 s. The PM of *Pérez et al.* [2014] is shifted, but no bias or scale factor are added to the shifted forecast. This explains why the CPM has better performance in terms of RMSE values than the PM of *Pérez et al.* [2014].

The RMSE values in Table 5.3 indicate that the number of delay variables of the input vector significantly determines the MLP's performance. The performance for all three prediction window variables is the worst when the number of variables in the input vector drops below one orbital period. This can be explained by taking a look at the schematic of the architecture of the NN model in Figure 3.6. Every variable in the input vector is fully-connected to the first hidden layer with eight neurons. This means that if too few input variables are used as input the model is underfitted and as consequence lacks forecasting performance. Given a specific prediction window and a number of delay variables higher than one orbital period, the model's performance seems to be insensitive to the variation in the number of delay variables. More testing is required to find out what number of delay variables needs to be chosen for future anal-

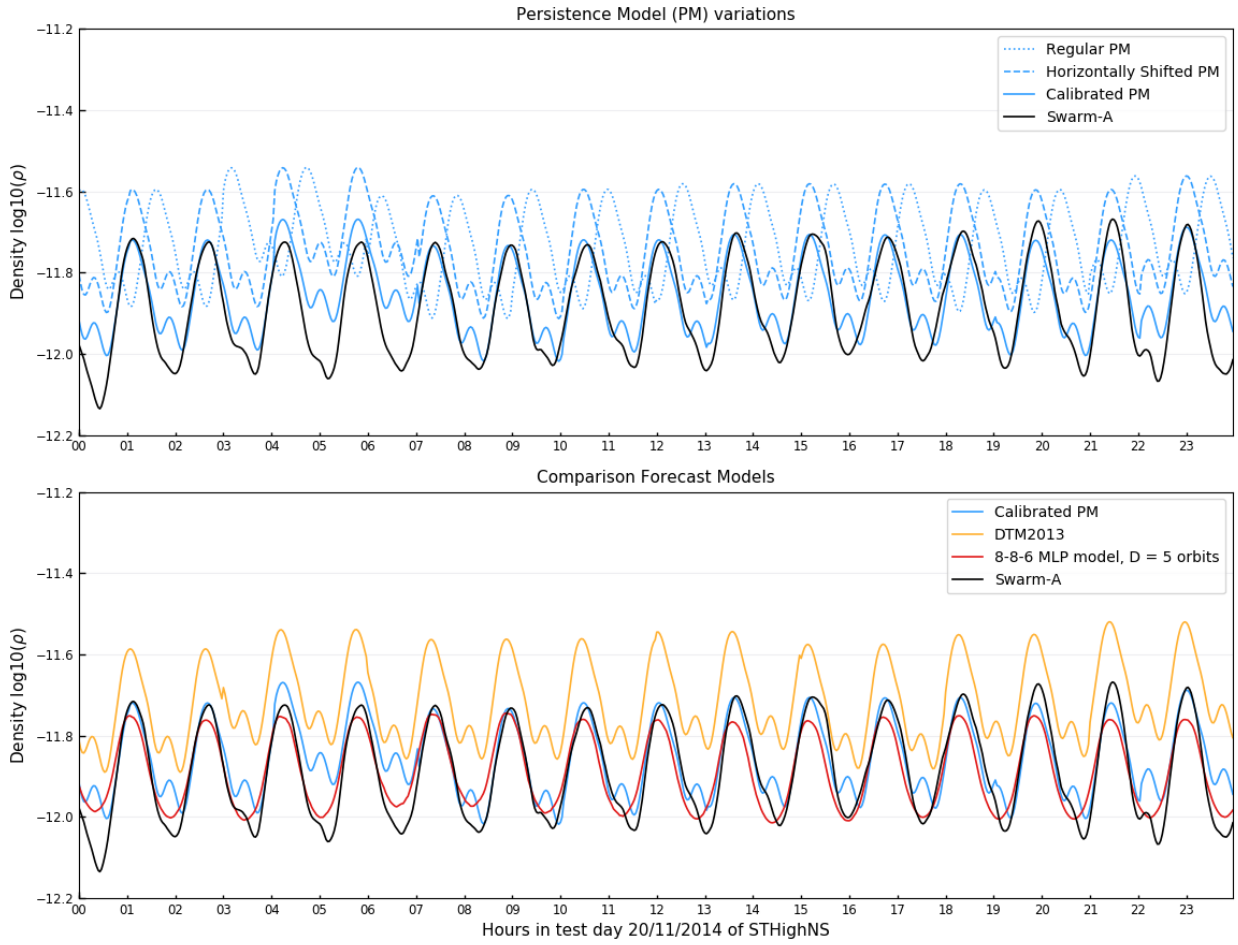


Figure 5.1: Evolution of Persistence Model (PM) performance (*top*) and comparison of the Calibrated Persistence Model (CPM), DTM2013 and 8-8-6 MLP model with the POD-derived density values (*bottom*). Trained on the STHighNS training data set and tested on the first day of the TestHighNS data set. (MLP with t_s = 120 s, D = 5 orbital periods and W_p = 15 orbital periods; CPM with a = 0.94, b = -0.82 and h = 3840 s)

ysis. For each specific prediction window, the best performing 8-8-6 MLP model is able to outperform both the CPM and the DTM2013. This indicates that the MLP models do have the ability to capture the complex relations between the input and output variables. In cases where the CPM outperforms the MLP models this is mainly caused by the MLP models which are subjected to overfitting or underfitting. The MLP models are superior when more delayed variables are used from which it can learn more complex and rare events. The CPM is not able to take such events properly into account and would e.g. only forecast the occurrence of a geomagnetic storm on a given day, if this same storm happened the day before as well, which is highly unlikely. The reason why *Pérez et al.* [2014] reaches optimal performance for delay variables lower than one orbital period is because of the single neuron in the first hidden layer of their neural network. Having only one neuron avoids the risk of overfitting your network when only a little number of delay variables is introduced as input (eg. half or one orbital period). However, their model loses the ability to perform well when the number of delay variables is increased.

Figure 5.1 graphically presents the time series behind the performance metrics in Table 5.3. The *top* graph compares the different type of persistence models to the POD-derived density values. This plot

serves as an illustration of what the effect is of horizontally shifting and calibrating the regular PM model. The key takeaway here is that on this test day the forecast performance increases from the regular PM to the horizontally shifted PM to then finally the Calibrated PM (CPM). The *bottom* graph presents both the POD-derived density values, as well as the one-day forecasts of the CPM, the DTM2013 and the 8-8-6 MLP model with five orbital periods worth of delay variables which is the best performing model in Table 5.2. The MLP model is able to smooth the little jumps which occur occasionally in the DTM2013 data at for example 03:00 am and 15:00 pm. These little jumps are the result of the three-hourly changing geomagnetic index, K_p. However, the MLP model is not able to capture the small non-linearities that are present in both the DTM2013 and POD-derived density values. On this test day, the CPM and MLP model do perform better in predicting the POD-derived density values in comparison with the DTM2013. Note that these DTM2013 values are the modelled thermospheric density values as if all input values (orbital state and solar and geomagnetic indices) would known beforehand. Despite this assumption of having perfectly forecasted values, the DTM2013 predicts higher densities than the actual target densities. This is in agreement with its bias values that are larger than one in Table 5.2. In this case, the best performing MLP models and CPM are the better options.

The results in Table 5.3 and Figure 5.1 do hold for predictions for one specific day into the future. Furthermore, it was concluded that the ideal number of delay variables is highly dependent on the prediction window and the test day. For this reason, it would be interesting to examine how the 8-8-6 MLP model performs on a set of different test days. The daily forecast performance of the 8-8-6 MLP is therefore tested for all days of the month of September, 2015. Figure 5.2 presents the hourly-averaged geomagnetic (*upper two*) and solar (*third*) indices of that month, as well as the POD-derived density values (*bottom*) to give some more context to this experiment. The 8-8-6 MLP model is trained on the STHighNS training data set (see Table 4.1) which has different solar and geomagnetic activity in comparison to September 2015. The monthly-averaged solar activity of the STHighNS training data set is around 70 sfu higher than in September 2015. On dates 07/09/2015 and 09/09/2015 a *moderate storm* occurs and on dates 11/09/2015 and 20/09/2015 a *strong storm* occurs, while no storm was present in the training data set. This difference in space weather indicates that the MLP model will lack forecasting performance at moments of high geomagnetic activity. A more in-depth analysis of the impact of solar and geomagnetic activity will be discussed in Section 5.3.

Figure 5.3 presents the daily RMSE values of the CPM and the 8-8-6 MLP model for a set of different delay variables for the aforementioned month. Delay variables that fall within one orbital period or less are discarded, as has been concluded earlier in this section. Delay variables within five, ten and fifteen orbital periods are compared since they show the best performance in terms of RMSE values for a prediction window of 15 orbital periods, see Table 5.3. A couple of interesting findings can be observed in Figure 5.3. First, the four models perform the worst around the dates 10/09/2015, 12/09/2015 and 21/09/2015. This can be declared to the aforementioned difference in geomagnetic activity in the training and test data sets. What is interesting to see is that these peaks in RMSE values occur one day after the extreme values of K_p and D_{st} in Figure 5.2. For the CPM this can be best explained by the fact that the density values of the past day are used as forecasts for the next day. For the MLP models, this can be explained by the input vector of the model which is one day apart from the time it forecasts the next density value. The univariate MLP models therefore miss out on the information that a geomagnetic storm will take place. Second, overall the 8-8-6 MLP models do perform better than the CPM model. This is a promising insight which means that MLP models can outperform the baseline models. Finally, the MLP model with ten orbits worth of delay variables performs overall the best of all three 8-8-6 MLP models.

Based on the results in Figure 5.3 and the findings of the previous paragraph, it is concluded that the

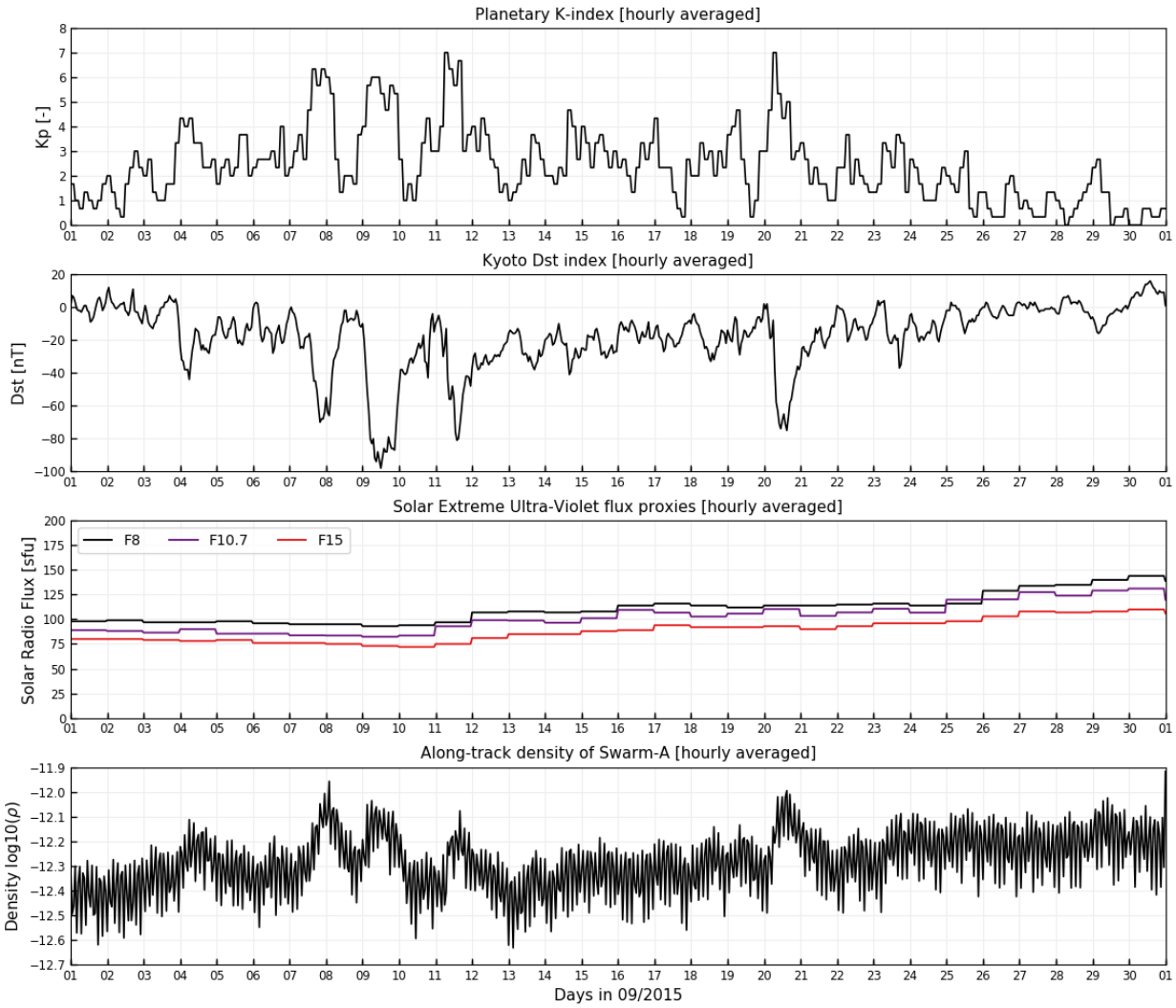


Figure 5.2: Overview of planetary K-index (*top*), D_{st} index (*second*) and solar EUV proxies (*third*). Swarm-A along-track POD-derived density values for September, 2015 (*bottom*).

8-8-6 MLP model with a prediction window of 15 orbital periods and ten orbital periods worth of delay variables is the best performing univariate MLP model of this section. Nevertheless, the CPM is a good approach if short processing times and a decent forecasting performance are required. The ideal number of delay variables for univariate MLP models with longer prediction windows or multivariate MLP models will be examined in Section 5.2 and Section 5.3, respectively.

5.2 Forecasting Performance

The appropriate structure of the univariate MLP model was defined in Section 5.1. This section has the goal to assess the forecasting performance of both the baseline models and the 8-8-6 MLP model when the prediction window size is enlarged or a larger training data set is used. Subsection 5.2.1 will elaborate on this variation in size of the prediction window to check if the neural network approach can achieve superior forecasting accuracy in comparison to the other baseline models. Finally, Subsection 5.2.2 will evaluate the performance when additional training data is used to train the MLP model.

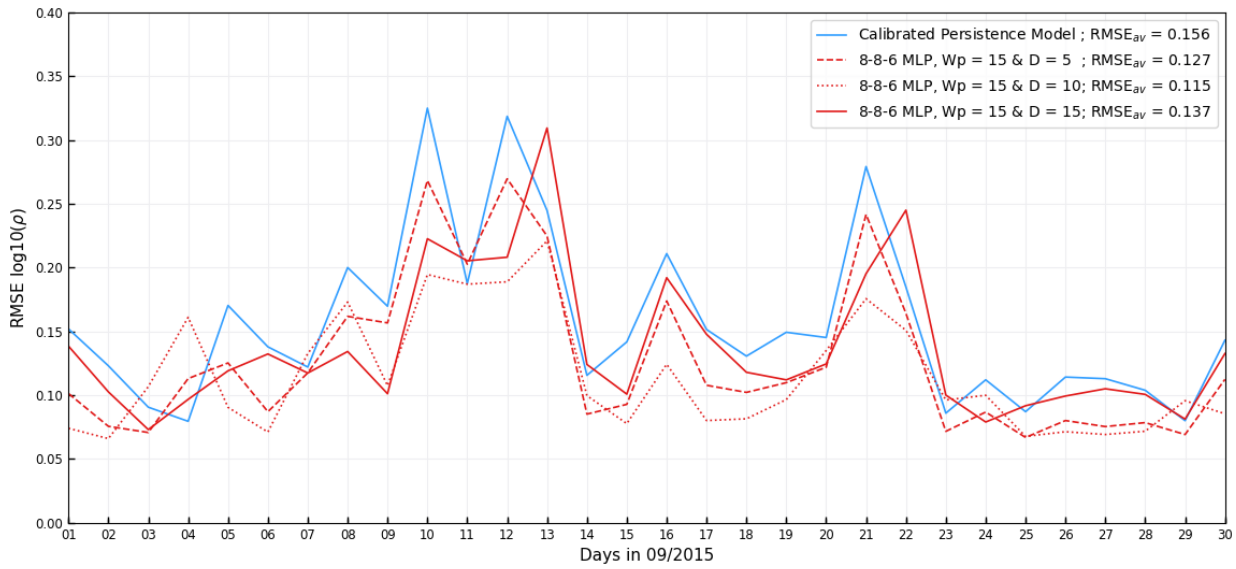


Figure 5.3: Daily computation of Root Mean Square Error values for the CPM and set of 8-8-6 MLP models trained on STHighNS and compared to POD-derived density values in September, 2015. Statistical significance test to check overall performance of CPM and 8-8-6 MLP models with t_s = 120 s, W_p = 15 orbital periods and varying D values.

