TU WIEN Informatics

A Deep Learning approach for spatiotemporal interpolation of GNSS-R soil moisture retrievals

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Data Science

eingereicht von

Oto Costa Pinho Alves, BSc
Matrikelnummer 12045798

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Wagner

Wien, 27. März 2024

Oto Costa Pinho Alves	Wolfgang Wagner

TU WIEN Informatics

A Deep Learning approach for spatiotemporal interpolation of GNSS-R soil moisture retrievals

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Oto Costa Pinho Alves, BSc
Registration Number 12045798

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Wagner

Vienna, March 27, 2024

Oto Costa Pinho Alves	Wolfgang Wagner

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Oto Costa Pinho Alves, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 27. März 2024

Oto Costa Pinho Alves

Acknowledgements

It is often said that it takes an entire village to raise a child. While this quote has grown into something of a cliché, I believe that it carries a lot of truth, and that it also applies to the writing of a thesis. As no person is really an island, this kind of achievement will always be, in one way or another, the result of a collective effort. I feel incredibly fortunate in being able to say that, very much in the spirit of this work, my "village" covers a vast swath of space and time, spanning many years and spreading out across several continents. In these brief pages, I would like to express my sincere gratitude to some of its inhabitants, all of whom, in their own unique ways, have helped me navigate this journey:

First and foremost, to my mother Marly Costa and my father Marco Pinho Alves, who, beyond being the best parents I could ever ask for, have always given me unconditional and unrelenting support in all of my endeavors, including the mildly crazy ones like moving to another continent amidst the uncertainty of a pandemic. I cannot possibly thank you enough. Amo vocês.

To my supervisor, Professor Wolfgang Wagner, for his invaluable advice, guidance and patience throughout the writing of this thesis. Through him, I would also like to thank all of my colleagues, current and past, at the TU Wien GEO department, with whom I have had the pleasure to work (and party) with since the start of my studies in Vienna.

To Emanuele Santi, Martin Unwin, Gabrielle Marigold and all the science partners of the HydroGNSS project, whose technical insights and fruitful discussions have greatly contributed to the development of this work.

To my extended family, especially to my aunt Bianca Pinho Alves and to her sons Bryan Russell and Nickolas Russell, who might be my cousins on paper, but in practice have always been the brothers that I never had.

And also to the family that I chose, and that in many ways also chose me: friends that I have been lucky to stumble upon on the many crossroads of life, and that have been there for the good times, the not-so-good times and everything in between. Thank you for all the laughter, for the deep and the not-so-deep conversations into the unholy hours of the night, for the memery, and ultimately, for helping me stay sane. Even though I would need dozens of pages to fit all of the praise that each and every one of you deserve, that won't stop me from throwing a shout-out to all of you who, directly or indirectly, have

contributed to the fruition of this thesis (in alphabetical order so that no one ends up at each other's throats): Alice Ziemer, Ana Beatriz Hrovat, Anne Valvezan, Bruna Osti, Fernanda Jófili, Fernando Trajano, Heric Gusso, Justin Lui, Julio León, Lía Schwarz, Luísa Borba, Lukas Mölschl, Michele Santos, Pedro Esperidião, Rafael Acúrcio, Rafael Bérgamo, Rafaella Küper, Ricardo Schmitz, Selma Osmanagić, Thais Beham, Vlada Reshetilo and Lord[1] Vinícius Barros, Protector of Cambridge[2]. The completion of this work would not have been possible without the continued support from all of you.

To the University of Pernambuco, where I have conducted my bachelors studies, and by extension, to the entire community of the public education system of Brazil, which has been continuously and heroically resisting against so many efforts to undermine it over the last decade. Particularly, to Professor Fernando Buarque, who first introduced me to scientific research.

And finally, to my late grandmother Adalcina Souza Dias, whose memory I hold dear. Although you could not be here to witness its completion, your enduring influence was a guiding light through every step of this journey.

[1] Self-proclaimed
[2] Self-proclaimed

Kurzfassung

In den letzten Jahren hat sich die Global Navigation Satellite System-Reflectometry (GNSS-R) als vielversprechende Technologie für die Fernerkundung der Bodenfeuchte herauskristallisiert, die einen kostengünstigen, aber noch sehr experimentellen Ansatz bietet. Ein wesentliches Problem der von GNSS-R abgeleiteten Bodenfeuchtedatenprodukte sind die erheblichen räumlich-zeitlichen Datenlücken, die für verschiedene Anwendungen problematisch sein können. Diese Lücken können durch die Verwendung spatiotemporaler Interpolationsalgorithmen gemildert oder sogar beseitigt werden. Der derzeit modernste spatiotemporale Interpolationsalgorithmus für GNSS-R-Bodenfeuchtedaten, die Previously-Observed Behavior Interpolation (POBI), beruht auf dem Training eines großen Ensembles von ortsspezifischen Regressionsmodellen, was zu Ineffizienzen bei der Informationskodierung und Parameterspeicherung führt. Um diese Einschränkung zu beheben, schlagen wir Deep Convolutional Spatiotemporal Interpolation (DCSTI) vor, eine neuartige Lösung für das Problem der spatiotemporalen Interpolation von GNSS-R Bodenfeuchtedaten. DCSTI nutzt Deep Learning, um ein einziges Regressionsmodell zu trainieren, das sowohl die allgemeinen als auch die regionsspezifischen spatiotemporalen Bodenfeuchtemuster lernen kann und somit an mehreren Orten anwendbar ist. Um diese Lösung zu bewerten, führen wir vergleichende Experimente durch, bei denen sowohl POBI als auch DCSTI verwendet werden, um spatiotemporale Lücken in Bodenfeuchtigkeitsdaten zu füllen, die von den CYGNSS-Satelliten der NASA gesammelt wurden. Unsere Ergebnisse zeigen, dass DCSTI in der Lage ist, Interpolationsfehler zu erreichen, die mit denen von POBI vergleichbar sind und dabei wesentlich weniger Parameter benötigt. Mit diesen Ergebnissen versuchen wir, einen neuen Ansatz für die Verwendung von Deep Learning bei der räumlich-zeitlichen Interpolation von GNSS-R-Daten zu entwickeln. Insbesondere erwarten wir, dass dieser Ansatz signifikante Vorteile bei der Analyse von Bodenfeuchtedaten aus zukünftigen GNSS-R Missionen wie ESA's HydroGNSS bietet, indem er Transfer-Learning-Techniken nutzt, um die Qualität zukünftiger Datenprodukte zu verbessern.

Abstract

In recent years, Global Navigation Satellite System-Reflectometry (GNSS-R) has emerged as a promising technology for remotely sensing soil moisture, offering a cost-effective, yet still very experimental approach. A significant challenge posed by GNSS-R-derived soil moisture data products is the presence of significant spatiotemporal data gaps, which can be problematic for various applications. These gaps can be mitigated or even eliminated through the usage spatiotemporal interpolation algorithms. The current state-of-the-art spatiotemporal interpolation algorithm for GNSS-R soil moisture data, Previously-Observed Behavior Interpolation (POBI), relies on the training of a large ensemble of location-specific regression models, which leads to inefficiencies in information encoding and parameter storage. To address this limitation, we propose Deep Convolutional Spatiotemporal Interpolation (DCSTI), a novel solution for the problem of spatiotemporal interpolation of GNSS-R soil moisture data. DCSTI employs deep learning to train a single regression model that is capable of learning both the overall and the region-specific spatiotemporal soil moisture patterns, thus being applicable across multiple locations. In order to evaluate this solution, we conduct comparative experiments where both POBI and DCSTI are used to fill spatiotemporal gaps in soil moisture data collected by NASA's CYGNSS satellites. Our results indicate that DCSTI is capable of achieving interpolation error levels comparable to those offered by POBI, while also demanding substantially fewer parameters. With these findings, we seek to establish a new framework for the usage of deep learning in spatiotemporal interpolation of GNSS-R data. In particular, we anticipate that this framework will offer significant benefits in the analysis of soil moisture data from upcoming GNSS-R missions such as ESA's HydroGNSS, leveraging transfer learning techniques to enhance the quality of future data products.

Contents

CHAPTER 1

Introduction

1.1 Motivation

As one of the 55 Essential Climate Variables (ECVs) currently listed by the World Meteorological Organization's Global Climate Observing System (GCOS) [BA12], Soil Moisture (SM) is known to play a critical role in a myriad of natural and human-driven phenomena within the Earth system. For fields like geography, hydrology, climatology and other Earth sciences, the understanding of soil moisture dynamics can be seen both as an end goal and as a valuable component in the modelling of related phenomena. For fields like meteorology, agronomy and disaster management, knowledge about soil moisture can deliver significant practical value to forecasting, planning and decision making [LML+11] [SCD+10].

Thus, the ability to collect accurate, reliable and timely measurements of soil moisture is relevant for a multitude of human endeavors, especially in times of accelerating climate change. For many applications, it is also of great importance that the soil moisture measurements are collected at a sampling rate that is sufficiently dense across both the spatial and temporal domains, in a way that all the potentially relevant physical dynamics are successfully captured. Moreover, several applications also require soil moisture data products that provide an extensive spatial coverage, sometimes encompassing the entire globe.

The notions of spatial and temporal distributions of the measurements within a soil moisture data product are often aggregated, somewhat loosely, within the concept of a product's spatiotemporal resolution. Different methods of soil moisture data acquisition have different strengths and weaknesses in terms of spatiotemporal resolution, and choosing between methods usually involves a trade-off between spatial and temporal sampling rates.

The most traditional technique for obtaining soil moisture measurements involves using in-situ sensors [BSJ+19], which are sensors that must be directly installed in the locations of interest and are able to measure the moisture of the soil directly adjacent to them, with a very dense temporal sampling rate, but under a very limited spatial distribution.

Since the turn of the century, Remote Sensing (RS) systems have also become an increasingly valuable tool for acquiring soil moisture data [dD16]. Those systems make use of electromagnetic sensors embarked on Earth observation (EO) satellites and, with the help of complex data processing and physical modelling pipelines, are able to "retrieve" SM measurements at a distance, achieving a dense spatial distribution and a wide or even global spatial coverage. This, however, comes the cost of a coarser sampling rate across the time domain due to the long revisit intervals at each location. Of particular relevance for remote sensing of soil moisture are the microwave sensors that are embarked on satellite missions such as SMAP [ENO+10], SMOS [KWW+10] and ASCAT [WHK+13], all of which have already yielded well-developed and mature SM data products with a multitude of users in the scientific community, as well as in the private and public sectors.

Over the last decade, there has also been a growing interest in usage of Global Navigation Satellite System-Reflectometry (GNSS-R) systems for the retrieval of SM measurements, such as NASA's CYGNSS mission [RUD+13] and ESA's upcoming HydroGNSS mission [UPC+21]. Unlike the traditional active microwave remote sensing satellites, GNSS-R sensors work by "opportunistically" capturing signals which are emitted by Global Navigation Satellite System (GNSS) satellites and then reflected by the Earth. In other words, they are active remote sensing systems in which the multiple transmitters and receivers are embarked on different satellites, in what is called a multistatic setup. This lack of a built-in transmitter makes GNSS-R satellites relatively cheap to build and launch, and thus allows for larger satellite constellations to be implemented. This, in turn, leads to shorter revisit times and denser temporal sampling rates

While GNSS-R seems like a promising alternative for the design of future soil moisture remote sensing missions, especially from a cost-benefit perspective, it is still considered a very incipient and experimental technology. Further studies are required in order to better understand the physical dynamics of GNSS-R-based soil moisture retrieval and improve the quality and maturity of the data products derived from it.

Furthermore, the amount of soil moisture data derived from GNSS-R sources remains relatively scarce. As of 2023, there is still just a limited amount of GNSS-R satellites in operation, and most of those have spatiotemporal issues such as a limited geographic coverage or a relatively short data record up to the present time. Therefore, the field of GNSS-R-based remote sensing of soil moisture could also benefit from studies that seek to understand how to best leverage and exploit the limited amount of data that is currently available.

1.2 Problem Statement and Aim of the Thesis

One of the most prevalent issues one faces when working with GNSS-R SM data products is that of data sparsity. These products are usually provided in a "Datacube" format. In general terms, a datacube is a multidimensional data array with dimensions that carry spatial or temporal semantics. GNSS-R SM datacubes, especially those products derived from small satellite constellations, might be considered too sparse for some applications, with measurements distributed too sparsely across time and/or space. For instance, in Figure 1.1, we show a temporal slice of a GNSS-R SM datacube derived from CYGNSS, containing SM measurements collected globally over an entire day and projected onto a map for better visibility. For applications that require daily or even sub-daily sampling of soil moisture at specific locations, this coverage is clearly insufficient.

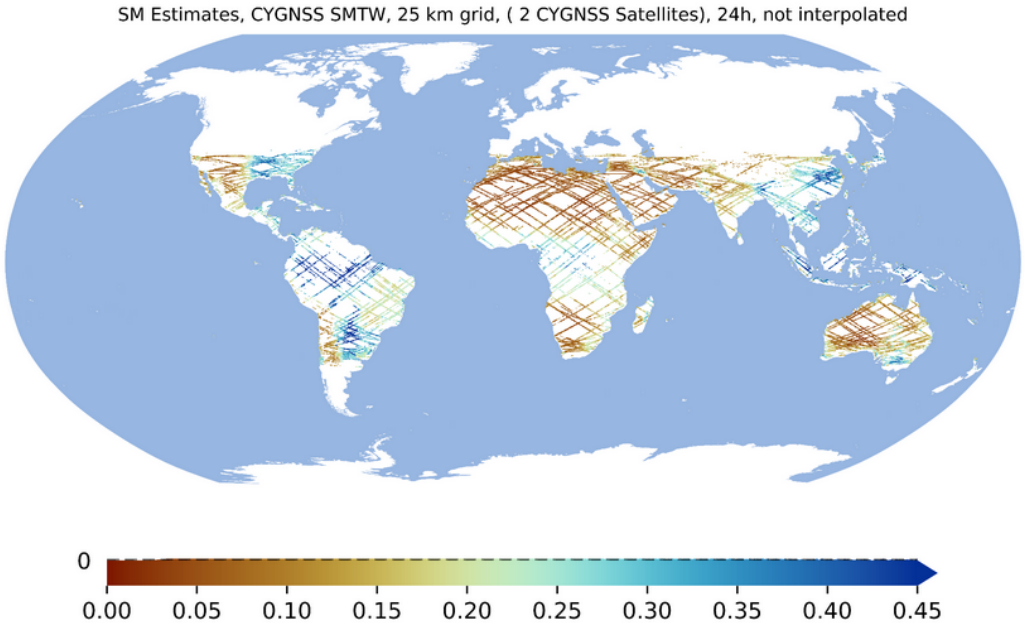


Figure 1.1: A temporal slice of a GNSS-R Soil Moisture datacube derived from the Trackwise version of the IFAC ANN CYGNSS SM data product, containing soil moisture retrievals collected over an entire day and binned into a 25km x 25km grid, and then projected into a map using to facilitate visual understanding.

The most straightforward approach to tackle the sparsity of GNSS-R SM datacubes would be to develop and launch larger constellations of satellites, which would in turn lead to a denser spatial coverage and shorter location revisit times, i.e. a denser temporal sampling rate. Naturally, that is often impractical or even infeasible due to budgetary reasons, even within the context of GNSS-R systems. A more cost-effective solution for mitigating datacube sparsity issues is employing Spatiotemporal Interpolation (STI) techniques, which seek to use the known measurements to estimate the values in the

empty datacube cells. This allows us to obtain less sparse datacubes, at the cost of an increase in the overall uncertainty of the SM value they contain.

The spatiotemporal interpolation of GNSS-R soil moisture datacubes can be framed as a regression problem, in which the target variable are soil moisture values of the cells with missing data. The predictors are soil moisture values contained in the nonempty cells in a volume around the cell whose value we want to estimate, also known as the cell's spatiotemporal neighborhood. In addition to those, the vector of predictors can also be enriched with ancillary data describing spatiotemporal information about the cell, such as its latitude and longitude coordinates, or even information derived from external datasets such as land cover or vegetation density.