5.2.1 Larger Prediction Window

Subsection 5.1.3 was all about forecasting the along-track density values up until 15 orbital periods (eg. one day) into the future. In this subsection, the size of the prediction window is changed to 30 and 45 orbital periods to see how the univariate 8-8-6 MLP model performs when it needs to forecast density values up to three days into the future (as defined by the research questions in Section 1.2). This would mean that the MLP models need to be able to learn the complex relations between the input and target values which are more than one day apart.

Table 5.4 presents the performance metrics of the CPM, DTM2013 and 8-8-6 MLP models for prediction window sizes of both 30 and 45 orbital periods into the future. The CPM and MLP models are trained on the STHighNS training data set and tested on the TestHighNS test data set for a sampling period of 120 s, similar to the models in Subsection 5.1.3. The set of delay variables (D) is similar to the ones listed in Table 5.3. In both cases, the performance of the MLP models seems rather insensitive to the variation in the number of delay variables. Therefore, the best performing MLP models are chosen to test their performance over a period of one month. Table 5.4 indicates that these MLP models are the ones when 30 orbital periods worth of delay variables are used in the input vector. These two MLP models are used to generate the RMSE values of the daily forecasts for all 30 days in September 2015. Figure 5.4 presents the results of this experiment and also includes the forecast performance of the CPM as a baseline to compare the performance of the MLP models with.

The results in Table 5.4 indicate that the 8-8-6 MLP models still perform better than the DTM2013 predictions. This is not unexpected since the TestHighNS test data set did not change in comparison to the results presented in Subsection 5.1.3. From the daily RMSE values in Figure 5.4 one can conclude that the performance of the 8-8-6 MLP models decreases when the size of the prediction window increases. This concludes that the MLP models are less effective in learning the relationship between its input and

Table 5.4: Univariate 8-8-6 MLP model performance analysis for different sets of number of time delays (D) and long term prediction windows (W_p). Trained on the STHighNS training data set and tested on the TestHighNS data set with a sampling period (t_s) of 120 s. The CPM and DTM2013 serve as baseline models to compare the MLP's performance with. (CPM = Calibrated Persistence Model with a = 0.94 and b = -0.82)

Prediction Window, W_p	**Delay Variables, D**	**RMSE**	**B**	**R**	**CPM settings**
30 orbits	CPM	0.067	1.004	0.912	h=1920s
	DTM2013	*0.189*	*1.016*	*0.894*	
	1/2 orbit	0.067	**1.003**	0.898	
	1 orbit	0.091	1.006	0.902	
	5 orbits	0.117	1.009	0.925	
	10 orbits	0.076	1.005	**0.930**	
	15 orbits (1 day)	0.107	1.007	0.926	
	30 orbits (2 days)	**0.055**	1.004	0.921	
	45 orbits (3 days)	0.124	1.009	0.911	
45 orbits	CPM	0.074	1.004	0.900	h=5640s
	DTM2013	*0.184*	*1.015*	**0.912**	
	1/2 orbit	0.086	1.005	*0.850*	
	1 orbit	0.110	1.008	0.895	
	5 orbits	0.086	1.005	0.895	
	10 orbits	0.165	1.013	0.901	
	15 orbits (1 day)	0.081	1.005	0.901	
	30 orbits (2 days)	**0.062**	**1.000**	0.905	
	45 orbits (3 days)	0.139	1.010	0.896	

target values if the distance between both increases beyond one day. The MLP model appears to hold up against the CPM if forecast windows are smaller or equal to two days. However, for forecasts of three days into the future an MLP model does overall perform worse than the baseline CPM. For future analysis, only the univariate 8-8-6 MLP model with a prediction window of 15 orbital periods and a number of delay variables worth ten orbital periods is used.

5.2.2 Additional Training Data

In this subsection, the effect of increasing the size of the training and validation data set is evaluated in terms of forecasting performance. Based on the results in Figure 5.3 it was concluded that the models are not able to capture sudden variations in solar or geomagnetic activity. This is not strange since the model never encountered such behaviour in its original training data set (eg. STHighNS). For this reason, the size of the training data set is increased to a maximum of four months worth of training data. The LTHigh data set (see Table 4.1) is used for this experiment and overlaps with both the STHighNS and STHighS data sets. The LTHigh training data set includes a multitude of days on which a geomagnetic storm took place, as well as an overall variation in solar activity. Table C.1 states that a set of data points are flagged in the LTHigh training data set. As a consequence, the missing data points are linearly interpolated as discussed in Section 4.1. A maximum of four months is considered long term (LT) in this research study due to the presence of too large data gaps in the rest of the monthly Swarm data. The development of more sophisticated interpolation techniques is beyond the scope of this report. In Figure 5.5 the daily RMSE values are plotted for the baseline CPM and two 8-8-6 MLP models for September 2015. These two MLP models have a prediction window of one day and ten orbits worth of delay variables. One MLP model is trained on the STHighNS training data set, while the other MLP model is trained on the LTHigh training data set. Both training data sets are resampled at 120 s.

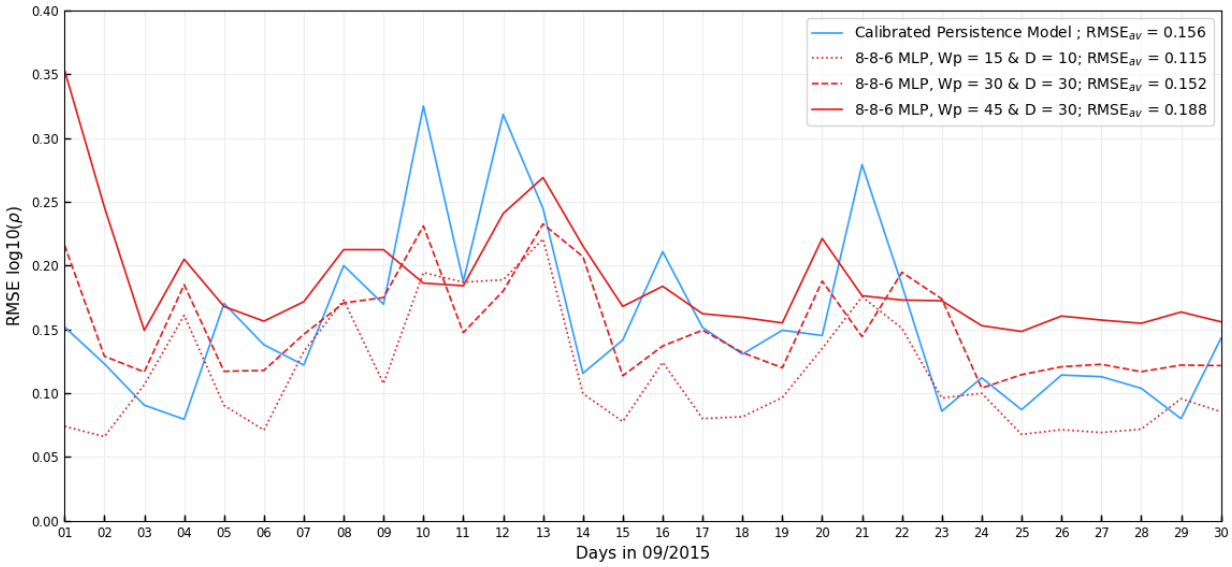


Figure 5.4: Daily computation of Root Mean Square Error values for the CPM and set of 8-8-6 MLP models trained on STHighNS and compared to POD-derived density values in September, 2015. Statistical significance test to check overall performance of CPM and 8-8-6 MLP models with t_s = 120 s and varying W_p and D values.

Based on the results in Figure 5.5 it is concluded that increasing the size of the training and validation data sets does not guarantee an increase in forecasting performance. Despite this conclusion being unexpected, this does not necessarily mean that adding data to a training data set is always harmful to the forecasting performance in this drag problem. A possible reason could be that the linear interpolation methods, which used to fill the data gaps that are present in the training data set, add bias to the trained model. Overall, the MLP model which is trained on the LTHigh training data set still has similar performance as the CPM.

To validate these results, one can compare the aforementioned findings with the approach that is presented in the research study of *Pérez et al.* [2014]. In this research study, they compared the performance of the same MLP model trained with two different training data set sizes. One using 365 days for training, while the other models are trained on one day worth of data. They concluded that overall the model with the larger training data set performed better than the one trained on the smaller training data set. However, the remark shall be made that the difference in size of their training data sets is much more significant in comparison to the difference in size of LTHigh and STHighNS. Unfortunately, it is not possible to increase the size of the LThigh data set any further. This is considered to be beyond the scope of this report. In conclusion, the short term training data sets are used for further analyses since they showed to perform best on the test case and do require less computational resources to train an MLP model.

5.3 Generalisation Beyond Training Data Set

This section has the goal to test the ability of an MLP model to generalise beyond the properties of its training data sets in terms of solar and geomagnetic activity, as well as orbital elements. First, multivariate MLP models are examined in Subsection 5.3.1. These models are then tested in different solar and geomagnetic conditions in Subsection 5.3.2. Finally, the performance of these models is tested on the other two Swarm satellites in Subsection 5.3.3.

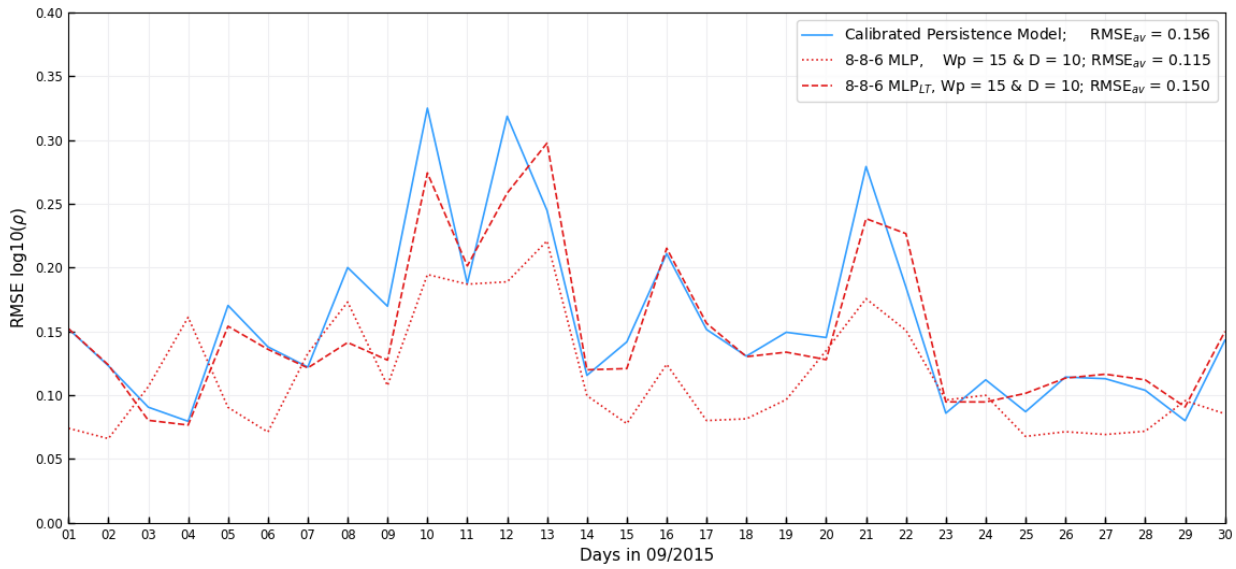


Figure 5.5: Daily computation of Root Mean Square Error values for the CPM and two 8-8-6 MLP models trained on STHighNS and LTHigh, and compared to POD-derived density values in September, 2015. Statistical significance test to check overall performance of CPM and 8-8-6 MLP models with t_s = 120 s, W_p = 15 orbital periods and D = 10 orbital periods.

5.3.1 Multivariate MLP Model

In this subsection, the univariate MLP model is extended with additional solar or geomagnetic indices to test if adding this extra information could potentially help the MLP with learning the complex relations that are embedded in the solar space weather. The theory of the multivariate MLP model has been discussed in Subsection 3.4.3. This multivariate MLP model adds one extra layer to the univariate model to accommodate two input time series. On the one hand, the DTM2013 density values while on the other hand, an additional solar or geomagnetic index. Each of these input vectors is connected to the 8-8-6 MLP model by an additional hidden layer with nine neurons. This transforms the univariate 8-8-6 MLP model to the multivariate 99-8-8-6 MLP model. In terms of solar indices, the $F_{3.2}$, F_8, $F_{10.7}$ and F_{15} indices are added to the MLP model. The F_{30} index is not considered to be a potential candidate since this index is already used as input for the DTM2013. In terms of geomagnetic index, only the Kyoto D_{st} index is used as additional input for the multivariate MLP model. The K_p index is not considered since this index is also already used as input for the DTM2013.

The multivariate 99-8-8-6 MLP model is tested separately for each additional solar and geomagnetic index. Every model is going to be trained on the STHighNS training data set with a sampling period of 120 s and a prediction window of one day. The number of epochs at which the training phase is stopped is defined by a maximum number of 50 epochs or the moment the validation loss has converged for a number of iterations. In contrast to the univariate models, the number of delay variables is increased from ten to 15 orbital periods worth of input data. This larger input vector is needed to guarantee that the MLP models get the opportunity to learn from an actual variation in solar and geomagnetic indices. 15 orbital periods are chosen since this is the time it takes for the daily solar indices to become available. All models are then tested by performing daily forecasts for every day in September 2015. Afterwards, their daily RMSE values are plotted for all 30 days of this month. The results for the 99-8-8-6 MLP models with additional solar indices are presented in Figure 5.6. The result for the multivariate MLP model with the

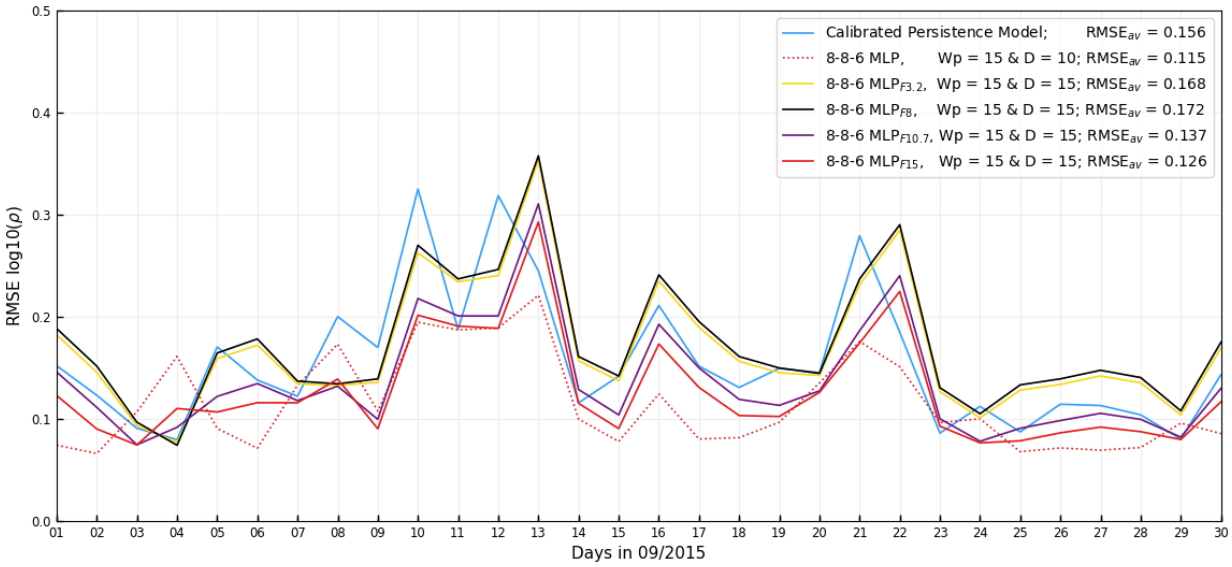


Figure 5.6: Daily computation of Root Mean Square Error values for the CPM and MLP models trained on STHighNS and compared to POD-derived density values in September, 2015. Statistical significance test to check overall performance of the CPM, univariate 8-8-6 MLP model and multivariate 99-8-8-6 MLP models with additional solar indices. MLP models with t_s = 120 s and W_p = 15 orbital periods with varying D.

added D_{st} index is presented in Figure 5.7. In both figures, the multivariate MLP models are compared to the baseline CPM and best-performing univariate MLP model of Section 5.1.