The current state-of-the-art for the spatiotemporal interpolation of GNSS-R data is the Previously-Observed Behavior Interpolation (POBI) algorithm [Che23], which has been used to interpolate reflectivity data collected by the GNSS-R sensors aboard the CYGNSS constellation of satellites. It does so by training a large collection of highly localized regressors which encode how each the reflectivity of each specific pixel along the spatial grid behaves with respect to the reflectivity observed on its spatiotemporal neighborhood.

While POBI has shown great success in interpolating CYGNSS reflectivity measurements, its reliance on pixel-specific models requires the storage of a large number of parameters, which occupies terabytes of disk space and creates a significant overhead at prediction time, both in terms of I/O and computational costs. Moreover, if the calibration period is changed or extended, it is necessary to retrain the POBI regressors from scratch on the new period.

Considering that every regressor trained by POBI is essentially modeling the same physical quantity, it should follow that, even if local factors like land cover and soil type have a significant impact on the models, there should be some level of redundancy among the dynamics that are captured across all local models. Thus, one possible path for improving POBI would be exploiting these redundancies to create a single global regressor which, with the help of ancillary data related to a cell's location and timestamp, is able to achieve a satisfactory performance while reducing storage space requirements and decreasing overhead at prediction time. Given their intrinsic capability to capture complex nonlinear patterns on high-dimensional data with spatially correlated features and compress them into a single model, artificial neural networks emerge as a natural candidate for this task.

In this thesis, we seek to address the problem of spatiotemporal interpolation of GNSS-R soil moisture retrievals by developing a novel, deep-learning-based regression algorithm which is able to generate unified, portable models that can be used across the entire datacube, taking advantage of a neural network's natural aptitude for compression. Specifically, we aim to apply this method to datacubes derived from the "Trackwise" version of the IFAC CYGNSS ANN SM data product [SPP+20] [SCC+22], which is retrieved from reflectivity data collected by the GNSS-R sensors mounted on the CYGNSS

constellation of satellites. In order to evaluate this new method, which we call Deep Convolutional Spatiotemporal Interpolation (DCSTI), we benchmark it against POBI applied to the same dataset. Additionally, we also evaluate the interpolated datacubes that it produces by validating them against external SM data products derived from independent sources.

1.3 Context of the Thesis: HydroGNSS

The experiments described within this thesis have been conducted as part of the development phase of the HydroGNSS mission. HydroGNSS is a new GNSS-R satellite constellation selected by the European Space Agency (ESA) as part of the Scout Program, which seeks to finance the development of low-budget Earth observation missions that can provide innovative scientific insights in an agile manner. Unlike the previous GNSS-R satellite missions such as TechDemoSat-1 (TDS-1) and CYGNSS, which are sensitive to soil moisture but were initially developed to measure observables over the oceans and cryosphere, HydroGNSS is being designed with the primary intention of measuring soil moisture and other land-based environmental variables. [UPC+21]

The development of HydroGNSS is being led by Surrey Satellite Technology (SSTL), a private aerospace conglomerate based in the United Kingdom, along with a team of science partners from universities and research institutions of several countries in Europe which includes the Remote Sensing research group of TU Wien's Department of Geodesy and Geoinformation.

HydroGNSS is currently planned to be launched in late 2024. Consequently, there is still no real data from that satellite which can be used for our experiments on spatiotemporal interpolation. Due to the multiple similarities between both instruments, our experiments will be performed with a SM data product derived from CYGNSS measurements.

While HydroGNSS provides several innovations over CYGNSS, like the coverage of the extratropical latitudes, the budget constraints imposed by the Scout program requirements have limited the constellation to the inclusion of only 2 satellites, contrasting with CYGNSS' grand total of 8. Thus, in order to emulate a HydroGNSS-like scenario, our experiments are focused on performing spatiotemporal interpolation of data collected by a subset of the CYGNSS constellation comprised of only 2 satellites. The fact that CYGNSS has a higher total of satellites also gives us the advantage that we can use the data extracted from the full CYGNSS constellation as ground truth values for training spatiotemporal interpolation models and for an initial validation of their outputs, not having to concern ourselves with inter-sensor biases that would emerge if using ground truths derived from other satellites.

Thus, even though we are only working with CYGNSS data, we expect that the experiments described within this thesis can pave the way for a spatiotemporal interpolation framework that can also be adapted to HydroGNSS-derived data once it becomes available.

1.4 Contribution

In this thesis we present DCSTI, a novel approach for addressing the problem of spatiotemporal interpolation of GNSS-R-derived soil moisture data. DCSTI leverages deep learning techniques to produce a unified regression model that is able to efficiently capture both the overall spatiotemporal dynamics of soil moisture as well as location-specific patterns. We then design and conduct an evaluation experiment that benchmarks DCSTI against the state-of-the-art solution for spatiotemporal interpolation of GNSS-R-derived soil moisture data, the POBI algorithm.

From an analysis of our experiment results, which are described in more detail in Chapter 7, we have concluded that DCSTI demonstrates an interpolation performance comparable to that of POBI. Moreover, it also effectively mitigates one of POBI's main drawbacks, the need for a substantial amount of parameters. Specifically, DCSTI was able to match POBI's performance while requiring a number of parameters that is smaller by one or two orders of magnitude, depending on which dataset was in use. We expect this difference in model size to become even more pronounced if a similar experiment is conducted using datasets with higher spatial resolutions.

1.5 Thesis Outline

This thesis is organized as follows. Chapters 1 through 4 are predominantly concerned with the theoretical backdrop of the research, covering multiple points related to spatiotemporal interpolation of GNSS-R SM data and deep learning. In Chapter 1, we provide an overview of the scope and objectives of this work. Chapter 2 explores a theoretical background about remote sensing of soil moisture, GNSS-R systems and the CYGNSS constellation. Chapter 3 focuses on the specific issue of spatiotemporal interpolation of GNSS-R SM data and introduces POBI, the state-of-the-art algorithm for addressing this problem. Chapter 4 delves into the foundations of neural networks and deep learning.

Chapters 5 through 7 focus on the practical aspects of this work, introducing a novel solution to address the problem of spatiotemporal interpolation of GNSS-R SM data, and describing the conduction of an experiment designed to evaluate the performance of this solution. In Chapter 5, we introduce the DCSTI algorithm, which leverages deep learning techniques to address the spatiotemporal interpolation of GNSS-R SM data. Chapter 6 outlines the design of an experiment which seeks to evaluate DCSTI, comparing its performance with that of POBI. In Chapter 7, we discuss the results of this experiment. Finally, Chapter 8 presents the concluding remarks of the thesis.

CHAPTER 2

Remote Sensing of Soil Moisture

2.1 Soil Moisture

In a general sense, Soil Moisture (SM), also known as the soil's water content, refers to the amount of water that is present within the outer layers of the soil. As an environmental variable, it plays an important role in a wide variety of processes, both natural, human-driven, or a mixture of both.

For descriptive sciences dealing with physical phenomena within the Earth system, such as physical geography, hydrology and climatology, the understanding of the soil moisture spatiotemporal dynamics themselves can naturally be seen as an end goal. Furthermore, this knowledge can also be a valuable component in the understanding and modelling of some of their other objects of study, as soil moisture is known to be heavily involved in the Earth's water, energy and carbon cycles [SCD+10].

Soil moisture is also a known driver of short-term weather patterns [BCM+14], which makes the near-real-time monitoring of soil moisture conditions a valuable tool for weather forecasting and other activities within the scope of meteorology. As a consequence, knowledge about soil moisture dynamics is also beneficial in fields like agronomy and natural disaster management, where the planning and decision-making process heavily relies on climatic and meteorological data.

Last, but not least, there is a strong correlation between soil moisture dynamics and long-term climatic patterns. In times of exacerbated climate change, with increases in greenhouse gas concentrations and with adverse events like droughts, floods and heatwaves becoming more frequent, it is becoming more difficult to draw reliable inferences from the climate data records from the previous centuries [SCD+10]. Thus, the extensive monitoring of environmental variables like soil moisture can be extremely beneficial for governments, supranational institutions, and other high-level actors who are required to make informed decisions under increasing uncertainty. With this in mind, the World

Meteorological Organization's Global Climate Observing System (GCOS) has decided to include Soil Moisture in its list of Essential Climate Variables (ECVs) since 2010 [BA12].

The two main sources of soil moisture data are in-situ sensors and satellite observations. In-situ sensors are devices that must be placed in direct contact with the soil sample of interest, and are able to provide SM data at a very dense temporal sampling rate and at a variety of soil depths, at the cost of a very sparse spatial coverage (see Section 2.2). Data obtained through satellite observations comes with a much denser spatial distribution and an extensive spatial coverage, sometimes spanning the entire globe. However, due to limitations imposed by the satellites' orbital periods, the observations are very coarsely distributed across time.

2.1.1 Quantifying Soil Moisture

Strictly speaking, there are multiple ways to define soil moisture, and multiple metrics that can be used to quantify it. The most straightforward metric used for this purpose is the volumetric soil moisture (θ_v), which simply indicates the percentage of water that is contained within a fixed volume of soil (Equation 2.1). In SI units, volumetric soil moisture is measured in terms of m^3m^{-3}, with the numerator corresponding exclusively to the volume of water and the denominator corresponding to the total volume of soil, including the water. While this formulation theoretically allows volumetric soil moisture to fluctuate between 0 and 1, the real-world measurements typically fall between 0.03 and 0.6 m^3m^{-3}.

$$\theta_v = \frac{V_{\text{water}}}{V_{\text{total}}} \tag{2.1}$$

For some applications, rather than measuring the absolute percentage of water contained in the composition of a soil sample, it is more useful to quantify how close the soil is to saturation, i.e. how close it is to having reached its maximum capacity for water retention. In order to do this, one can analyse the degree of saturation (m_s), which measures how much of the space within the soil's pores is occupied by water. The degree of saturation is thus a function of both the volumetric soil moisture and the porosity level of the soil under analysis, and can be computed by Equation 2.2, where θ_v is the volumetric soil moisture in m^3m^{-3} and ϕ is the porosity rate, also given in m^3m^{-3}. [WHK+13].

$$m_s = \frac{\theta_v}{\phi} \tag{2.2}$$

Soil moisture can be measured at different soil depths, with each soil layer behaving differently with respect to its spatiotemporal SM dynamics. In general, the outermost layers of the soil tend to have very fast SM temporal dynamics, being highly susceptible to rainfall events. The deeper layers, on the other hand, only interact with rainfall indirectly, with the water flowing down from the outer layers. Because of this, the deeper

layers are effectively integrating the SM fluctuations from the outer layers over time, and thus exhibit a slower change rate in their SM dynamics [WLR99]. Most users of SM data, especially within the fields of meteorology, climatology and agronomy, are mainly interested in data collected at a depth ranging between 0 and 100 cm, which is known as root-zone soil moisture.

2.2 In-Situ Sensing of Soil Moisture

The most traditional way to obtain soil moisture measurements is by using devices known as in-situ sensors. These are devices that must be installed in direct contact with the soil sample which they seek to measure, and are able to provide accurate and precise soil moisture measurements at a very dense temporal sampling rate. [BSJ+19]

The main drawback of in-situ sensors is that they are limited to providing data about the exact location in which they were installed, which, for spatial distribution purposes, is referred to as a point scale. Multiple in-situ sensors might be required to reliably monitor an area of interest, even if it is a small agricultural field, as using a single sensor could bias the monitoring towards very localised phenomena happening in its vicinity. On a regional or even global scale, it becomes economically infeasible to have an extensive coverage of in-situ sensors, as they require substantial installation and maintenance costs, especially in hard to access areas. Thus, while this method can be very effective for monitoring specific locations of interest, particularly for farming purposes, it is not suitable for applications that require soil moisture data with a dense spatial distribution and a spatial coverage that spans large areas, such as climatology studies.

The spatial resolution coverage limitations that the usage of in-situ sensors imposes upon large-scale soil moisture monitoring efforts can be mitigated by using earth observation satellites and remote sensing techniques, which will be expanded upon in the next section.

2.3 Remote Sensing of Soil Moisture

In a strict sense, remote sensing refers to any activity that involves collecting data about an object of interest, known as the observable, from a long distance and without physical contact. Within the context of Earth sciences, the phrase “Remote Sensing” almost always refers to the usage of electromagnetic radiation sensors embarked on aircraft or Earth observation satellites to collect information about observables located on Earth.

As shown in Figure 2.1 remote sensing system can be considered either passive or active, depending on the original source of the sensed signals. Passive systems do not require a transmitter, receiving signals either emitted or reflected by their targets. In the case of EO applications, this often consists of measuring electromagnetic radiation that was originally emitted by the Sun and then reflected by objects on Earth. Active systems, on the other hand, involve both a transmitter and a receiver. They use the transmitter to

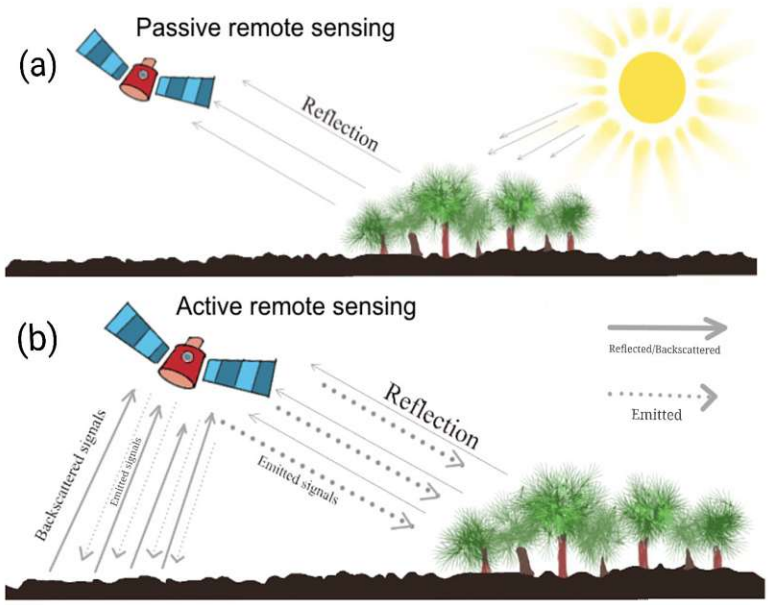


Figure 2.1: Simplified diagram of active and passive remote sensing systems used for Earth observation. From [JASC23]

emit their own electromagnetic radiation in order to "illuminate" the target, and then measure the signal that is reflected towards the receiver.

Both passive and active remote sensing systems are widely used for remote sensing of environmental variables. In the case of soil moisture, a particular emphasis can be placed on sensors that capture electromagnetic radiation in frequency bands between 0.3 and 300 GHz, known as the microwave domain. While there are other types of signals that can be used for this purpose, such as optical sensors, Microwave Remote Sensing (MRS) has an advantage in the fact that microwave signals in certain frequency bands are much less attenuated by precipitation or cloud cover, effectively giving the corresponding microwave sensors the ability to "see through" unfavourable weather conditions. While this is a valuable characteristic in its own, it is particularly useful for monitoring observables whose physical dynamics are typically intertwined with rainfall events, such as soil moisture and floods. Moreover, unlike optical sensors, microwave sensors do not depend on illumination from sunlight, and are thus not restricted to daytime observations. Additionally, they are also able to penetrate vegetation canopies to a certain degree.

Remote sensors aboard EO satellites offer a very extensive spatial coverage, often encompassing the entire globe. The spatial resolution they are able to provide varies greatly, depending on the specific characteristics of the sensor and the satellite, and there is typically a trade-off between a sensor's spatial and temporal resolutions.

Due to physical constraints imposed by the interactions with electromagnetic waves and the soil, MRS satellites are only able to collect soil moisture data from the outermost

soil layers, typically between 0 and 5cm, which is known as Surface Soil Moisture (SSM). This limitation can be circumvented by using models such as the Soil Water Index (SWI) [WLR99], which seek to estimate the soil moisture at deeper layers by integrating surface soil moisture data over time. Thus, remotely sensed SSM data products are valuable even for applications that are mainly tied to root zone SM dynamics.