Figure 5.6 compares the performance of the 99-8-8-6 MLP model with extra solar indices. The figure indicates that no significant improvements are realised when extra information on the solar activity is added. In the best case, such a multivariate model does achieve similar or slightly better performance when compared to the CPM. A possible explanation is that a maximum of only two different solar indices can be encapsulated in the input vector. This might not add enough new information to the network to learn from. On the other hand, adding more delay variables has the disadvantage that the MLP model overfits, as discussed in Subsection 5.1.3. A solution could be to use different types of neural networks that include delay mechanisms which allow for more past information to be included. Given the current model architecture, a similar effect could be achieved when the sampling period is increased for the input vector that contains the solar indices. Nevertheless, the 15 cm and 10.7 cm solar radio fluxes are considered as the best candidates. Overall, the addition of the F_{15} index resulted in the best performing multivariate MLP model. A potential explanation is that this solar index has the strongest linear correlation with the POD-derived density values, as presented in Table 3.2. In this section, this multivariate MLP model is only tested and trained on limited data sets with specific solar and geomagnetic activity. Therefore, the 99-8-8-6 MLP model with added F_{15} index will be used in Subsection 5.3.2 to further test its performance using a set of different test data sets.

Figure 5.7 compares the baseline CPM and best performing univariate model with the multivariate MLP model with additional geomagnetic D_{st}. After having added the D_{st} index the model was hard to train due to the rapidly changing hourly D_{st} values. This results in large peaks of high forecasting errors when low geomagnetic activity is present. This shows that the model is not able to map the relation between its input and target values. However, during periods of high geomagnetic activity the model is able to outperform the best performing 8-8-6 MLP model, which has been considered the best performing model

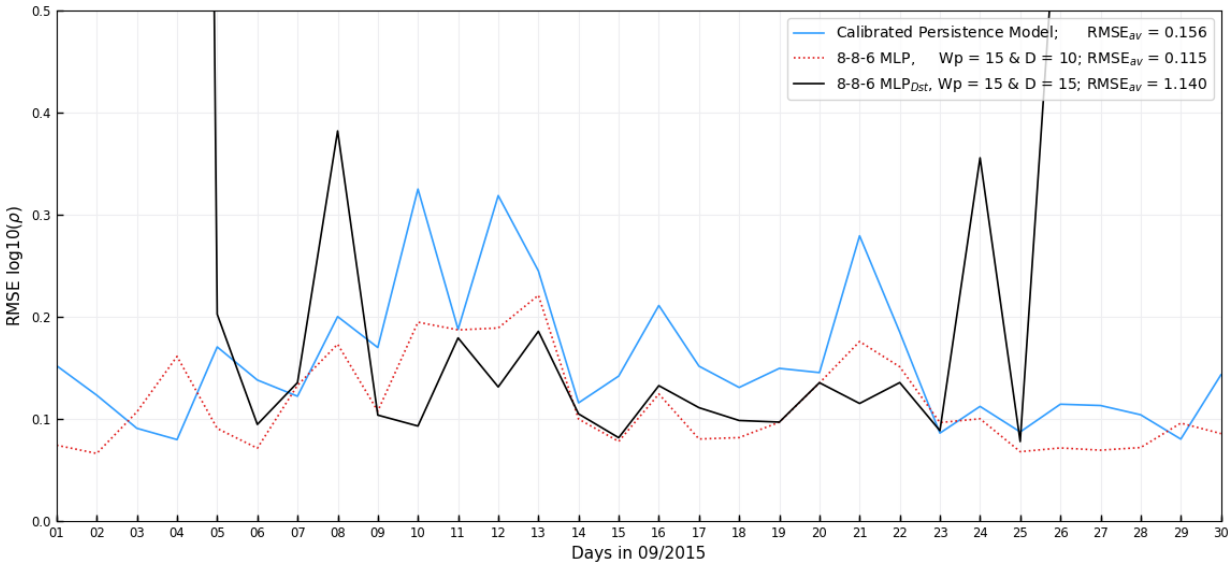


Figure 5.7: Daily computation of Root Mean Square Error values for the CPM and MLP models trained on STHighNS and compared to POD-derived density values in September, 2015. Statistical significance test to check overall performance of the CPM, univariate 8-8-6 MLP model and multivariate 99-8-8-6 MLP models with the additional geomagnetic index, D_{st}. MLP models with t_s = 120 s and W_p = 15 orbital periods with varying D.

so far. For this reason, the 99-8-8-6 MLP model with additional geomagnetic index is kept for future analyses, where every model will be trained and tested on a combination of different data sets.

5.3.2 Variability in Solar and Geomagnetic Activity

This subsection will test the forecasting performance of the baseline CPM and DTM2013, as well as the best performing univariate MLP model of Subsection 5.1.3 and multivariate MLP model of Subsection 5.3.1. The CPM and all MLP models are going to be trained on the STHighNS, STHighS, STLowNS and STLowS training data sets with a sampling period of 120 s and prediction window of one day. Afterwards, the forecasting performance of every trained model is examined by testing every model on all four test data sets, namely TestHighNS, TestHighS, TestLowNS and TestLowS. A forecast is performed for all three days in the test data sets. This allows to test if an MLP model has the ability to outperform the baseline models when the solar and geomagnetic conditions of the training and test data sets are different. The training and test data sets are presented in Table 4.1 and Table 4.2, respectively. The performance metrics, as discussed in Section 3.5, are used to assess the forecasting capabilities. Table 5.5 lists all the performance metrics for the models trained on the data sets that are characterised by high solar activity, while Table 5.6 lists the results for the data sets during periods with low solar activity. A set of conclusions are drawn from these performance metrics and are listed below.

The performance of the baseline models (eg. DTM2013 and CPM) are discussed next. From the results in Table 5.5 and Table 5.6 one can conclude that the DTM2013 actually outperforms both the CPM and MLP models when tested on periods of low solar activity, irrespective from the levels of geomagnetic activity. This can be explained by the research study presented by *Bruinsma* [2015], who concluded that the DTM2013 is most precise for the data sets that were assimilated. These data sets are presented in Table 2.1. Much satellite data during the 2008-2009 solar minimum was assimilated during the development of the DTM2013 which enables the DTM2013 to perform well under similar conditions as the

TestLowNS and TestLowS data sets. Overall, the performance of the CPM is insensitive to the levels of solar activity in the training data sets. When training data sets of periods of low solar activity would only be available, using a CPM would be preferred to using an MLP model. This is because the CPM seems to have a better forecasting performance than the MLP models when trained on data sets with low solar activity and afterwards tested on data sets with any solar or geomagnetic activity. The CPM is, therefore, a good alternative for the DTM2013 when one needs to forecast density values at periods of low solar activity. The CPM is on top of this more easy to implement and set-up than the DTM2013 since one does not need to forecast the input values for this DTM2013.

Training an MLP model on a training data set with low solar activity is almost always inferior to training this model on a training data set with high solar activity. Therefore it is decided that in case of predicting density values in a period of high solar activity, one can best use a univariate MLP model trained on a data set with high solar and low geomagnetic activity. However, when training an MLP model for different solar activity than the training data set, adding the F_{15} solar index to the 99-8-8-6 MLP models seems to help in increasing the performance. A potential reason why a neural network does not perform well over periods of low solar activity is due to other factors that influence the variability in thermospheric density values. Their effect can become stronger if the solar perturbations weaken. Such effects could be related to weather phenomena in the lower atmospheric layers such as thunderstorms or even tides and waves. *Doornbos* [2012] also indicates that the dissipation of tidal motions and gravity waves are minor heat sources that are currently neglected in the empirical thermosphere models. This is considered as a topic beyond the scope of this report. Overall it also can be concluded that both baseline and MLP models do suffer in terms of forecasting performance when geomagnetic storms are present in the test data, even when the models are trained on a data set that includes periods of high geomagnetic activity. From Subsection 5.1.3 it was already concluded that the current MLP models are not robust enough to perform well under high geomagnetic activity. Similar conclusions were reached by *Lundstedt et al.* [2002] and *Lastovicka* [2002] which do force us to accept that forecasting peaks in geomagnetic activity is currently not achievable in the scope of this research study. Therefore, if one wants to predict the POD-derived density values it is recommended to give priority to the properties in terms of the solar activity of a training data set, before taking into account the presence of geomagnetic storms in this data set.

Based on the findings from Table 5.5 and Table 5.6, a set of different recommendations can be made if one wants to forecast the along-track density values at periods of different solar and geomagnetic activity. It is assumed that all past data can be used to train the MLP models on different solar and geomagnetic conditions. For periods of high solar activity, irrespective of the presence of geomagnetic storms, the univariate MLP model trained on a data set of high solar and low geomagnetic activity can best be used. For periods of low solar activity, also irrespective of the presence of geomagnetic storms, the DTM2013 can best be used to forecast future density values. Note that, these are still the DTM2013 values with the observed input values, which vary when the input values would have been predicted as well. It is therefore recommended for future research initiatives to start collecting published forecasts of the input solar and geomagnetic indices.

The performance metrics in Table 5.5 and Table 5.6 were used to discuss the baseline and MLP models' ability to perform under different solar and geomagnetic activity. However, the values presented in these tables are numerous and are easier to interpret visually. Moreover, the performance metrics in these tables are computed based on four test data sets of only three days each. A forecast of only three days is not more than a snapshot of a model's capabilities and does not allow to make a strong statement on the robustness of a specific forecasting model. Therefore, the CPM, DTM2013, 8-8-6 MLP and 99-8-8-6 MLP models are used to perform daily forecasts for September 2015. Figure 5.8 presents the results for

Table 5.5: Uni- and multivariate model performance analysis for different combinations of training and test data sets with a sampling period (t_s) of 120 s and a prediction window (W_p) of one day. The CPM and DTM2013 serve as baseline models to compare the MLP models' performance with. Training and test data sets vary in their solar and geomagnetic properties. Models are trained on data sets in periods of high solar activity.

Training Data	**Test Data**	**Model Configuration**	**RMSE**	**B**	**R**
STHighNS	TestHighNS	CPM, a=0.94 & b=-0.82	0.086	1.005	*0.881*
		DTM2013	*0.184*	*1.015*	0.912
		8-8-6 MLP, W_p=15 & D=10	0.059	0.998	0.902
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.064	1.003	0.902
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	**0.054**	**1.001**	**0.916**
	TestHighS	CPM, a=0.94 & b=-0.82	0.201	1.011	*0.763*
		DTM2013	*0.251*	*1.020*	**0.921**
		8-8-6 MLP, W_p=15 & D=10	0.140	**1.000**	0.805
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.137	1.004	0.832
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	**0.132**	1.004	0.842
	TestLowNS	CPM, a=0.94 & b=-0.82	0.116	0.995	0.891
		DTM2013	**0.093**	**1.004**	**0.926**
		8-8-6 MLP, W_p=15 & D=10	0.225	0.985	0.851
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.178	0.990	0.825
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*1.028*	*1.030*	*0.128*
	TestLowS	CPM, a=0.94 & b=-0.82	0.228	**0.991**	0.488
		DTM2013	**0.183**	1.010	**0.869**
		8-8-6 MLP, W_p=15 & D=10	0.319	0.982	0.414
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.278	0.986	0.392
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*13.423*	*0.568*	*-0.649*
STHighS	TestHighNS	CPM, a=0.93 & b=-0.97	**0.067**	**1.003**	0.881
		DTM2013	0.184	1.015	**0.912**
		8-8-6 MLP, W_p=15 & D=10	0.080	1.004	0.891
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.100	1.007	0.893
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.220*	*0.985*	*-0.075*
	TestHighS	CPM, a=0.93 & b=-0.97	0.183	1.008	0.763
		DTM2013	0.251	*1.020*	**0.921**
		8-8-6 MLP, W_p=15 & D=10	**0.159**	**1.006**	0.794
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.193	1.011	0.788
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.302*	1.016	*-0.121*
	TestLowNS	CPM, a=0.93 & b=-0.97	0.133	0.993	0.891
		DTM2013	**0.093**	**1.004**	**0.926**
		8-8-6 MLP, W_p=15 & D=10	0.161	0.991	0.852
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.127	0.996	0.826
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*1.068*	*1.050*	*-0.010*
	TestLowS	CPM, a=0.93 & b=-0.97	0.241	0.989	0.486
		DTM2013	**0.183**	1.010	**0.869**
		8-8-6 MLP, W_p=15 & D=10	0.263	0.987	0.423
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.232	**0.993**	0.380
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*6.668*	*0.819*	*-0.421*

Table 5.6: Uni- and multivariate model performance analysis for different combinations of training and test data sets with a sampling period (t_s) of 120 s and a prediction window (W_p) of one day. The CPM and DTM2013 serve as baseline models to compare the MLP models' performance with. Training and test data sets vary in their solar and geomagnetic properties. Models are trained on data sets in periods of low solar activity.

Training Data	**Test Data**	**Model Configuration**	**RMSE**	**B**	**R**
STLowNS	TestHighNS	CPM, a=0.94 & b=-0.84	**0.063**	1.019	0.881
		DTM2013	0.184	1.015	0.912
		8-8-6 MLP, W_p=15 & D=10	0.072	**1.005**	**0.934**
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.088	0.994	0.930
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.760*	*0.941*	*0.011*
	TestHighS	CPM, a=0.94 & b=-0.84	0.178	1.007	0.763
		DTM2013	0.251	1.020	**0.921**
		8-8-6 MLP, W_p=15 & D=10	0.192	1.008	0.751
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	**0.161**	**0.998**	0.778
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.421*	*0.972*	*-0.040*
	TestLowNS	CPM, a=0.94 & b=-0.84	0.143	0.992	0.891
		DTM2013	**0.093**	**1.004**	**0.926**
		8-8-6 MLP, W_p=15 & D=10	0.185	0.992	0.768
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.254	0.984	0.774
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.368*	*0.980*	*-0.033*
	TestLowS	CPM, a=0.94 & b=-0.84	0.249	0.988	0.488
		DTM2013	**0.183**	**1.010**	**0.869**
		8-8-6 MLP, W_p=15 & D=10	0.275	0.989	0.325
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.339	0.981	0.317
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.459*	*0.979*	*-0.109*
STLowS	TestHighNS	CPM, a=0.93 & b=-0.95	**0.077**	1.004	0.881
		DTM2013	0.184	1.015	**0.912**
		8-8-6 MLP, W_p=15 & D=10	0.115	1.004	0.530
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.111	**1.002**	0.487
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.460*	*0.964*	*0.000*
	TestHighS	CPM, a=0.93 & b=-0.95	0.195	1.010	0.763
		DTM2013	*0.251*	*1.020*	**0.921**
		8-8-6 MLP, W_p=15 & D=10	**0.189**	1.008	0.699
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.189	1.008	0.703
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	0.234	**0.995**	*0.000*
	TestLowNS	CPM, a=0.93 & b=-0.95	0.120	*0.994*	0.891
		DTM2013	**0.093**	1.004	**0.926**
		8-8-6 MLP, W_p=15 & D=10	0.156	0.991	0.841
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.142	0.993	0.846
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.350*	**1.001**	*0.042*
	TestLowS	CPM, a=0.93 & b=-0.95	0.231	0.991	0.488
		DTM2013	**0.183**	1.010	**0.869**
		8-8-6 MLP, W_p=15 & D=10	0.244	*0.988*	0.512
		8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.235	0.989	0.517
		8-8-6 MLP$_{Dst}$, W_p=15 & D=15	*0.307*	**1.004**	*0.440*

the two baseline and three MLP models (8-8-6 and 99-8-8-6 with F_{15} and D_{st} added) for all four different Swarm-A training data sets. The Modified Taylor Diagram is used to visually present a summary of the statistical results of comparing the forecasts of those models and the POD-derived density values of Swarm-A, as discussed in Section 3.5. The yellow star indicates the bias-free hypothetical truth point to which the models are referenced. The *Norm Factor* is used to scale the relevant statistics.

Figure 5.8 confirms the conclusions that were made from the results in Table 5.5 and Table 5.6. Firstly, in terms of relative MLP model performance, the 8-8-6 MLP and the 99-8-8-6 MLP with the additional F_{15} index are the best performing MLP models. The multivariate MLP model with additional D_{st} is not even present on all plots, due to the unstable behaviour that was also witnessed in Figure 5.7. Secondly, the DTM2013 performs best overall on this month of medium-to-low solar activity. This corresponds to the conclusions made in the previous earlier. The thermospheric thermosphere model outperforms the other models for all performance metrics except for the normalised bias. Thirdly, the CPM model is more robust to training data sets with different solar and geomagnetic activity. This can be explained by the biases and scale factors which are quite similar for all four training data sets. Finally, the CPM's performance is better than the one of the MLP models if both model types are trained on a data set with low solar activity.