Remote microwave sensors cannot measure soil moisture directly. Instead, they measure observables related to the backscatter of electromagnetic energy that is reflected from the surface of the Earth. These are then converted into soil moisture estimates by going through data processing pipelines derived from either explicit physical models or black-box models obtained by means of machine learning. Those pipelines often also benefit from using ancillary datasets containing information about variables that can affect soil moisture dynamics or the interaction between microwaves and the surface, such as quantitative data on vegetation density or categorical data on land use.

This process of using models to obtain soil moisture estimates from observations of electromagnetic radiation is known as Data Retrieval (DR), and, accordingly, the soil moisture estimates that it yields are also known as soil moisture retrievals. The data retrieval process carries a considerable level of uncertainty, stemming from multiple systemic and random sources, including, but not limited to:

- Assumptions and simplifications made in the modeling process, in the case of explicit physical models;
- Inaccuracies within models created by machine learning algorithms;
- Interference from electromagnetic radiation used for telecommunication;
- The presence of large water bodies or multiple land cover types within the area illuminated by an active sensor, known as the satellite's footprint;
- Uncertainties carried over from ancillary data used in the modelling or retrieval processes;
- Errors in model calibration;
- Instrument or measurement noise.

Because of those sources of uncertainty, one can expect less precision and accuracy from remotely sensed soil moisture observations than from those obtained by more direct measurement techniques, such as in-situ sensing. The temporal sampling rate of remotely sensed soil moisture observations can also be too coarse for some applications, especially those that need information about the fast-paced surface soil moisture dynamics following quick rainfall events. Similarly, the spatial resolution of remotely sensed data is often insufficient for applications involving the monitoring of meter-scale SM patterns, such as irrigation management.

2.4 Global Navigation Satellite Systems Reflectometry (GNSS-R)

Satellite-based navigation systems have been in use since the mid-20th century, providing geopositioning services with varying levels of precision, timeliness and spatial coverage levels, for both military and civilian purposes. Among those systems, those that offer a global spatial coverage are categorized as Global Navigation Satellite Systems (GNSS). Strictly speaking, there are only four of such systems currently available, all of which are being operated by state agencies [JWD22]. They are :

- **Global Positioning System (GPS)**, developed by the United States of America and currently operated by the U.S. Space Force, a branch of the American military;
- **Global Navigation Satellite System (GLONASS)**, originally developed by the Soviet Union and currently operated by Roscosmos, a state corporation of the Russian Federation;
- **BeiDou Navigation Satellite System (BDS)**, developed by the People's Republic of China and operated by the China National Space Administration (CNSA);
- **Galileo**, developed by the European Union and operated by the European Space Agency (ESA) and the European Union Agency for the Space Programme (EUSPA).

In 1993, [MN93] first proposed the utilization of the backscatter of GNSS signals to extract information about the surfaces from which they were reflected. During the following decades, multiple scientific studies have been conducted in order to leverage those "signals of opportunity", leading to the development of the Global Navigation Satellite Systems Reflectometry (GNSS-R) concept. GNSS-R remote sensing systems make use of microwave radiation receivers to capture electromagnetic signals which have been originally emitted by GNSS satellites and then reflected by the surface of the Earth. Those sensors are usually embarked on aircraft or Earth observation satellites (Figure 2.2).

Since GNSS-R satellites lack their own transmitters, they can be considered an example of a passive remote sensing system. However, if framed as a component of a larger system in conjunction with the GNSS satellites, then GNSS-R satellites can also be considered an active remote sensing technology. Regardless of the interpretation, GNSS-R systems effectively act as a multistatic radar, i.e. a radar system with multiple transmitters and receivers, which are located far apart from each other.

The first GNSS-R applications sought to collect data over the oceans and cryosphere [UPC+21]. In 2000, [MZKE00] conducted an experiment using aircraft-mounted GNSS receivers to first demonstrate the sensitivity of such sensors to soil moisture. In 2016, observations made over land by the GNSS-R satellite TDS-1 were also shown to be sensitive to SM [CSZ+16] [CPP+16]. These results have paved the way for multiple

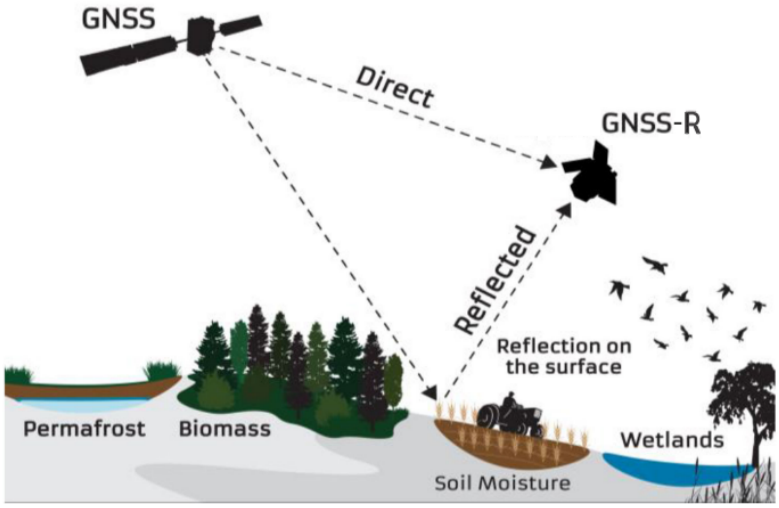


Figure 2.2: Illustration of the GNSS-R concept, adapted from [UPC+21]

studies on the usage of spaceborne GNSS-R sensors for SM retrievals. As of February 2024, this research still primarily relies on satellite missions not originally designed for monitoring SM, such as TDS-1 and CYGNSS. However, there are also new dedicated missions going through their development phase, such as ESA's upcoming HydroGNSS constellation [UPC+21].

Albeit still an experimental technology for the purposes of monitoring soil moisture, GNSS-R satellites are typically much cheaper than the spacecraft used in well-established missions such as SMAP and SMOS. This facilitates the development of larger constellations of satellites, which in turn provides shorter revisit times for each location and facilitates the creation of SM data products with higher temporal resolutions.

2.4.1 The CYGNSS Mission

CYGNSS, which stands for Cyclone Global Navigation Satellite System, is a constellation of eight identical low Earth orbit (LEO) GNSS-R satellites launched by NASA in 2016, with the original purpose of monitoring the activity of cyclones in the intertropical regions. All CYGNSS satellites are equipped with identical receivers, which are able to capture L-band electromagnetic signals that were originally emitted by satellites of the GPS constellation and then reflected by the surface of the Earth [RCL+18].

Despite being originally designed to monitor tropical cyclones by measuring wind speeds over oceans, CYGNSS has also been demonstrated to be sensitive to several observables over land, such as above-ground biomass and soil moisture [CS18]. Notably, CYGNSS is also highly sensitive to the presence of inundation and inland water bodies, even when those are covered by dense vegetation, as is often the case in tropical rainforest biomes [NZS+17]. Since the original scope of the mission was to cover the intertropical regions,

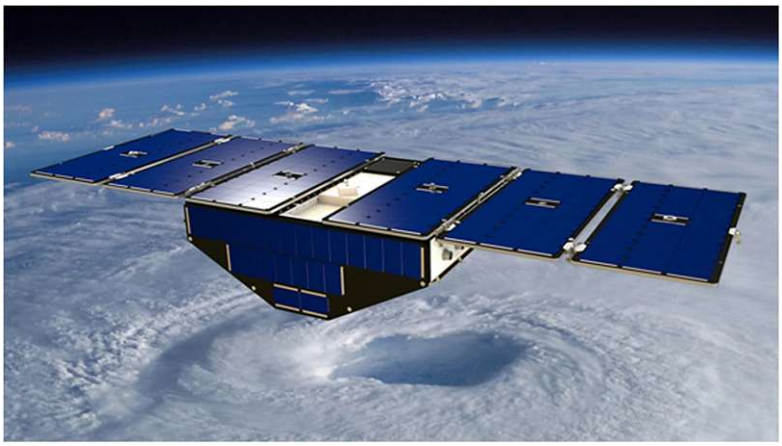


Figure 2.3: Artist's rendition of a single CYGNSS satellite, from [RCL+18]

the CYGNSS satellites follow an orbit that spans latitudes between -35 and 35 degrees [RUD+13], as shown in Figure 2.4.