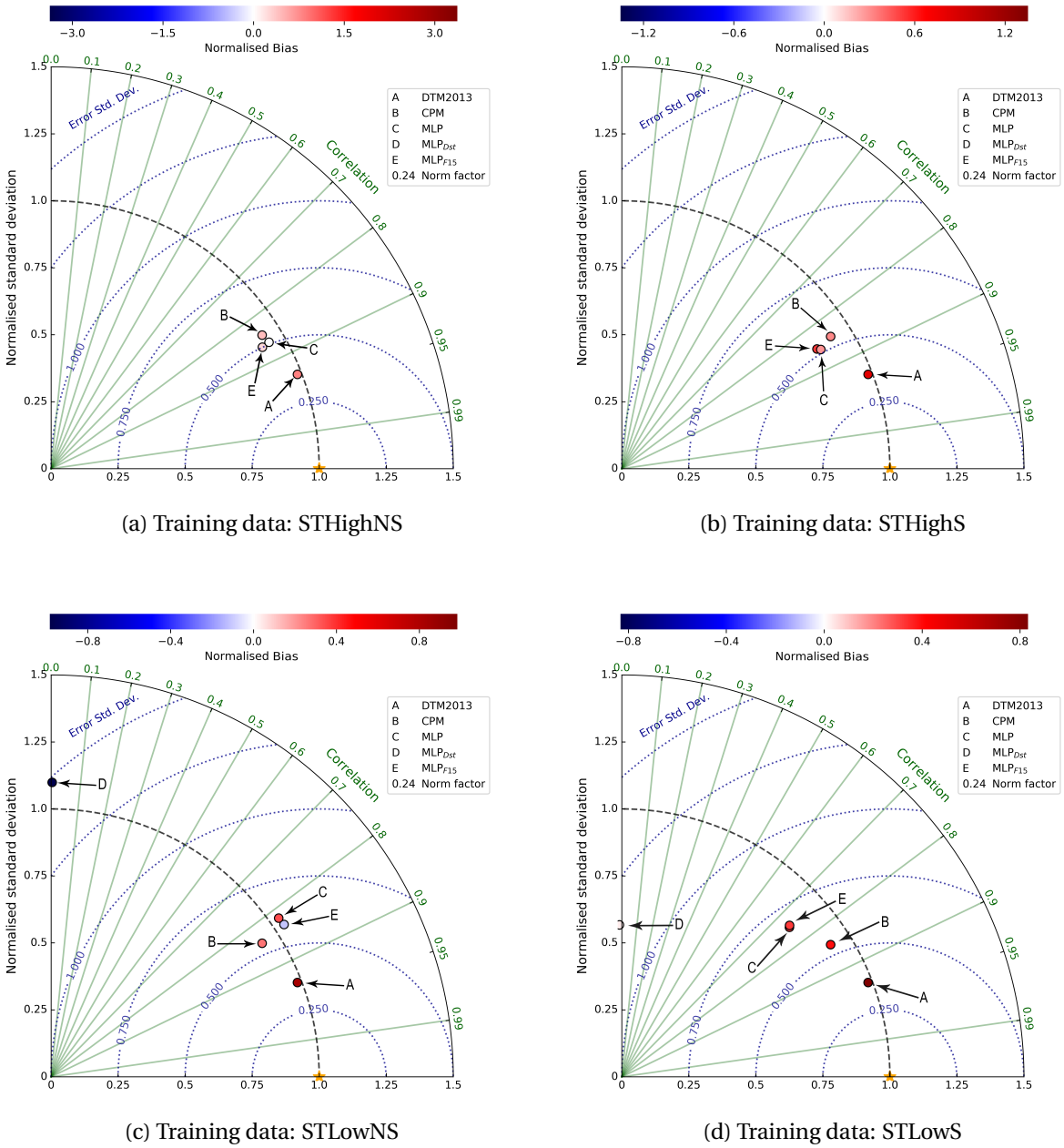
5.3.3 Variability in Orbital Elements

In the previous subsection the goal was to test the ability of the CPM and MLP models to forecast the POD-derived density values of the Swarm-A satellite while trained on Swarm-A data sets with periods of different solar and geomagnetic conditions compared to the test data sets. The difference in this subsection is that the same models (i.e. trained on Swarm-A data sets) will be tested if they are able to forecast the POD-derived densities measured by the Swarm-C and Swarm-B satellites which fly in orbits with different orbital elements. The Swarm-C satellite is used to test the forecast capabilities for satellites flying in orbits with similar orbital elements where only the longitude slightly varies. The Swarm-B satellite is used as a test case for two satellites flying at different orbital altitudes. All three satellites are identical.

This subsection will test the forecast performance of the baseline CPM and DTM2013, as well as the best performing univariate MLP model of Subsection 5.1.3 and multivariate MLP model with additional F_{15} index of Subsection 5.3.1. The CPM and all MLP models are going to be trained on the STHighNS and STLowNS training data sets of Swarm-A (see Table 4.1) with a sampling period of 120 s and prediction window of one day. Periods of low geomagnetic activity are picked due to the poor forecasting performance of the trained CPM and MLP models during periods of high geomagnetic conditions. Afterwards, the forecasting performance of every trained model is examined by testing the model on two periods of both high and low solar activity. These two test data sets correspond time-wise to the TestHighNS and TestLowNS data sets but are composted of Swarm-B and Swarm-C data. A daily forecast is performed for all three days in the test data sets. The performance metrics, as discussed in Section 3.5, are used to assess the forecasting capabilities. Table 5.7 lists all the performance metrics for the models that are tested on the Swarm-C satellite, while Table 5.8 lists the results for the models that are tested on the Swarm-B satellite.

Test case: Swarm-C

The Swarm-A and Swarm-C satellites form the lower pair of the satellite constellation flying side-by-side with a 1.4° separation in longitude at the equator and at an initial orbital altitude of 462 km. Both satellites fly in close formation to monitor the small time-variability aspects of the geomagnetic field. It is



(a) Training data: STHighNS

(b) Training data: STHighS

(c) Training data: STLowNS

(d) Training data: STLowS

Figure 5.8: Modified Taylor diagram showing the low-order statistics for all best performing uni- and multivariate MLP models for training data sets *(a)* STHighNS, *(b)* STHighS, *(c)* STLowNS and *(d)* STLowS. Model forecasts are compared to POD-derived Swarm-A density values in September 2015. MLP models with t_s = 120 s and W_p = 15 with varying D.

advised to compare the results in Table 5.7 to the results in Table 5.5 and Table 5.6 to see the impact of a different longitude in training and test data set. Overall, one can conclude that all CPM and MLP models do perform worse on the Swarm-C test data than when they were tested on the Swarm-A test data set. When the forecasting performance of the MLP model is compared to the baseline CPM and DTM2013, one can conclude that using an MLP model does deliver less accurate density forecasts. The CPM, however, seems to be the best performing model overall. In conclusion, if one wants to forecast

Table 5.7: Uni- and multivariate model performance analysis for different combinations of training and test data sets with a sampling period (t_s) of 120 s and a prediction window (W_p) of one day. The CPM and DTM2013 serve as baseline models to compare the MLP models' performance with. The models are trained on Swarm-A density data, while tested on Swarm-C density data in both high and low solar activity without the presence of geomagnetic storms.

Training Data	**Test Data**	**Model Configuration**	**RMSE**	**B**	**R**
STHighNS	20/11/2014 -	CPM, a=0.94 & b=-0.82	**0.065**	**0.998**	*0.870*
	22/11/2014	DTM2013	0.096	1.007	**0.924**
	(High solar	8-8-6 MLP, W_p=15 & D=10	*0.127*	*0.990*	0.907
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.080	0.995	0.911
	15/05/2016 -	CPM, a=0.94 & b=-0.82	**0.159**	0.992	0.192
	17/05/2016	DTM2013	0.190	**1.001**	**0.524**
	(Low solar	8-8-6 MLP, W_p=15 & D=10	*0.337*	*0.982*	0.192
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.303	0.987	*0.125*
STLowNS	20/11/2014 -	CPM, a=0.94 & b=-0.84	0.089	0.994	*0.870*
	22/11/2014	DTM2013	0.096	1.007	**0.924**
	(High solar	8-8-6 MLP, W_p=15 & D=10	**0.063**	**0.997**	0.908
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	*0.172*	*0.986*	0.915
	15/05/2016 -	CPM, a=0.94 & b=-0.84	**0.186**	0.989	-0.006
	17/05/2016	DTM2013	0.190	**1.001**	**0.524**
	(Low solar	8-8-6 MLP, W_p=15 & D=10	0.319	0.990	*-0.006*
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	*0.372*	*0.982*	-0.004

the along-track density values of a satellite with a similar form factor in an orbit with a slightly different longitude (eg. after an orbital manoeuvre), one can best use the baseline CPM model as a first approach. Moreover, the CPM model is attractive due to the ease of implementation and does not require any expert knowledge on neural networks or thermospheric density models.

As a remark, one could argue that the POD-derived density values of the Swarm-C satellite could be used as an additional training data set. This could be an interesting experiment for future research studies if no additional efforts are made to interpolate the time gaps that are currently present in some months of the training data sets (see Table C.1). However, one risk might be that the MLP model will be subjected to overfitting since too much redundant information is fed to the model during the training phase of this neural network. The similarities in the along-track densities of both Swarm-A and Swarm-C can be seen in the POD-derived densities which are plotted in the *bottom* graph of Figure 2.3.

Test case: Swarm-B

The Swarm-A and Swarm-B satellites fly in orbits with quasi-similar orbital elements. The main difference is that both satellites fly at different orbital altitudes in different orbital planes. This difference in orbital altitude becomes clear when taking a closer look at the *top* graph in Figure 2.3. The challenge in this specific test case is that the CPM and MLP models are trained on data sets where the density values are much larger than the along-track density values the Swarm-B satellite encountered at the same periods in time. The variability in thermospheric density is showcased in Figure 2.2 which shows that the thermospheric density decreases almost exponentially with altitude. It is recommended to compare the results in Table 5.8 to the results in Table 5.5 and Table 5.6 to see the impact of a difference in orbital altitude in training and test data set.

Table 5.8: Uni- and multivariate model performance analysis for different combinations of training and test data sets with a sampling period (t_s) of 120 s and a prediction window (W_p) of one day. The CPM and DTM2013 serve as baseline models to compare the MLP models' performance with. The models are trained on Swarm-A density data, while tested on Swarm-B density data in both high and low solar activity without the presence of geomagnetic storms.

Training Data	**Test Data**	**Model Configuration**	**RMSE**	**B**	**R**
STHighNS	20/11/2014 -	CPM, a=0.94 & b=-0.82	**0.072**	**1.001**	0.746
	22/11/2014	DTM2013	0.117	1.008	**0.841**
	(High solar	8-8-6 MLP, W_p=15 & D=10	*0.161*	*0.992*	0.012
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.143	0.997	*-0.168*
	15/05/2016 -	CPM, a=0.94 & b=-0.82	**0.172**	0.993	**0.395**
	17/05/2016	DTM2013	0.216	**1.000**	*-0.493*
	(Low solar	8-8-6 MLP, W_p=15 & D=10	*0.332*	*0.981*	-0.448
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	0.276	0.987	-0.369
STLowNS	20/11/2014 -	CPM, a=0.94 & b=-0.84	**0.077**	0.998	0.746
	22/11/2014	DTM2013	0.117	1.008	**0.841**
	(High solar	8-8-6 MLP, W_p=15 & D=10	0.119	**1.000**	0.208
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	*0.181*	*0.990*	*-0.029*
	15/05/2016 -	CPM, a=0.94 & b=-0.84	**0.195**	0.990	**0.395**
	17/05/2016	DTM2013	0.216	**1.000**	-0.493
	(Low solar	8-8-6 MLP, W_p=15 & D=10	0.271	0.989	*-0.521*
	activity)	8-8-6 MLP$_{F15}$, W_p=15 & D=15	*0.313*	*0.983*	-0.431

A clear insight is that all MLP models do perform worse than when they would have been trained on training data sets of the same satellite for which they need to forecast the along-track density values. The same conclusion as in Subsection 5.3.2 still holds that the solar activity between the training and test data needs to be similar to deliver the best MLP performance. The CPM is the best alternative if there does not exist any opportunity to retrain the MLP model on a training data set of a satellite with similar solar and geomagnetic activity, as well as similar orbital elements. Similar to the previous test case, the CPM has overall the best performance of all four models. The CPM even sees an absolute improvement in terms of RMSE values when tested on periods of high solar activity in comparison to its performance on Swarm-A test data. This means that the scale factor and bias to transform the regular PM to a CPM hold when differences in orbital elements are present. The performance of the DTM2013 improves during high solar activity, but worsens during periods of low solar activity when compared to the performance metrics in Table 5.5 and Table 5.6. However, no general conclusion can be made solely based on these two test data sets. *Bruinsma* [2015] concluded that in general, the DTM2013 is most precise for the data that is assimilated into the model. The Swarm-B satellite operated at orbital altitudes between 515 km and 530 km from 2014 up until the end of 2017. No data sets of satellites that fly at this orbital altitude were used in the construction of the DTM2013. This could explain why the performance of the DTM2013 varies for both test cases. One should be aware that this conclusion is reached by examining the performance over only a short period in time. Many interesting case studies are still left to explore. As an example, it would be interesting to further investigate how much the orbital altitude needs to change before a specific model cannot be used anymore. It would also be interesting to find out how the forecasting performance of a trained model would behave if tested on cases where satellites operate in orbits with high eccentricity. This is left for future research.

Chapter 6

Conclusions and Recommendations

The goal of this thesis was to develop a tool which is able to improve the forecasting accuracy of along-track density values by using machine learning models and past acceleration and solar activity data sets provided by the Swarm satellite mission and space weather observatories, respectively. The development of such a tool would help in concluding if machine learning predictors are a feasible option to improve thermospheric density forecasts to such a degree they can be used in future space missions and research efforts. Section 6.1 will come up with a conclusion on both the thesis goal and research objective. Machine learning is currently one of the state-of-the-art solutions for many problems in the science and engineering space. Therefore, a set of recommendations will be listed in Section 6.2 in the light of this highly researched subset of artificial intelligence.

6.1 Conclusions

To give an answer on the goal and research objective of this thesis, the following research question was set-up: *Can Artificial Neural Networks (ANNs) forecast thermospheric densities and generalise beyond the properties of an acceleration dataset using the Swarm satellites as study case?* This research question was broken down into two parts which will be answered next. In order to provide an answer to these questions, three forecasting models have been set-up. First, the Multi-Layer Perceptron (MLP) model is the ANN under investigation in this research study since it has shown promising performance in literature, while allowing the internal architecture to be tweaked such that the following questions can be answered. Two baseline models are used as reference forecasting models, namely the Calibrate Persistence Model (CPM) and the DTM2013. Note that, the predictions of the DTM2013 are in fact the density values modelled with the observed input values. These input values are the known orbital state, and solar and geomagnetic activity at a given point in time. The CPM is constructed by applying three transformation parameters to the regular persistence model, hence increasing its overall forecasting performance; i.e. a horizontal displacement of h seconds, followed by a scale factor and additional bias term.

1) What is the overall forecasting performance of an ANN model when predicting the along-track density values of the Swarm satellites? To answer this question the best performing internal architecture for the MLP model needed to be created first. A stable environment was therefore set-up such that both the CPM and MLP model could be trained and tested on periods of similar high solar and low geomagnetic activity. Finally, the best performance was achieved for a three-layer MLP model with eight neurons on its first hidden layer and six neurons on both the second and third hidden layers at a sampling frequency of 120 s. The range of delay variables that are incorporated in the input vector was found to be optimal when

around five to 15 orbital periods worth of data was added. This specific model architecture allowed for daily forecasting performances superior to both baseline models in terms of the root-mean-squared error values. An increase in performance of 18% and 72% was achieved compared to the CPM and DTM2013 baseline models, respectively, when tested on a test day with similar solar and geomagnetic activity as the training data set. An MLP model with two hidden layers did also show good performance during the nested cross-validation test with average RMSE values similar to the MLP model with three hidden layers. Such models are recommended when the time to train the MLP models is of primary importance. Single-layer MLP models were discarded since they were susceptible to underfitting when too many additional variables in the input vector were added. If more than three hidden layers or ten neurons on one layer were introduced to the MLP model, the model overfitted on the training data set.

The forecasting performance of the 8-6-6 MLP model degraded when POD-derived density values needed to be forecasted for prediction windows beyond one day. It was shown that the more the prediction window increases beyond one day, the more challenging it was for the MLP model to learn the complex relations between its input vector and the target value. The MLP models also required a larger input vector with past data to enable its neurons to learn the information embedded in the training data set. Nevertheless, the MLP model was able to outperform the CPM and DTM2013 predictions for forecasts up until two days. The CPM is suggested for forecasts of more than two days into the future due its ease of implementation.

Introducing additional information to the MLP model has shown the potential to increase the forecasting performance of MLP models. This has been done in two different ways, first by increasing the training data set in size, and second by adding an additional input vector to the univariate MLP model. Training the univariate MLP model on a larger data set did not show significant improvements in overall forecasting performance. Regardless of the fact that the MLP model still performed better than the other two baseline models, the forecasting accuracy decreased in comparison to the original data set. A potential explanation is that only a maximum of four months worth of density data could be used due to the presence of large data gaps in the rest of the monthly Swarm data. This is left for future research. Extra information was also added by introducing an additional input vector that contains solar or geomagnetic indices. Introducing the Kyoto D_{st} index has shown to generally deteriorate the forecasting performance in comparison to the univariate and other two baseline models. On the other hand, when additional solar indices were introduced the sensitivity of the MLP model was more nuanced. In the best case, such a multivariate model did achieve similar or slightly better performance when compared to the CPM. A possible explanation is that a maximum of only two different solar indices can be encapsulated in the input vector. The 15 cm and 10.7 cm solar radio fluxes are considered as the best candidates. Overall, the addition of the F_{15} solar index resulted in the best performing multivariate MLP model. A potential explanation is that this solar index has the strongest linear correlation with the POD-derived density values.