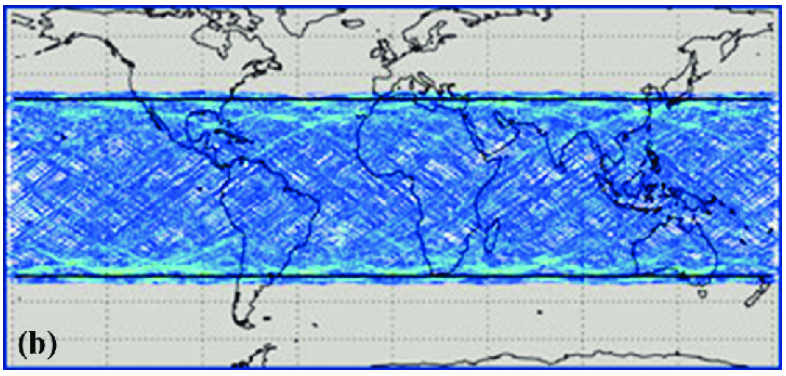


Figure 2.4: Map demonstrating the spatial coverage of the CYGNSS constellation over a period of 24 hours, from [RCLC+22]

While most traditional MRS satellites follow a deterministic data collection pattern, revisiting each location in a repeating and predictable manner, CYGNSS is bound to collecting measurements in a pseudo-random fashion, as for each observation the geometrical arrangement of the transmitter, receiver and the reflection point will be different [CS20]. Thus, in order to produce SM maps from CYGNSS, it is necessary to define a spatiotemporal grid which we can use to aggregate multiple measurements within predefined spatiotemporal cells.

Even though CYGNSS was originally designed to remain operational for only two years, the mission is still active, having remained operational for over seven years as of February 2024. However, an unexpected failure of satellite FM06 in November 2022 has reduced the number of active CYGNSS satellites from eight to seven. Thus, from that date onward, the initial expectations regarding CYGNSS' coverage and revisit times no longer hold.

Nevertheless, the fact that we still have ~87% of the original constellation remaining after a critical failure of one of its members highlights the value of deploying large constellations of cheaper satellites, which facilitates risk mitigation in the long run.

Several SM data products have been developed from CYGNSS reflectivity data. Some of these products employ a more explicit modelling approach, directly applying physical principles [CS20], while others utilize machine learning in conjunction with external SM data products to derive a black-box model that retrieves SM estimates from CYGNSS reflectivity data [SPP+20] [RCS+21] [EKBG19]. For the experiments we have performed in this thesis, we have used the "trackwise" version of the IFAC ANN CYGNSS SM data product developed by Santi *et al.* [SPP+20]. For a detailed description of this product, please refer to Section 6.2.

2.4.2 The HydroGNSS Mission

HydroGNSS is an upcoming GNSS-R mission commissioned by the European Space Agency as part of its Scout Program, which seeks to finance the development of low-budget and scientifically innovative Earth observation missions. As of February 2024, HydroGNSS is still in its development phase, with the launch planned for the end of the same year. The development of the mission is being led by Surrey Satellite Technology (SSTL), in close cooperation with a team of science partners from several academic and research institutions [UPC+21].

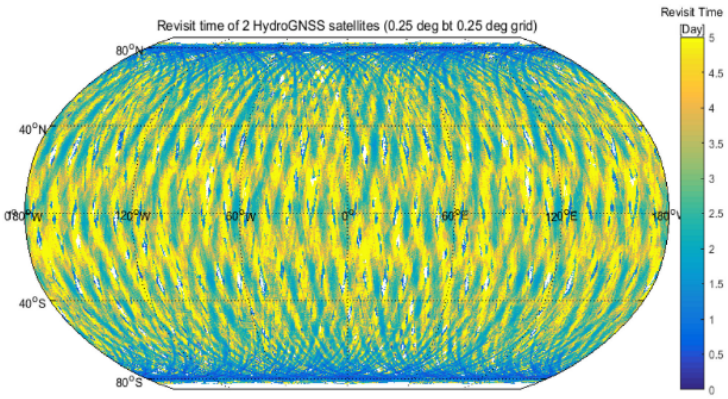


Figure 2.5: Map demonstrating the expected spatial coverage of HydroGNSS, as well as the mean revisit time per location, from [UPC+21]

The HydroGNSS constellation is planned to contain 2 satellites, operating within a near-polar orbit which allows them to cover the extratropical regions not covered by CYGNSS (see Figure 2.5). The global mean revisit time has been calculated to be under 4 days, with a much denser temporal resolution in the boreal and polar regions. It will collect signals emitted by both the Galileo and GPS constellations. Unlike CYGNSS,

HydroGNSS is being designed primarily for applications over land, with the goal of monitoring the following essential climate variables (ECVs):

- Surface Soil Moisture (SSM)
- Above-Ground Biomass (AGB)
- Surface Inundation
- Freeze/Thaw State

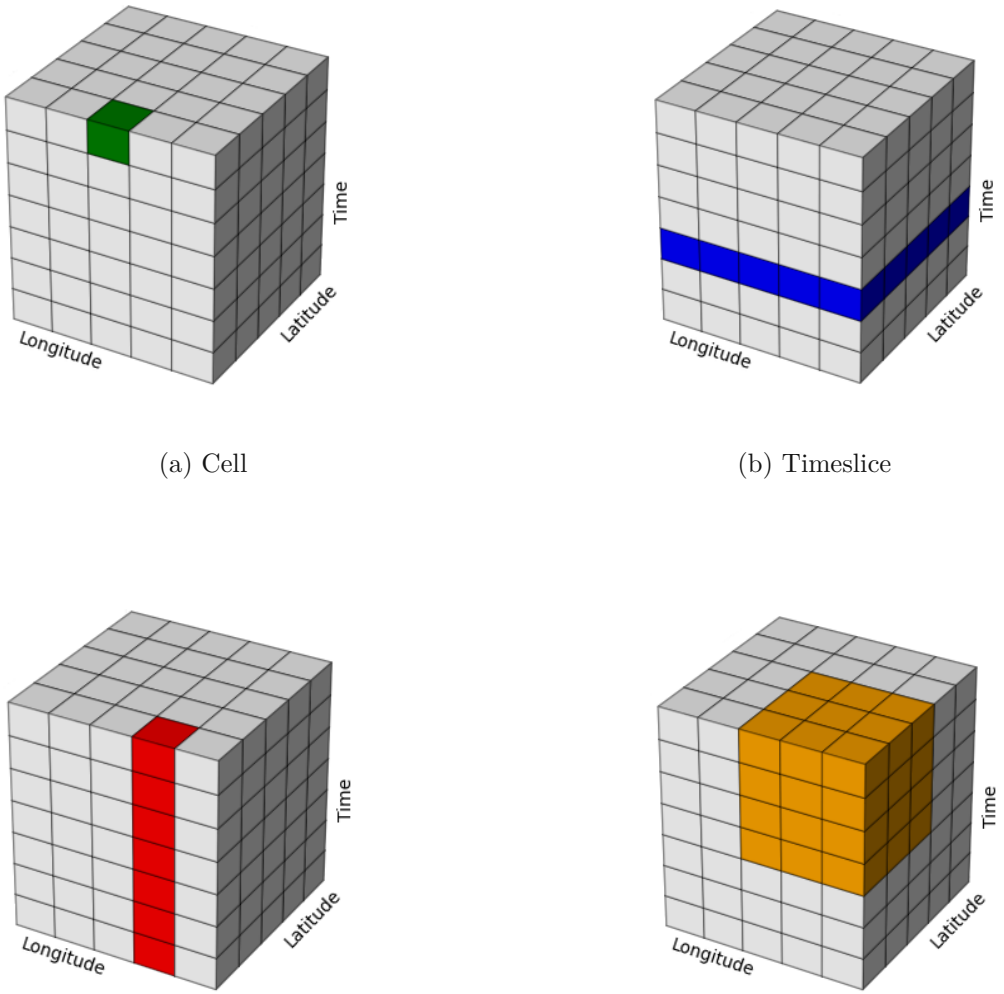
As mentioned in Section 1.3, even though the work described in this thesis utilizes CYGNSS data, it has emerged as part of the HydroGNSS development efforts, in which the Remote Sensing research group of TU Wien's GEO department is involved as a science partner. By understanding how spatiotemporal interpolation can improve the quality of SM data products derived from subsets of the CYGNSS constellation, we hope to introduce a framework that can also be applied to the HydroGNSS constellation and perhaps other future GNSS-R missions.