2) Can ANNs generalise beyond the properties of a given acceleration training data set for satellites with the same form factor? To answer this questions an assessment was made of the change in forecasting performance of the baseline CPM and DTM2013, as well as the best performing uni- and multivariate MLP models when these were trained and tested on data sets with similar or different solar and geomagnetic activity, as well as different orbital elements in terms of longitude and orbital altitude. In terms of variations in solar activity, it was concluded that in case of predicting density values in a period of high solar activity, one can best use a univariate MLP model trained on a data set with high solar and low geomagnetic activity. However, when training an MLP model for different solar activity than the test data set, adding the F_{15} solar index to the 99-8-8-6 MLP models seemed to help in increasing its performance. When one would like to predict the along-track density values for periods characterised by low solar ac-

tivity, the DTM2013 has shown to outperform both the CPM and MLP models, irrespective from the levels of geomagnetic activity. A downside of using the DTM2013 is that one needs to forecast the input values as well. A suggested alternative is to use the CPM as predictor if only training data sets of periods of low solar activity would be available. This is because the CPM seems to have a better forecasting performance than the MLP models when trained on data sets with low solar activity and afterwards tested on data sets with any solar or geomagnetic activity. An additional benefit is that the CPM is more easy to implement and set-up than the DTM2013 since one does not need to forecast the input values for this DTM2013.

In terms of forecasting periods of high geomagnetic activity it was concluded that both baseline and MLP models do suffer in terms of forecasting performance when geomagnetic storms are present in the test data, even when the models are trained on a data set that includes periods of high geomagnetic activity. For this reason, if one wants to predict the along-track density values for a LEO satellite it is recommended to first take into account the different levels of solar activity between the training and test data sets, before looking into the potential presence of geomagnetic storms throughout the test data set.

Finally, the MLP model does not have the ability to perform adequately when trained on a data set with different orbital longitude and altitude compared to the test data set. This situation could occur for identical satellites that are part of a constellation (eg. Swarm satellite constellation) or when a satellite ends up in a different orbit after having performed an orbital manoeuvre. The CPM model has shown to be the best alternative in this case, if it is not possible to retrain the MLP model. Due to time constraints and the lack of long periods of high quality data, a detailed analysis in terms of spatial and temporal variations is still recommended for future research.

6.2 Recommendations

The promise of machine learning is that it can solve complex problems automatically, faster and more accurately than a manually specified solution and at a larger scale. The complex, nonlinear nature of the interactions between the Sun and the Earth's atmosphere require more advanced tools than those traditionally learned and used by space physicists. This thesis has shown the potential of using machine learning techniques (eg. neural network models) to forecast the along-track density values of a LEO satellite. A set of recommendations in terms of the development of the current model and future research topics is described in Subsection 6.2.1 and Subsection 6.2.2, respectively. The main goal of these recommendations is to first improve the robustness of the neural network model before it is introduced to a new set of promising applications.

6.2.1 Model Improvements

1) Performing an in-depth hyperparameter analysis: Some of the hyperparameters were kept constant throughout this whole thesis project (eg. percentage validation data, cost function, stochastic gradient descent, activation function and bias types). Although they were chosen based on references in literature and their overall good performance, their effect on an MLP's performance shall be examined more thoroughly. This was left out of this research project because tweaking hyperparameters can cause MLP models to train very slowly. It is therefore strongly recommended to look into different hyperparameter configurations if one wants to reach the best MLP performance possible. Another recommendations is to tweak these hyperparameters by training and testing these different hyperparameter settings on more data with periods of different solar and geomagnetic activity. A potential technique could be to grid search the hyperparamaters by using the scikit-learn machine learning library in Python.

2) Performing sophisticated interpolation techniques for erroneous and missing data points: *Domingos* [2012], among others, states that improving the performance of a neural network model is best achieved by adding more training data before looking into different and more complex neural network types. Due to time constraints, the flagged and missing data points in the Swarm data set were not corrected. This had as a consequence that only a training data set of a maximum of four months worth of data points has been used. It is therefore strongly recommended to implement sophisticated interpolation techniques to cope with those imperfect time stamps. By doing so, the MLP models can be trained on years worth of training data with a potential increase in forecasting performance as a consequence. The reprocessing efforts of these data points is currently planned by the processing facilities which will significantly reduce these data gaps.

3) Assessing different multi-layer perceptron architectures: The goal of this thesis project was the assess the feasibility of using machine learning techniques to forecast the along-track density values of LEO satellites. The 8-8-6 and 99-8-8-6 MLP architectures were chosen, but many other possibilities exist in redefining how such a model should look like. As an example, one could merge the different input vectors into one input vector directly, or create an MLP model with multiple input and output values. The difficulty here is in the preparation and the definition of the shape of the data sets. It is therefore suggested to asses the differences in performance for such model architectures.

4) Assessing the effect of adding alternative data sources as input: Only empirical model data, as well as specific solar and geomagnetic indices were used as input values for the uni- and multivariate MLP models. It would be interesting to see what the effect would be when different types of input data were used. The empirical model data set could be enlarged by adding density data from other thermosphere models or replace this data by for example accelerometer data from the satellite itself. Instead of using solar EUV proxies, one could potentially train the MLP model on space-based EUV measurements. As an example, one could use the measurements of the Solar EUV Monitor (SEM) aboard the SOlar and Heliospheric Observatory (SOHO) satellite which has been measuring the solar EUV flux data since 1996 [*Judge et al.*, 1998]. However, a counterargument of this would be that the data produced by those instruments is not easily accessible on a daily basis. Another example for the specific case of the Swarm satellite constellation would be to replace the geomagnetic indices by the geomagnetic information collected by its onboard instruments.

5) Assessing the performance of alternative neural network types: The field of machine learning is continuously evolving, leading to more complex models than ever before. Nevertheless, it is better to start with the development of the less complex models to get a feel for the problem that is at stake. However, if one would like to reach optimal performance it is recommended to rely on the most state-of-the-art model developments. As an example, one could take a look at Long Short-Term Memory models (LSTM) which are a type of recurrent neural network that allows a more constant error to be back-propagated through time and layers [*Hochreiter and Schmidhuber*, 1997]. LSTMs are the ideal next step if one wants a neural network model to learn how to store information over extended time intervals, which could show more robustness against changes in solar activity.

6.2.2 Outlook for Further Research

1) Design for operational requirements: In this thesis study the goal was to create an MLP model that was optimised to make density forecasts as accurate as possible. Nevertheless, it is recommended to find synergies between the academic world and operational environment. It would therefore be interesting to take other end user requirements into account such as efficiency of the software and software-hardware

interaction, reliability of the network and automation of the software. Another interesting case study would be to examine how computationally intensive it is to retrain the weights of these MLP models every time new observations become available. This could allow a satellite to perform the orbit propagation autonomously on its onboard computers.

2) Assess the effect of improved density predictions on the orbit determination and forecasting phase: The requirements for satellite orbit prediction are not constantly met due to the lacking performance of the atmospheric density models. In this report the focus was only on the forecast performance of the thermospheric density values. It is still unclear how the improvements in forecasting the along-track density values benefit on the outcome of the orbit propagation techniques. It would therefore be interesting to test the actual effect on the prediction of orbital trajectories when an MLP model is introduced.

3) Promote density data collection and preprocessing in past and future missions: Before an MLP model can be used as the new standard in forecasting thermospheric densities, the method needs to proof that its performance is robust to changes in solar and geomagnetic activity and changes in a satellite's orbital elements as well as form factor. It is therefore essential to train and test these MLP models on different satellite cases. The density products (eg. POD-derived and accelerometer-derived density data) used in this thesis have already been preprocessed before being released for scientific use. It is recommended to keep promoting the preprocessing of satellite data such that plenty along-track density products become available for future testing.

4) Work towards a model to improve density forecasting performance irrespective from the type of satellite: This thesis only investigated the robustness of an MLP model for satellites flying in orbits with different longitudes and orbital altitudes. It would be interesting to examine if machine learning models are still effective when applied to satellite missions with specialised orbits. Long-term data sets such as the one examined by *Emmert* [2009], which derived the density values from the orbits of $\tilde{5}000$ objects from routine orbital elements compiled by the U.S. Air Force, could be a good starting point.

Bibliography

[1] R. F. Berger and G. Casella. *Statistical Inference: 2nd Edition*. Duxbury Press, 2001. ISBN 0-534-24312-6.

[2] C. Bergmeir and J. M. Benítez. On the Use of Cross-validation for Time Series Predictor Evaluation. *Information Sciences*, Volume 191: pages 192–213, 2012. ISSN 0020-0255. doi: 10.1016/j.ins.2011.12.028.

[3] A. Borovykh, S. Bohte, and C. W. Oosterlee. Conditional Time Series Forecasting with Convolutional Neural Networks. *arXiv e-prints*, 2017. URL `http://arxiv.org/abs/1703.04691`.

[4] B. Bowman, W. K. Tobiska, F. Marcos, C. Huang, C. Lin, and W. Burke. A New Empirical Thermospheric Density Model JB2008 Using New Solar and Geomagnetic Indices. *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2008. ISSN 2008-6438. doi: 10.2514/6.2008-6438.

[5] S. Bruinsma. The DTM-2013 Thermosphere Model. *Journal of Space Weather and Space Climate*, Volume 5, 2015. ISSN 2115-7251. doi: 10.1051/swsc/2015001.

[6] S. Bruinsma, D. Arnold, A. Jäggi, and N. Sánchez-Ortiz. Semi-empirical Thermosphere Model Evaluation at Low Altitude with GOCE Densities. *Journal of Space Weather and Space Climate*, Volume 7: article A4, 2017. ISSN 2115-7251. doi: 10.1051/swsc/2017003.

[7] E. Camporeale, S. Wing, J. Johnson, C. M. Jackman, and R. McGranaghan. Space Weather in the Machine Learning Era: A Multidisciplinary Approach. *Space Weather*, Volume 16: pages 2–4, 2018. ISSN 15427390. doi: 10.1002/2017SW001775.

[8] CCMC. Community coordinated modeling center, 2018. URL `https://ccmc.gsfc.nasa.gov/index.php`. Online accessed on 2018-04-15.

[9] A. Choury, S. Bruinsma, and P. Schaeffer. Neural Networks to Predict Exosphere Temperature Corrections. *Space Weather*, Volume 11: pages 592–602, 2013. ISSN 15427390. doi: 10.1002/2013SW000969.

[10] J.M.H. Claes. Thermospher Modelling using Machine Learning. *AE4020 - Literature Study*, 2018.

[11] C. F. M. Coimbra and H. T. C. Pedro. Chapter 15 - stochastic-learning methods. *Solar Energy Forecasting and Resource Assessment*, pages 383–406, 2013. doi: 10.1016/B978-0-12-397177-7.00015-2.

[12] P. Domingos. A Few Useful Things to Know About Machine Learning. *Communications of the ACM*, Volume 55, 2012. ISSN 00010782. doi: 10.1145/2347736.2347755.

[13] R. V. Donner, V. Stolbova, G. Balasis, J. F. Donges, M. Georgiou, S. M. Potirakis, and J. Kurths. Temporal organization of magnetospheric fluctuations unveiled by recurrence patterns in the Dst index. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, Volume 28, 2018. doi: 10.1063/1.5024792.

[14] E. Doornbos. *Thermospheric Density and Wind Determination from Satellite Dynamics*. Springer-Verlag Berlin Heidelberg, 2012. ISBN 978-3-642-25128-3. doi: 10.1007/978-3-642-25129-0.

[15] T. Dudok De Wit and S. Bruinsma. The 30 cm Radio Flux as a Solar Proxy for Thermosphere Density Modelling. *Journal of Space Weather and Space Climate*, Volume 7: article A9, 2017. doi: 10.1051/swsc/2017008.

[16] E. Echer, W. D. Gonzalez, F. L. Guarnieri, A. D. Lago, and L. E. A. Vieira. Introduction to Space Weather. *Advances in Space Research*, Volume 35: pages 855–865, 2005. doi: 10.1016/j.asr.2005.02.098.

[17] S. Elvidge, M. J. Angling, and B. Nava. On the Use of Modified Taylor Diagrams to Compare Ionospheric Assimilation Models. *Radio Science*, Volume 49: pages 737–745, 2014. doi: 10.1002/2014RS005435.

[18] S. Elvidge, H. C. Godinez, and M. J. Angling. Improved Forecasting of Thermospheric Densities Using Multi-Model Ensembles. *Geoscientific Model Development*, Volume 9: pages 2279–2292, 2016. ISSN 19919603. doi: 10.5194/gmd-9-2279-2016.

[19] J. T. Emmert. A Long-term Data Set of Globally Averaged Thermospheric Total Mass Density. *Journal of Geophysical Research: Space Physics*, Volume 114: pages 1–17, 2009. ISSN 21699402. doi: 10.1029/2009JA014102.

[20] W. Enders. *Applied Econometric Time Series: 4th Edition*. Wiley, 2014. ISBN 978-1-118-80856-6.

[21] J. H. F. Flores, P. M. Engel, and R. C. Pinto. Autocorrelation and Partial Autocorrelation Functions to Improve Neural Networks Models on Univariate Time Series Forecasting. *The 2012 International Joint Conference on Neural Networks*, pages 1–8, 2012. ISSN 2161-4407. doi: 10.1109/IJCNN.2012.6252470.

[22] X. Glorot, A. Bordes, and Y. Bengio. Deep Sparse Rectifier Neural Networks. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Volume 15: pages 315–323, 2011. URL `http://proceedings.mlr.press/v15/glorot11a.html`.

[23] Z. Guoqiang, B. E. Patuwo, and M. Y. Hu. Forecasting with Artificial Neural Networks: The State of the Art. *International Journal of Forecasting*, Volume 14: pages 35–62, 1998. ISSN 0169-2070. doi: 10.1016/S0169-2070(97)00044-7.

[24] S. Haykin. *Neural Networks and Machine Learning: 3rd Edition*. Pearson Education Inc., 2009. ISBN 9780131471399.

[25] C. J. Henney, W. A. Toussaint, S. M. White, and C. N. Arge. Forecasting F10.7 with Solar Magnetic Flux Transport Modeling. *Space Weather*, Volume 10, 2012. doi: 10.1029/2011SW000748.

[26] S. Hochreiter and J. U. Schmidhuber. Long Short-Term Memory. *Neural Computation*, Volume 9: pages 1735–1780, 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

[27] W. W. Hsieh and B. Tang. Applying Neural Network Models to Prediction and Data Analysis in Meteorology and Oceanography. *Bulletin of the American Meteorological Society*, Volume 79: pages 1855–1870, 1998. doi: 10.1175/1520-0477(1998)079<1855:ANNMTP>2.0.CO;2.

[28] J. R. Johnson and S. Wing. A Solar Cycle Dependence of Nonlinearity in Magnetospheric Activity. *Journal of Geophysical Research: Space Physics*, Volume 110: article A9, 2005. doi: 10.1029/2004JA010638.

[29] D. L. Judge, D. R. McMullin, H. S. Ogawa, D. Hovestadt, B. Klecker, M. Hilchenbach, E. Möbius, L. R. Canfield, R. E. Vest, R. Watts, C. Tarrio, M. Kühne, and P. Wurz. First Solar EUV Irradiances Obtained from SOHO by the CELIAS/SEM. *Solar Electromagnetic Radiation Study for Solar Cycle 22*, Volume 177: pages 161–173, 1998. doi: 10.1007/978-94-011-5000-2_12.

[30] D. P. K. Kingma and J. L. Ba. Adam: a Method for Stochastic Optimization. *Internal Conference on Learning Representations*, pages 1–13, 2015. URL `https://arxiv.org/abs/1412.6980`.

[31] T. Kodikara, B. Carter, and K. Zhang. The First Comparison Between Swarm-C Densities and Physical and Empirical Model Estimates. *Journal of Geophysical Research: Space Physics*, Volume 123: pages 5068–5086, 2018. doi: 10.1029/2017JA025118.

[32] J. Lastovicka. Monitoring and Forecasting of Ionospheric Space Weather - Effects of Geomagnetic Storms. *Journal of Atmospheric and Solar-Terrestrial Physics*, Volume 64: pages 697–705, 2002. doi: 10.1016/S1364-6826(02)00031-7.

[33] J. A. Lazzús, P. Vega, P. Rojas, and I. Salfate. Forecasting the Dst Index Using a Swarm-optimized Neural Network. *Space Weather*, Volume 15: pages 1068–1089, 2017. doi: 10.1002/2017SW001608.

[34] H. G. Lewis, G. G. Swinerd, and R. J. Newland. The Space Debris Environment: Future Evolution. *The Aeronautical Journal*, Volume 115: pages 241–247, 2016. doi: 10.1017/S0001924000005698.

[35] H. Lundstedt, H. Gleisner, and P. Wintoft. Operational Forecasts of the Geomagnetic Dst Index. *Geophysical Research Letters*, Volume 29: pages 34.1–34.4, 2002. doi: 10.1029/2002GL016151.

[36] W. D. McClain and D. A. Vallado. *Fundamentals of Astrodynamics and Applications*. Space Technology Library. Springer Netherlands, 2001. ISBN 978-0-7923-6903-5.

[37] N. Olsen, E. Friis-Christensen, R. Floberghagen, P. Alken, C. D. Beggan, A. Chulliat, E. Doornbos, J. T. da Encarnação, B. Hamilton, G. Hulot, J. van den IJssel, A. Kuvshinov, V. Lesur, H. Lühr, S. Macmillan, S. Maus, M. Noja, P. E. H. Olsen, J. Park, G. Plank, C. Püthe, J. Rauberg, P. Ritter, M. Rother, T. J. Sabaka, R. Schachtschneider, O. Sirol, C. Stolle, E. Thébault, A. W. P. Thomson, L. Tøffner-Clausen, J. Velímský, P. Vigneron, and P. N. Visser. The Swarm Satellite Constellation Application and Research Facility (SCARF) and Swarm data products. *Earth, Planets and Space*, Volume 65: pages 1189–1200, 2013. doi: 10.5047/eps.2013.07.001.

[38] D. Pérez and R. Bevilacqua. Neural Network Based Calibration of Atmospheric Density Models. *Acta Astronautica*, Volume 110: pages 58–76, 2015. ISSN 00945765. doi: 10.1016/j.actaastro.2014.12.018.

[39] D. Pérez, B. Wohlberg, T. A. Lovell, M. Shoemaker, and R. Bevilacqua. Orbit-centered Atmospheric Density Prediction Using Artificial Neural Networks. *Acta Astronautica*, Volume 98: pages 9–23, 2014. ISSN 00945765. doi: 10.1016/j.actaastro.2014.01.007.

[40] M. Qi and G. P. Zhang. Trend Time – Series Modeling and Forecasting With Neural Networks. *IEEE Transactions on Neural Networks*, Volume 19: pages 808–816, 2008. doi: 10.1109/TNN.2007.912308.

[41] L. Qian and S. C. Solomon. Thermospheric Density : An Overview of Temporal and Spatial Variations. *Space Science Reviews*, Volume 168: pages 147–173, 2012. doi: 10.1007/s11214-011-9810-z.

[42] S. Ruder. An Overview of Gradient Descent Optimization Algorithms. pages 1–14, 2017. URL https://arxiv.org/abs/1609.04747.

[43] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Representations by Back-propagating Errors. *Nature*, Volume 323: pages 533–536, 1986.

[44] C. Siemes, J. De Teixeira Da Encarnação, E. Doornbos, J. Van Den Ijssel, J. Kraus, R. Pereštý, L. Grunwaldt, G. Apelbaum, J. Flury, and P. E. Holmdahl Olsen. Swarm Accelerometer Data Processing from Raw Accelerations to Thermospheric Neutral Densities. *Earth, Planets and Space*, Volume 68, 2016. ISSN 18805981. doi: 10.1186/s40623-016-0474-5.

[45] D. A. Stanica, D. Stanica, J. Blecki, T. Ernst, W. Jozwiak, and J. Slominski. Pre-seismic Geomagnetic and Ionosphere Signatures Related to the Mw5.7 Earthquake Occurred in Vrancea Zone on September 24, 2016. *Acta Geophysica*, Volume 66: pages 167–177, 2018. doi: 10.1007/s11600-018-0115-4.

[46] N. B. Stastny, F. R. Chavez, C. Lin, T. A. Lovell, R. A. Bettinger, and J. Luck. Localized Density/Drag Prediction for Improved Onboard Orbit Propagation. *2009 Advanced Maui Optical and Space Surveillance Technologies Conference*, pages 51–58, 2009.

[47] M. F. Storz, B. R. Bowman, J. I. Branson, S. J. Casali, and W. K. Tobiska. High Accuracy Satellite Drag Model (HASDM). *Advances in Space Research*, Volume 36: pages 2497–2505, 2005. ISSN 02731177. doi: 10.1016/j.asr.2004.02.020.

[48] D. Sussillo and L. F. Abbott. Random Walk Initialization for Training Very Deep Feedforward Networks. *International Conference on Learning Representations*, pages 1–10, 2014. URL https://arxiv.org/abs/1412.6558.

[49] Y. Tang, W. W. Hsieh, B. Tang, and K. Haines. A Neural Network Atmospheric Model for Hybrid Coupled Modelling. *Climate Dynamics*, Volume 17: pages 445–455, 2001. ISSN 14320894. doi: 10.1007/s003820000119.

[50] K. E. Taylor. Summarizing Multiple Aspects of Model Performance in a Single Diagram. *Journal of Geophysical Research: Atmospheres*, Volume 106: pages 7183–7192, 2001. doi: 10.1029/2000JD900719.

[51] S. Varma and R. Simon. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, Volume 7: article 19:1–8, 2006. doi: 10.1186/1471-2105-7-91.

[52] H. P. Warren, J. T. Emmert, and N. A. Crump. Linear Forecasting of the F10.7 Proxy for Solar Activity. *Space Weather*, Volume 15: pages 1039–1051, 2017. ISSN 15427390. doi: 10.1002/2017SW001637.

[53] A. F. Wayne. *Introduction to Statistical Time Series: 2nd Edition*. Wiley, 1995. ISBN 978-0-471-55239-0.

[54] D. R. Weimer, E. K. Sutton, M. G. Mlynczak, and L. A. Hunt. Intercalibration of Neutral Density Measurements for Mapping the Thermosphere. *Journal of Geophysical Research: Space Physics*, Volume 121: pages 5975–5990, 2016. ISSN 21699402. doi: 10.1002/2016JA022691.

[55] G. R. Wilson, D. R. Weimer, J. O. Wise, and F. A. Marcos. Response of the Thermosphere to Joule Heating and Particle Precipitation. *Journal of Geophysical Research: Space Physics*, Volume 111: pages 1–9, 2006. doi: 10.1029/2005JA011274.

[56] P. Yaya, L. Hecker, T. Dudok De Wit, C. Le Fèvre, and S. Bruinsma. Developing New Space Weather Tools: Transitioning Fundamental Science to Operational Prediction Systems Solar Radio Proxies for Improved Satellite Orbit Prediction. *Journal of Space Weather and Space Climate*, Volume 7: article A35, 2017. doi: 10.1051/swsc/2017032.

[57] Y. Yu, J. Koller, S. Zaharia, and V. Jordanova. L* Neural Networks from Different Magnetic Field Models and Their Applicability. *Space Weather*, Volume 10, 2012. ISSN 15427390. doi: 10.1029/2011SW000743.

[58] Y. Zhang and L. J. Paxton. Solar EUV Flux Proxy Using Multifrequency Solar Radio Flux. *Space Weather*, Volume 16: pages 434–441, 2018. doi: 10.1029/2017SW001763.

Appendix A

Atmospheric Properties in Function of Altitude

In this appendix the theory behind the variation of the atmospheric properties in function of altitude is elaborated on since it is not the main goal of this thesis report. For all altitudes below 120 km the U.S. Standard Atmosphere relations are used, which are described in Section A.1. For all altitudes higher than 120 km, one needs to rely on more advanced physical or empirical models. Section A.2 briefly describes the inner workings of the thermosphere model used in this report, namely the Drag Temperature Model (DTM2013).

A.1 U.S. Standard Atmosphere

Below 120 km the U.S. Standard Atmosphere model is used to estimate the temperature and density values with varying altitude. The World Meteorological Organization (WMO) has defined a Standard Atmosphere which is an idealised, steady-state representation of mean annual conditions of Earth's atmosphere from the surface to 1000 km at latitude 45°N, as it is assumed to exist during a period with moderate solar activity. The defining meteorological elements are sea-level temperature and pressure, and a temperature-height profile to 1000 km. The air is assumed to be dry, and at heights sufficiently below 86 km, the atmosphere is assumed to be homogeneously mixed with a relative-volume composition leading to a constant mean molecular weight.

All figures incorporated in this section do contain the atmospheric relations [1] to compute the temperature, pressure and density at altitudes ranging 0 to 86 km (Figure A.1) and at altitudes ranging 86 to 120 km (Figure A.2 and Figure A.3). These relations are a first-order approximation and are only used as a preliminary indication of the variations in atmospheric state at altitudes lower than 120 km.

A.2 DTM2013

The DTM2013 is able to produce the representation of the total density in the altitude range of 120 km to around 1500 km. This is achieved by summing the contributions of the main thermosphere constituents (N_2, O_2, O, He, H), under the hypothesis of independent static diffuse equilibrium. The total density, ρ, at altitude z is calculated as defined by Equation A.1.

[1] Obtained from: `http://www.braeunig.us/space/atmmodel.html`

U.S. Standard Atmosphere, 0 to 86 km

Layer	Geopotential Altitude, h (km')	Molecular-Scale Temperature, T_M (K)	Pressure, P (Pa)
Troposphere	0-11	288.15 − 6.5 × h	101325.0 × [288.15 / (288.15 − 6.5 × h)] $^{(34.1632 / -6.5)}$
Stratosphere	11-20	216.65	22632.06 × EXP[−34.1632 × (h − 11) / 216.65]
	20-32	196.65 + h	5474.889 × [216.65 / (216.65 + (h − 20))] $^{(34.1632)}$
	32-47	139.05 + 2.8 × h	868.0187 × [228.65 / (228.65 + 2.8 × (h − 32))] $^{(34.1632 / 2.8)}$
Mesosphere	47-51	270.65	110.9063 × EXP[−34.1632 × (h − 47) / 270.65]
	51-71	413.45 − 2.8 × h	66.93887 × [270.65 / (270.65 − 2.8 × (h − 51))] $^{(34.1632 / -2.8)}$
	71-84.852	356.65 − 2.0 × h	3.956420 × [214.65 / (214.65 − 2 × (h − 71))] $^{(34.1632 / -2)}$

Density, ρ (kg/m^3) = P/(RT$_M$)
Speed of sound, C (m/s) = (γRT$_M$)$^{1/2}$

Specific gas constant, R = 287.053 J/kg-K
Specific heat ratio, γ = 1.400

Figure A.1: The complete set of equations used to compute atmospheric properties from sea level to a geometric altitude of 86 km. Altitudes are geopotential altitude in this table.

U.S. Standard Atmosphere, 86 to 1000 km

Geocentric Altitude, z (km)	Kinetic Temperature, T (K)
86-91	186.8673
91-110	263.1905 − 76.3232 × SQRT[1 − ((z − 91) / −19.9429)^2]
110-120	240 + 12 × (z − 110)

Figure A.2: Set of equations to compute the kinetic temperature from a geometric altitude of 86 km to 120 km.

$$\rho(z) = \sum^{i} \rho_i(120km) f_i(z) exp(G_i(L)) \tag{A.1}$$

The partial densities ρ_i are specified at 120 km altitude and are propagated to higher altitudes using the height function $f(z)$, which is based on the temperature. G is a spherical harmonic function which models the exospheric temperature and the partial density variations as a function of the environmental parameters L. These include the latitude, local solar time, solar flux and geomagnetic activity. G is also used to incorporate the model periodic and non-periodic variations [*Bruinsma*, 2015].

U.S. Standard Atmosphere, 86 to 1000 km					
Geometric Altitude, z (km)	Basic equation form: EXP(A × z^4 + B × z^3 + C × z^2 + D × z + E)				
	A	B	C	D	E
Pressure, P (Pa)					
86-91	0.000000	2.159582E-06	-4.836957E-04	-0.1425192	13.47530
91-100	0.000000	3.304895E-05	-0.009062730	0.6516698	-11.03037
100-110	0.000000	6.693926E-05	-0.01945388	1.719080	-47.75030
110-120	0.000000	-6.539316E-05	0.02485568	-3.223620	135.9355
120-150	2.283506E-07	-1.343221E-04	0.02999016	-3.055446	113.5764
150-200	1.209434E-08	-9.692458E-06	0.003002041	-0.4523015	19.19151
200-300	8.113942E-10	-9.822568E-07	4.687616E-04	-0.1231710	3.067409
300-500	9.814674E-11	-1.654439E-07	1.148115E-04	-0.05431334	-2.011365
500-750	-7.835161E-11	1.964589E-07	-1.657213E-04	0.04305869	-14.77132
750-1000	2.813255E-11	-1.120689E-07	1.695568E-04	-0.1188941	14.56718
Density, ρ (kg/m^3)					
86-91	0.000000	-3.322622E-06	9.111460E-04	-0.2609971	5.944694
91-100	0.000000	2.873405E-05	-0.008492037	0.6541179	-23.62010
100-110	-1.240774E-05	0.005162063	-0.8048342	55.55996	-1443.338
110-120	0.00000	-8.854164E-05	0.03373254	-4.390837	176.5294

Figure A.3: Tabulated coefficients to compute the pressure and density for geometric altitudes from 86 km to 1000 km.

Appendix B

Conversion Tables ap to Kp

Table B.1: Conversion table for conversion between a_p and K_p provided by the GFZ German Research Centre for Geosciences, and the K_p-index in its original notation and the K_p-index in decimal format. The auroral activity is added as an indicator.

a_p	K_p	**decimal** K_p	**auroral activity**
0	0o	0.00	Quiet
2	0+	0.33	Quiet
3	1-	0.67	Quiet
4	10	1.00	Quiet
5	1+	1.33	Quiet
6	2-	1.67	Quiet
7	2o	2.00	Quiet
9	2+	2.33	Quiet
12	3-	2.67	Unsettled
15	3o	3.00	Unsettled
18	3+	3.33	Unsettled
22	4-	3.67	Active
27	4o	4.00	Active
32	4+	4.33	Active
39	5-	4.67	Minor storm
48	5o	5.00	Minor storm
56	5+	5.33	Minor storm
67	6-	5.67	Moderate storm
80	6o	6.00	Moderate storm
94	6+	6.33	Moderate storm
111	7-	6.67	Strong storm
132	7o	7.00	Strong storm
154	7+	7.33	Strong storm
179	8-	7.67	Severe storm
207	8o	8.00	Severe storm
236	8+	8.33	Severe storm
300	9-	8.67	Severe storm
400	9o	8.00	Extreme storm

Appendix C

Training and Test Data Sets

This appendix elaborates on the performed data analysis to identify the most ideal training/validation and testing data sets. The candidate data sets and the parameters needed to make this decision are presented in Section C.1. Section C.2 graphically presents the solar and geomagnetic properties of these training data sets, while Section C.3 does the same for the testing data sets.

C.1 Overview Monthly Model Density Data Sets

This section describes the solar and geomagnetic properties of all different months in the available Swarm-A data set. Not all monthly data sets in a given year are discussed due to missing POD-derived density values. When a specific month contains this type of missing data, the month is discarded as a potential candidate. The different parameters that are evaluated in Table C.1 are shortly discussed below:

- **Av. Scaled** F_{30} **[sfu]**: Monthly-averaged value of the F_{30} value scaled to $F_{10.7}$. These values are used as DTM2013 input values and are an indicator for the intensity of the solar activity.
- **Max.** K_p **[-]**: Maximum monthly value of the Planetary K-index and are used as an indicator for the presence of geomagnetic storms.
- **Nb.Flags**: The number of flags which are defined by the European Space Agency and are considered to be the anomalous measurement data points at a resampling rate of 30 s.
- log_{10} **operation**: The log_{10} operation is performed on all POD-derived density values to reduce the variance in the given density data set. However, the programming software cannot convert those values where the density value approaches zero and thus causing time gaps in the time series. When this case is present, it is indicated by *No* in the table. Those months shall be avoided as training data set.

Additionally, the months that are picked as training and test data sets are marked by an identifier. The identifier is constructed as follows:

$$\text{AABBBBCC} = \text{AA/BBBB/CC}$$

where AA includes the length of the data sets, namely *ST* for Short Term and *LT* for Long Term data sets. BBBB categorises the solar activitiy that is present throughout the time interval of the data set. Two possible tags are used, namely *High* or *Low*, indicating high or low solar activity, respectively. Finally, CC indicates the presence of a Moderate Storm or higher geomagnetic activity. A Moderate Storm is defined by a K_p value higher than 5.67, as defined in Table B.1.