2.5 Datacubes

Remotely sensed SM data products are often provided in the so-called "datacube" format. In essence, a datacube consists of an N-dimensional data array, with $N \geq 3$, in which some or all dimensions encode some sort of spatiotemporal semantics. The most basic case of a datacube would be a three-dimensional array with two spatial dimensions and a temporal dimension.

The spatial dimensions of a datacube usually have their indices mapped to a well-known system of spatial coordinates such as latitude and longitude, while the temporal dimensions usually have their indices mapped to evenly-sampled timestamps. Thus, every cell of a tridimensional SM datacube contains a measurement, which can also be empty, as well as a spatial location and a timestamp, which are implicitly encoded in its indices. In the context of georeferenced datacubes, specific (latitude, longitude) coordinate pairs along the spatial axes are commonly referred to as "pixels". As shown in Figure 2.6, a subset of the datacube covering all pixels at a fixed timestamp is commonly referred to as a timeslice, whereas a subset covering every timestamp at a single pixel is referred to as that pixel's timeseries. Similarly, a limited data volume that is spread across both the spatial and temporal axes is known as a spatiotemporal neighborhood.

For some tasks, it is useful to think of a datacubes as a video-like data stream, with each timeslice being a frame that displays a specific state of a time-dependent process that is spread across space. For other tasks, it might be more convenient to think of datacubes as multivariate timeseries, with each pixel along the spatial axes defining a different variable. In practice, both of those analogies describe the same data structure, meaning that



(a) Cell

(b) Timeslice

(c) Timeseries

(d) Spatiotemporal Neighborhood

Figure 2.6: Examples of datacube-related terminology, adapted from [LSGD23]

techniques designed for both video processing and for multivariate timeseries processing have the potential to deliver value to problems involving datacube processing.

Datacubes can also have more than three dimensions, which allows for a convenient encoding of information from multiple channels or multiple variables, with a shared spatiotemporal indexing. This shared index facilitates the execution of spatiotemporal queries or operations, and can be leveraged by using hardware optimized for matrix computations such as Graphical Processing Units (GPUs).

CHAPTER 3

Previously-Observed Behavior Interpolation

3.1 Interpolation of CYGNSS Datacubes

As shown in Figure 3.1, the observations made by CYGNSS are pseudo-randomly distributed along tracks that roughly correspond to the projection on Earth of the orbits followed by each CYGNSS satellite. The real footprint of each observation made by CYGNSS is influenced by a myriad of factors, such as topography and incidence angle. However, the minimum theoretical footprint of these observations has been estimated to be of ~ 0.5 km in the across-track direction. In the along-track direction, it is estimated to be of ~ 7 km until July 2019, when there was an update in the CYGNSS integration time (see Section 6.2.2) that changed the theoretical along-track footprint length to ~ 3.5 km. [CS20].

Thus, even though CYGNSS has relatively large number of satellites, providing a wide spatial coverage and short average revisit times, the daily or sub-daily timeslices of CYGNSS-derived datacubes still exhibit large spatial gaps between the tracks, especially in the case of datacubes that bin observations in a high-resolution spatial grid. Moreover, the pseudo-random spatial distribution of the measurements, which depends on the trajectory of both the CYGNSS and the GPS satellites, does not guarantee fixed revisit times for each location.

As a consequence of the existence of those gaps, datacubes with a higher spatial resolution contain many pixels whose timeseries have multiple large and randomly-distributed temporal gaps between consecutive observations. Many pixels of those datacubes will also have a lower than daily average sampling rate. This is especially common in the regions close to the equator, where due to the CYGNSS orbit geometry the tracks are spread further apart from each other. Those issues can be problematic in applications

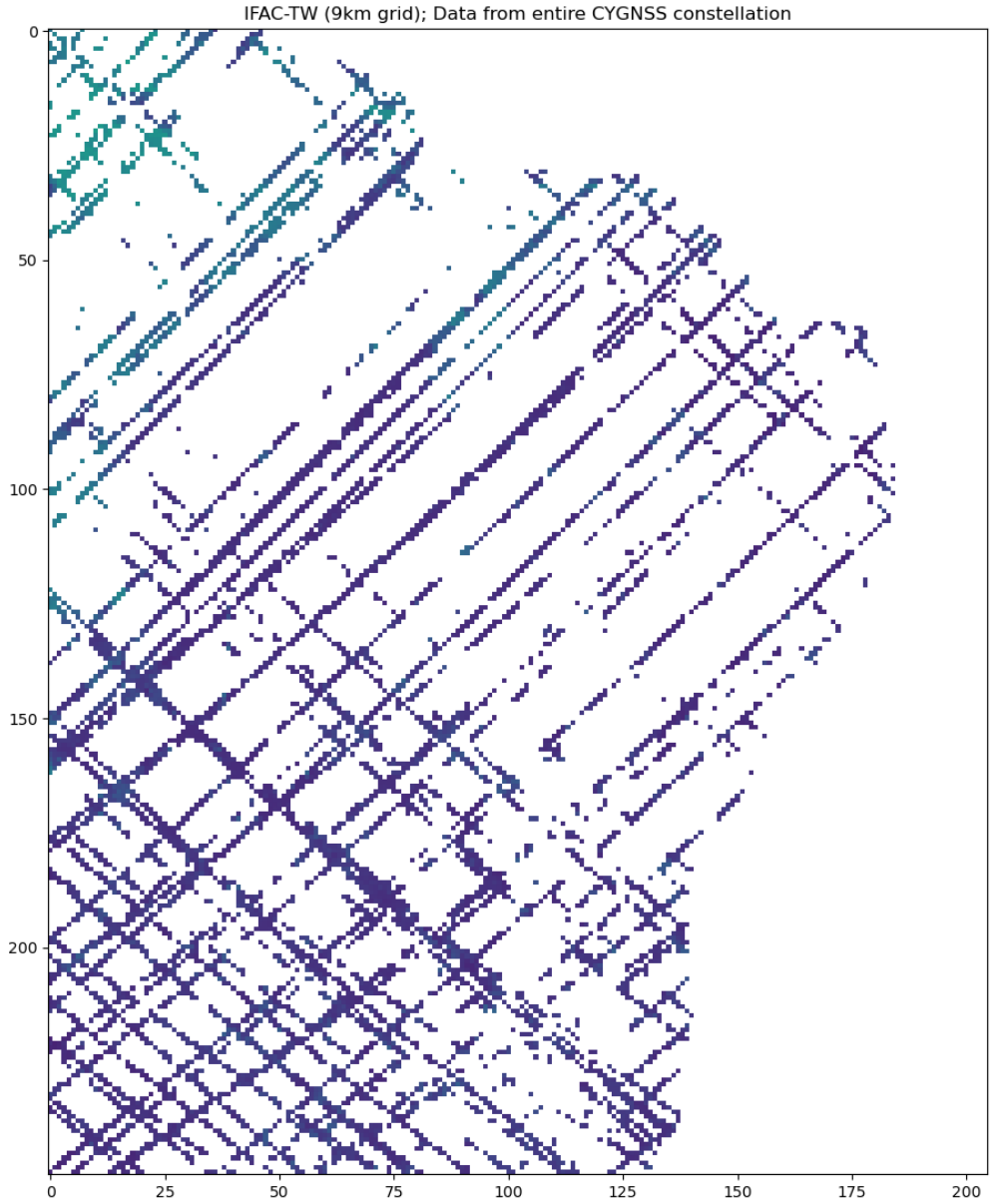


Figure 3.1: Close-up of a timeslice extracted from a CYGNSS-derived SM datacube. In this case, the datacube has been spatially binned along regular grid with a 9km resolution, and temporally binned in 24h windows.

within fields where a high temporal resolution is considered critical, such as meteorology and disaster management.