Table C.1: Data analysis for all available monthly data sets of the Swarm-A satellite. The data is derived from the input and output variables of the DTM2013. The flags are defined by the postprocessing phase executed by the European Space Agency. *Test* data sets span a 3-day period.

Month in *2014*	**Av. Scaled F_{30} [sfu]**	**Max. K_p [-]**	**Nb. Flags**	**log_{10} operation**
03	163.27	4.0	726	yes
04	153.45	5.0	194	yes
06	143.02	6.33	3246	yes
08	138.10	5.33	907	no
09 - STHighS/LTHigh	150.34	6.33	0	yes
10 - LTHigh	106.01	5.0	1645	yes
11 - TestHighNS/LTHigh	168.82	4.67	1294	yes
12 - STHighNS/LTHigh	179.33	5.33	8	yes
Month in *2015*				
01	166.73	6.33	3252	yes
02 - TestLTHigh	157.64	5.33	56	yes
04	140.45	6.0	14	yes
05	128.21	5.67	11	yes
08	113.42	6.33	0	yes
09 - TestHighS	109.48	7.0	0	yes
10	117.89	7.33	1082	yes
11	122.68	6.0	0	yes
12	127.36	6.67	1802	yes
Month in *2016*				
01	117.16	6.0	3242	yes
02	113.53	5.67	0	yes
03	100.02	6.33	2524	yes
04	93.31	5.67	3602	yes
05 - TestLowNS	93.42	6.33	0	yes
06	95.06	6.33	3244	no
07 - LTLow	88.68	5.0	0	yes
08 - LTLow	84.95	5.33	0	yes
09 - LTLow	90.1	6.0	0	yes
10 - STLowS/LTLow	87.13	6.33	0	yes
11	81.05	5.33	542	yes
12	79.64	6.0	0	yes
Month in *2017*				
01	79.45	4.33	722	no
03	72.98	6.33	0	no
04	79.26	5.67	0	no
05	73.62	7.0	0	no
06	79.07	5.0	182	yes
07	76.90	6.0	1084	yes
08	72.33	5.33	0	yes
09 - TestLowS	85.55	8.33	3608	yes
10 - STLowNS	80.84	5.67	0	yes
11	75.58	6.33	0	yes
12	74.19	5.0	724	yes

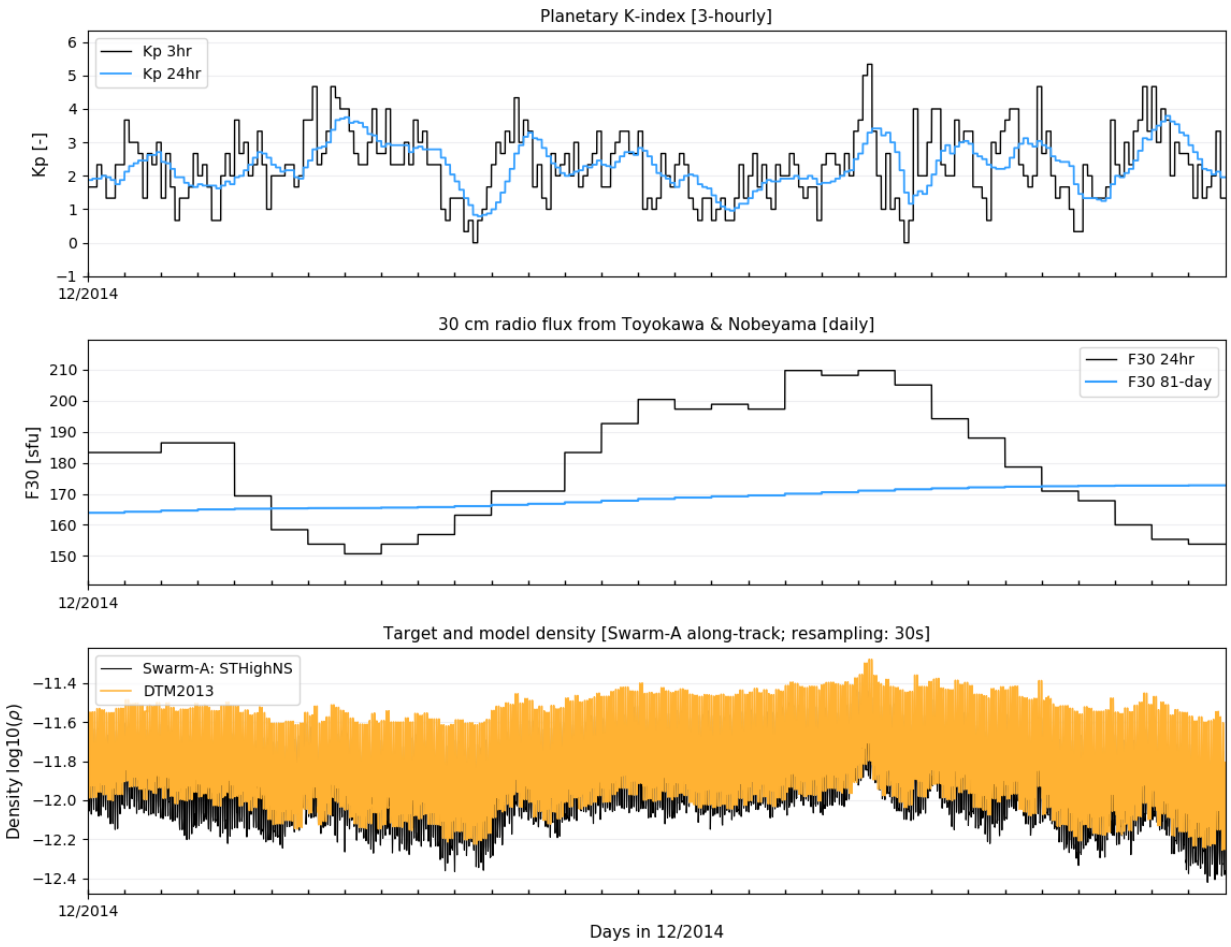


Figure C.1: Training data set with identifier STHighNS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for December 2014 using the DTM2013 empirical model (*bottom*).

C.2 Properties Training Data Sets

This section has the goal to provide a graphical representation of the input values (solar and geomagnetic indices) and output values (POD-derived and modelled along-track density values of Swarm-A) for all training data sets. The STHighNS, STHighS, STLowNS and STLowS training data sets are presented in Figure C.1, Figure C.2, Figure C.3 and Figure C.4, respectively.

C.3 Properties Test Data Sets

This section has the goal to provide a graphical representation of the input values (solar and geomagnetic indices) and output values (POD-derived and modelled along-track density values of Swarm-A) for all testing data sets. One needs to be aware that the solar and geomagnetic indices are the input values of the DTM2013 model hence are not the actual properties on that point in time. As an example, all scaled F_{30} indices are delayed by one day since this is a requirement of the DTM2013. The TestHighNS, TestHighS, TestLowNS and TestLowS test data sets are presented in Figure C.5, Figure C.6, Figure C.7 and Figure C.8, respectively.

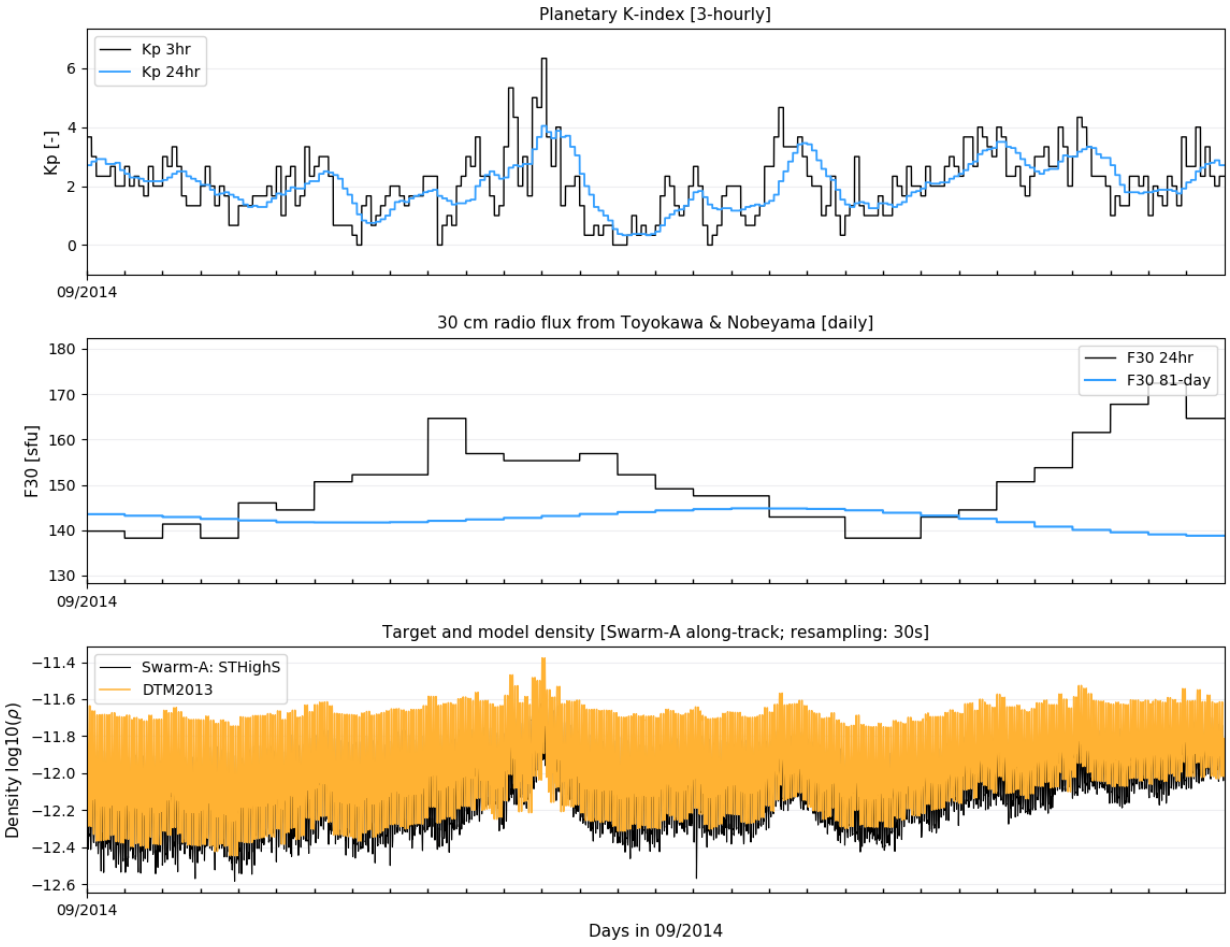


Figure C.2: Training data set with identifier STHighS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for September 2014 using the DTM2013 empirical model (*bottom*).

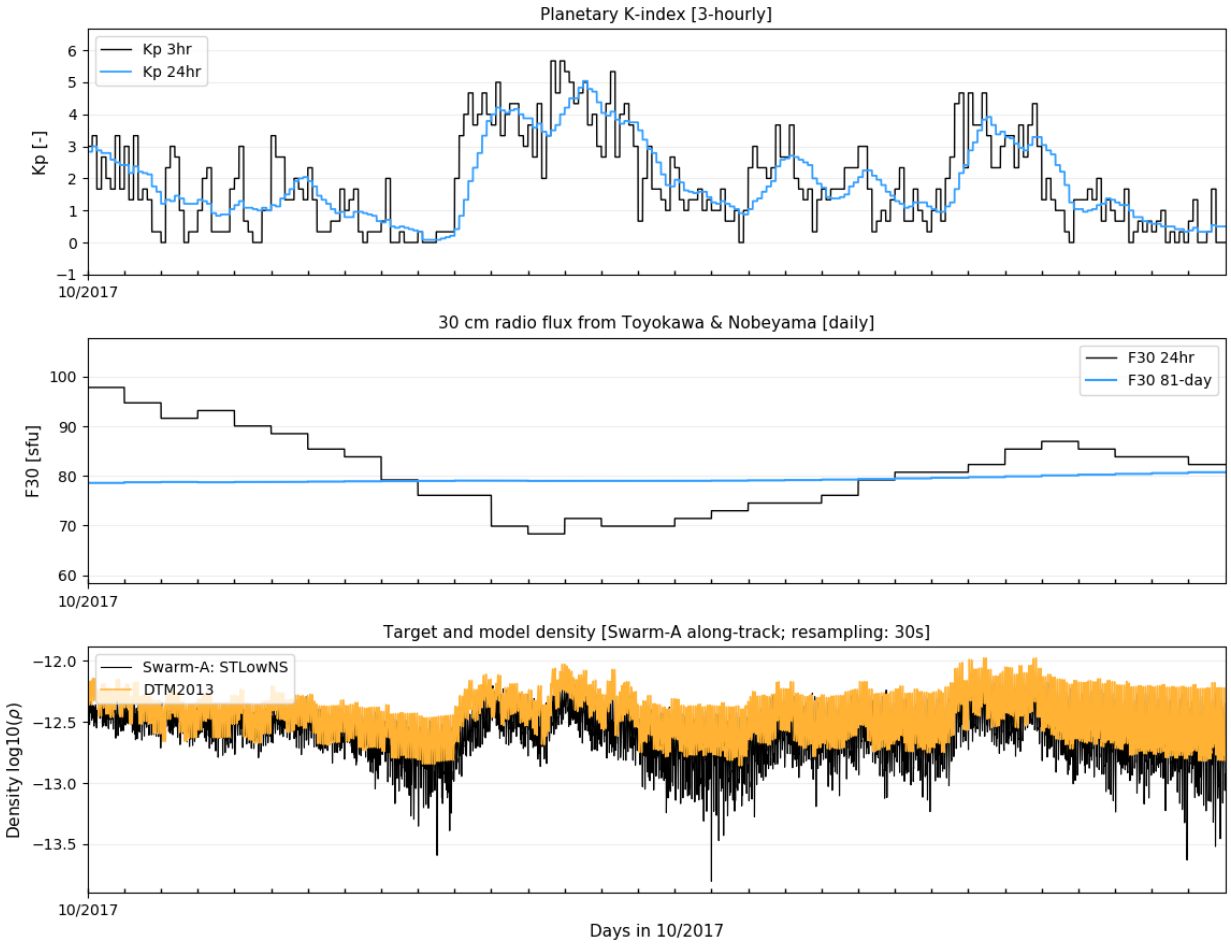


Figure C.3: Training data set with identifier STLowNS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for October 2017 using the DTM2013 empirical model (*bottom*).

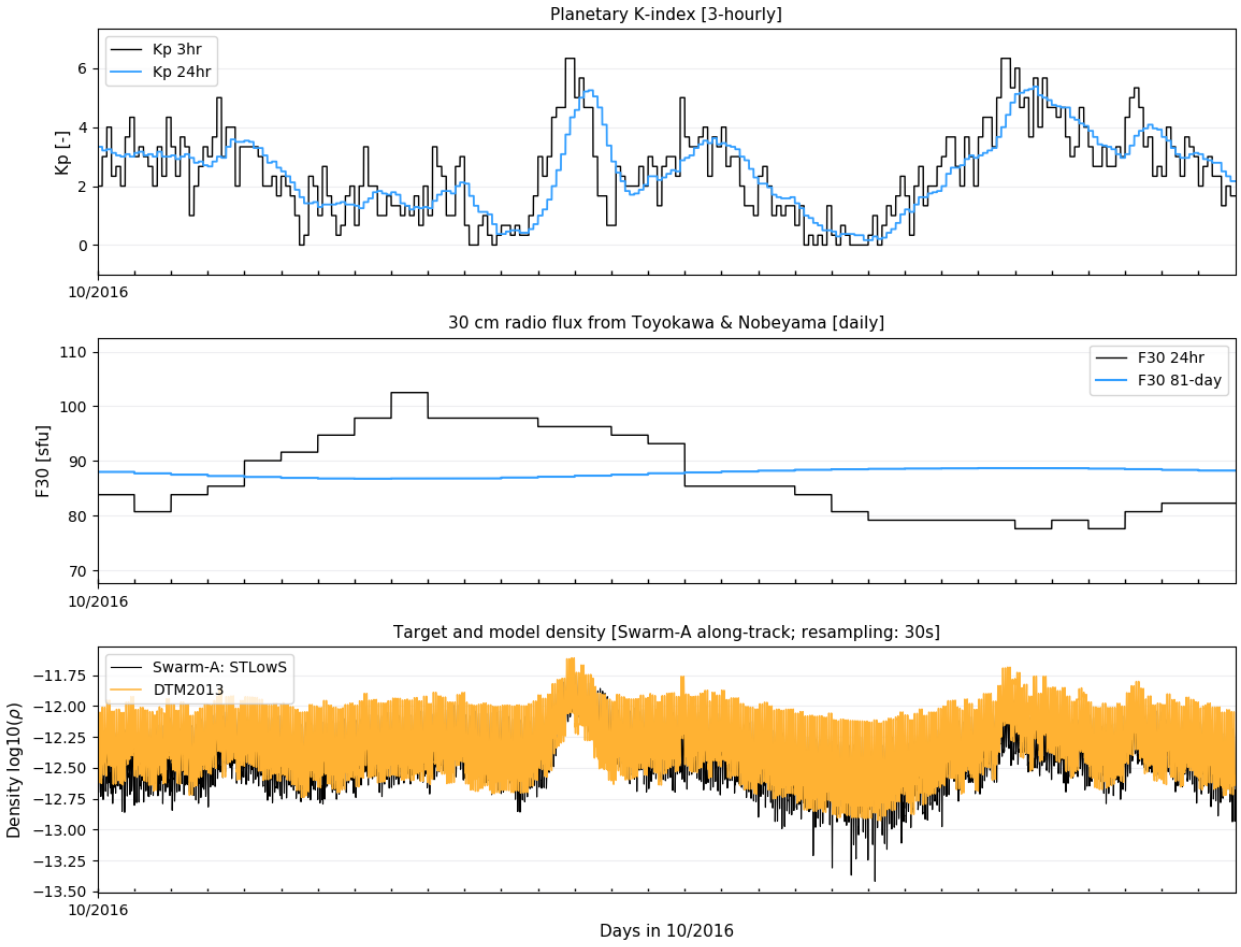


Figure C.4: Training data set with identifier STLowS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for October 2017 using the DTM2013 empirical model (*bottom*).

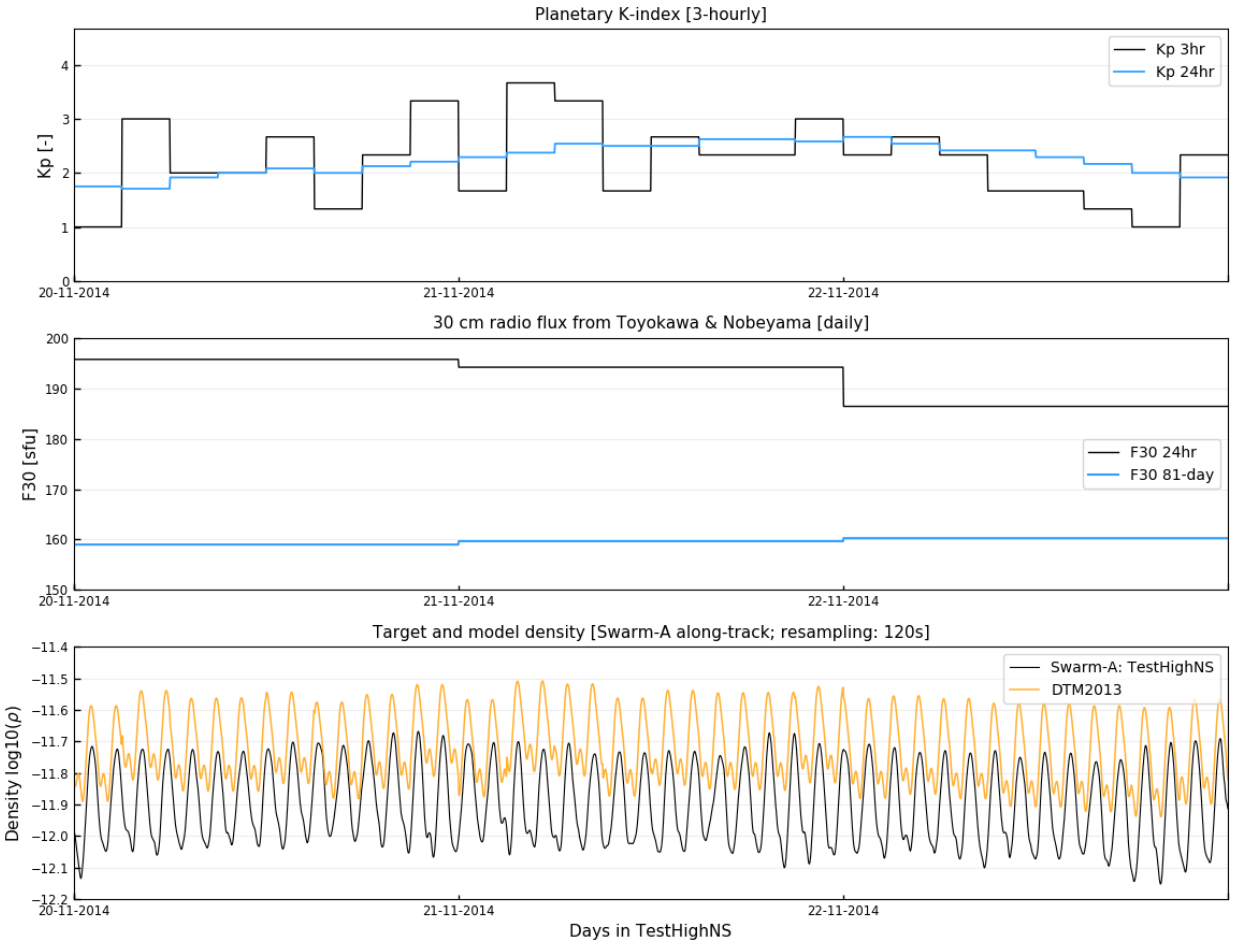


Figure C.5: Test data set with identifier TestHighNS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for 20/11 until 22/11 in 2014 using the DTM2013 empirical model (*bottom*).

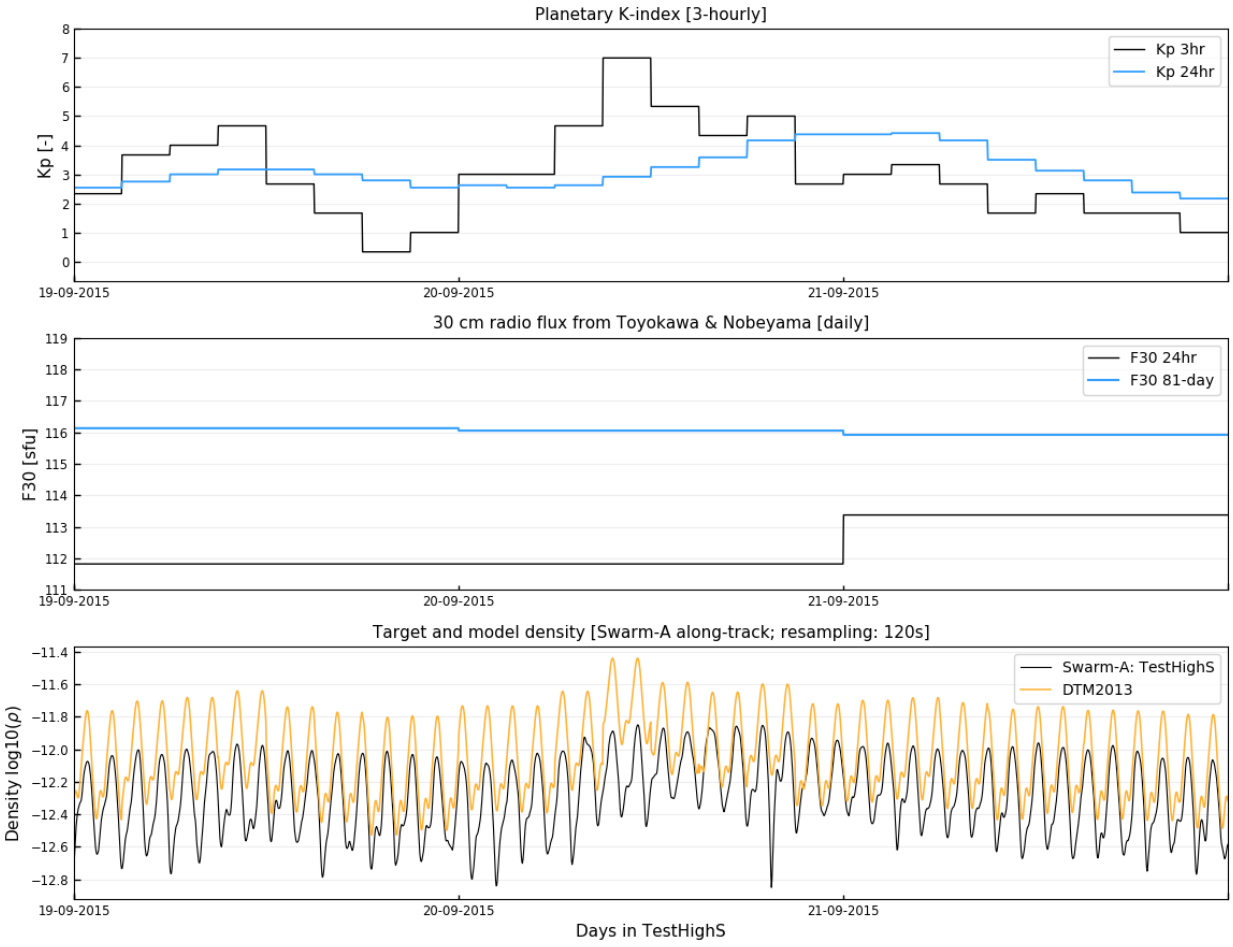


Figure C.6: Test data set with identifier TestHighS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for 19/09 until 21/09 in 2015 using the DTM2013 empirical model (*bottom*).

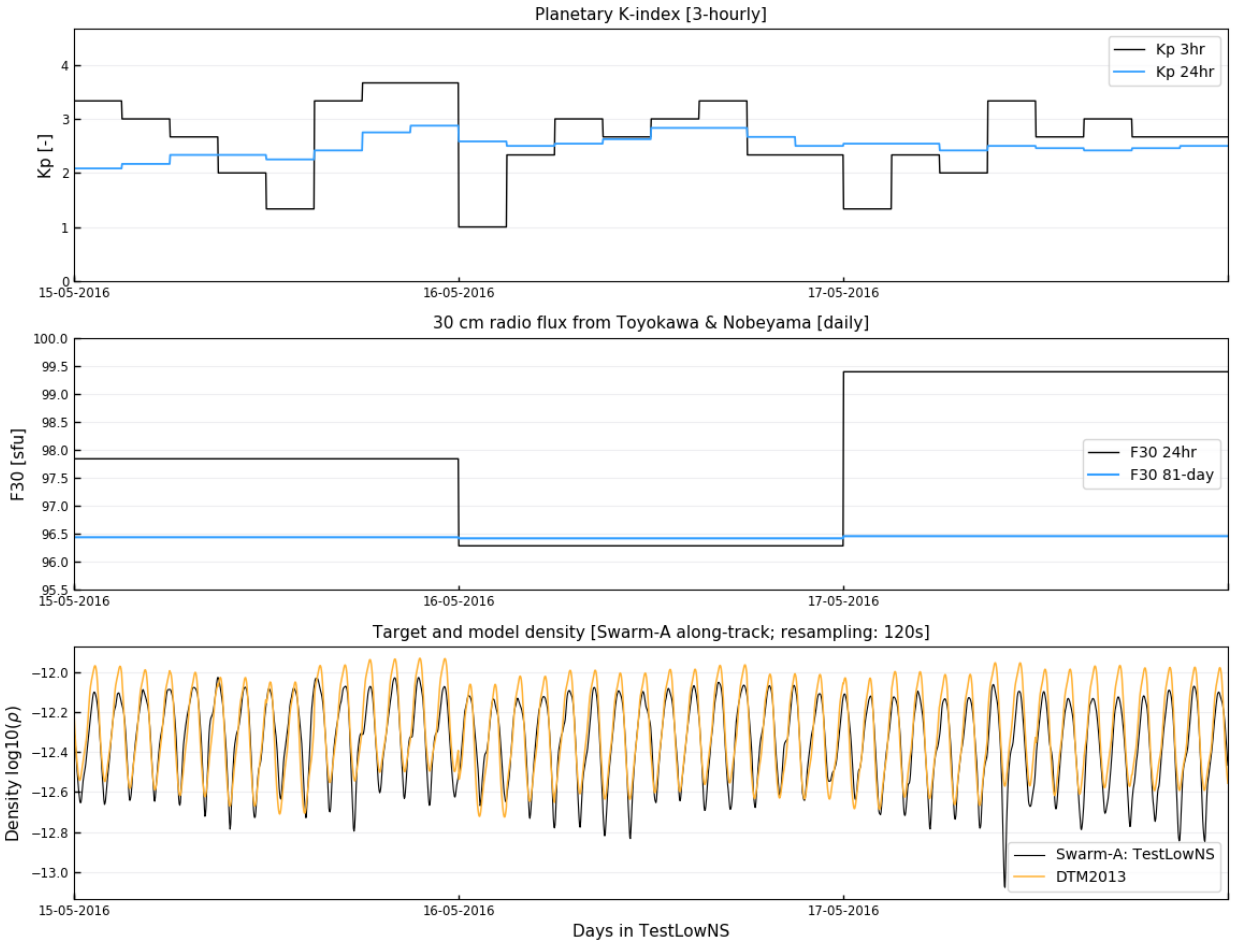


Figure C.7: Test data set with identifier TestLowNS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for 15/05 until 17/05 in 2016 using the DTM2013 empirical model (*bottom*).

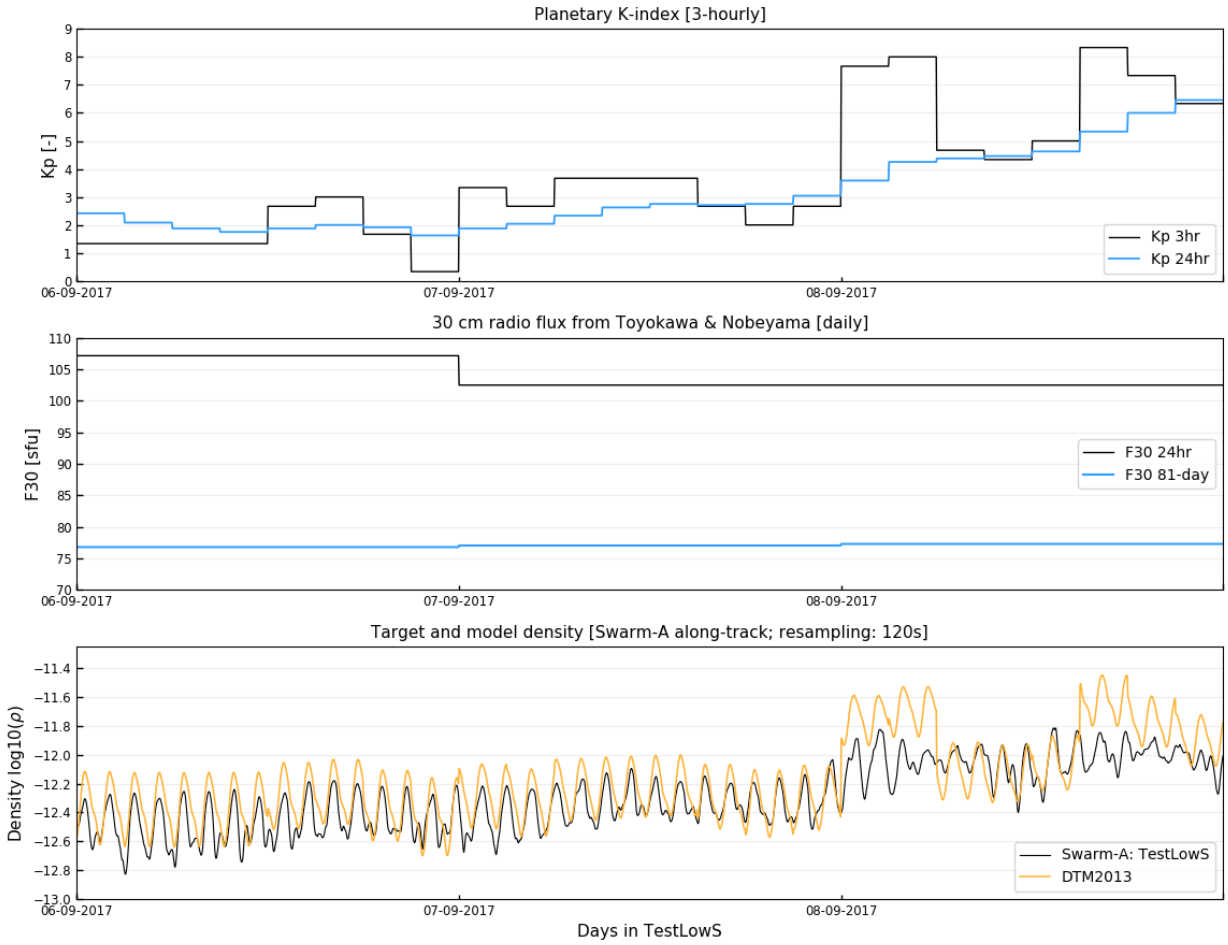


Figure C.8: Test data set with identifier TestLowS. Input solar proxies (*top*) and geomagnetic indices (*middle*). Swarm-A along-track density values for 06/09 until 08/09 in 2017 using the DTM2013 empirical model (*bottom*).