In this thesis, we are seeking to address this issue by filling those gaps in the datacubes with interpolation techniques. In general terms, interpolation consists in estimating an unknown or missing value of a signal based on the known values in its vicinity. While it can theoretically be performed in domains with an arbitrarily high dimensionality, the most common applications involve domains ranging between one and three dimensions. In our particular problem, we are working with signals that are a function of two spatial and one temporal dimension, and thus we refer to the problem as spatiotemporal interpolation. Interpolation can be framed as a regression problem, where the unknown value is the response variable and the set of neighboring known values are the predictors.

As CYGNSS backscatter observations are very sensitive to surface soil moisture and the presence of inland water bodies, CYGNSS-derived datacubes containing data about either electromagnetic radiation or SM retrievals exhibit a high degree of spatial heterogeneity and discontinuity, especially in regions with an abundance of rivers or lakes. Thus, a large portion of the classic interpolation methods, such as linear or cubic interpolation, are unable to perform well with with CYGNSS-derived datacubes, as they rely on rather strong assumptions about the smoothness of the signal being interpolated [Che23]

3.2 Theoretical Basis - Temporal Stability of Soil Moisture

In general terms, Temporal Stability of Soil Moisture (TSSM) refers to a phenomenon in which the temporal dynamics of soil moisture at the local level are strongly correlated to the temporal dynamics of soil moisture at the regional scale in the surrounding area. In this context, "local" usually refers timeseries extracted from point-scale measurements or from pixels in high-resolution grids, while "regional" usually refers to the corresponding pixels in a coarser grid, or even an aggregate timeseries averaged over the surrounding spatial neighborhood.

TSSM was first introduced in 1985 through an analysis of in-situ data [VPDSBV85]. Since then, a multitude of regional studies have been conducted in order to analyze the effects of TSSM under different environmental conditions and spatial scales [VVH+12].

In 2008, it was first demonstrated by Wagner *et al.* [WPD+08] that temporal stability can also be observed in radar backscatter data collected by C-band microwave remote sensing satellites. This result is a consequence of the well-established fact that the backscatter of microwave signals over land is highly correlated with the dielectric constant of the soil being illuminated. This constant, in turn, is a function of the water content of the topmost 5 cm of soil, i.e. the surface soil moisture (SSM) [HUD+85]. This relationship between microwave backscatter and SSM is usually linear, and also varies across space, as it is influenced by location-dependent factors such as topography and vegetation cover. It has been extensively exploited in the development of remotely sensed SM data

products derived from microwave sensors such as those aboard SMAP [ENO+10] and SMOS [KWW+10], and has also been observed in CYGNSS data [CS18][CS20]

The temporal stability of soil moisture acts as the underlying mechanism driving several interpolation and downscaling algorithms designed for improving remotely sensed SM data products. A notable example is the SCATSAR-SWI algorithm [BMPH+18], which exploits the historic dynamics of localized spatiotemporal SM patterns of a high-resolution product derived from the Sentinel-1 constellation to create a large collection of pixel-specific downscaling parameters. Those parameters are then used to improve the spatial resolution of coarser SM data products derived from the ASCAT constellation. TSSM is also the driving factor behind POBI, a spatiotemporal interpolation algorithm designed specifically for GNSS-R derived datacubes, which we will describe in detail in the following section.

3.3 Previously-Observed Behavior Interpolation (POBI)

The Previously-Observed Behavior Interpolation algorithm (POBI) [Che23] is a spatial interpolation solution that was specifically designed to address the challenges brought by the spatiotemporal distribution of GNSS-R measurements. Conceptually, it takes inspiration in autoregression, which is a timeseries forecasting technique that exploits a variable's historical relationship with its past values to estimate the current of future observations. However, instead of simply exploiting the past temporal dynamics of a pixel's timeseries, POBI focuses on exploiting the correlations between that pixel's timeseries and those of the pixels its spatial neighborhood.

In essence, POBI trains a collection of highly localized, pixel-specific regression models which rely on the historic relationship between the target pixel and a fixed-size spatial neighborhood around it. After training, those models, which we will refer to as "POBI Interpolators", can be used to fill the empty cells of the datacube, provided that there are enough known values in their respective neighborhoods to serve as input to the models. While at prediction time POBI only performs spatial interpolation, it also relies on temporal information at training time, and thus can be classified as a spatiotemporal interpolation algorithm.

In the paper where it was introduced [Che23], POBI was applied to CYGNSS-derived reflectivity data. More specifically, it was used to interpolate gaps in datacubes containing information on the effective surface reflectivity (Γ). Γ is measured in dB and defined for the CYGNSS instrument according to Equation 3.1. In this equation, P_r is the peak value of the Delay-Doppler Map computed by CYGNSS at that location, λ is the wavelength of the GPS signals, P^t is the transmitted power, G^t and G^r are the gains of the transmitting and receiving antennas, respectively. R_{ts} is the distance between the observation's specular point and the GPS transmitter, and R_{rs} is the distance between that point and the CYGNSS receiver. [CS20]

$$\Gamma[\mathrm{dB}] = 10\,(\log \mathrm{P_r} - \log \mathrm{P^t} - \log \mathrm{G^t} - \log \mathrm{G^r}) + 20\,(\log(\mathrm{R_{ts}} + \mathrm{R_{sr}}) - \log \lambda + \log 4\pi) \quad (3.1)$$

Given the fact that Γ is strongly influenced by SMM, which is encapsulated in Equation 3.1 by the term P_r, and given the fact that POBI relies on the assumption that the temporal relationship between each pair of neighboring pixels is stationary, it follows that POBI also implicitly relies on the temporal stability of soil moisture to interpolate reflectivity data from CYGNSS. Thus, we expect that POBI's performance in interpolating CYGNSS reflectivity datacubes to translate relatively well to SSM datacubes whose values have been retrieved from CYGNSS reflectivity data.

3.3.1 POBI algorithm - Training

In order to model the spatiotemporal behavior of CYGNSS reflectivity or SSM retrievals across space and time, POBI adopts a location-centric strategy, where a separate prediction model or interpolator is trained for each pixel along the spatial grid used in the datacube, modeling each pixel's specific behavior with respect to its local neighborhood. Those pixel-wise interpolators are, in essence, a weighted ensemble of linear regressors, with one regressor per neighbor. It should be noted that, in the context of POBI, the term "neighbor" refers not only to the pixels directly adjacent to the target pixel, but rather to any pixels contained within the fixed-size spatial neighborhood. At prediction time, a cell with missing data can be interpolated by feeding the known values in its neighborhood to the corresponding regressors, and then performing a weighted mean over the outputs of those.

Before training the POBI interpolators, the user must choose values for three hyperparameters:

- Neighborhood Length (l_n): A POBI interpolator estimates a cell's missing value from the valid measurements that are available within that cell's spatial neighborhood in the same timeslice. This spatial neighborhood is defined as a square across the spatial axes, with a discrete length l_n, having the target cell in the center. Thus, l_n must be an odd integer greater than or equal to 3.
- Concurrency Window (w_c): In order to establish the relationship between two neighboring pixels, POBI has to find all instances of concurrent observations across both of their timeseries. The co-occurrence window specifies how distant in time two observations must be in order to be still considered concurrent, and will typically take on values ranging between a few hours and a week.
- Minimum Concurrency Threshold (c_{min}): When training a POBI interpolator, we want to avoid establishing spurious correlations due to a lack of enough concurrent measurement between two neighbors. Thus, unless there are at least c_{min} concurrent measurements between the target pixel and a neighbor's timeseries, this neighbor will be considered invalid and will be ignored by the interpolator at prediction time.

The user must also select a calibration period which will be used to train the interpolators. This period must be contained within the time span covered by the datacube, and ideally cover multiple years, in a way that for each pixel there are a few examples of each season. The calibration period, however, should not cover too much of the datacube's available time span, as a fair evaluation of POBI's performance must be carried out using the timeslices that were not used for training. This setup is equivalent to the standard train/test split in a classic machine learning experiment.

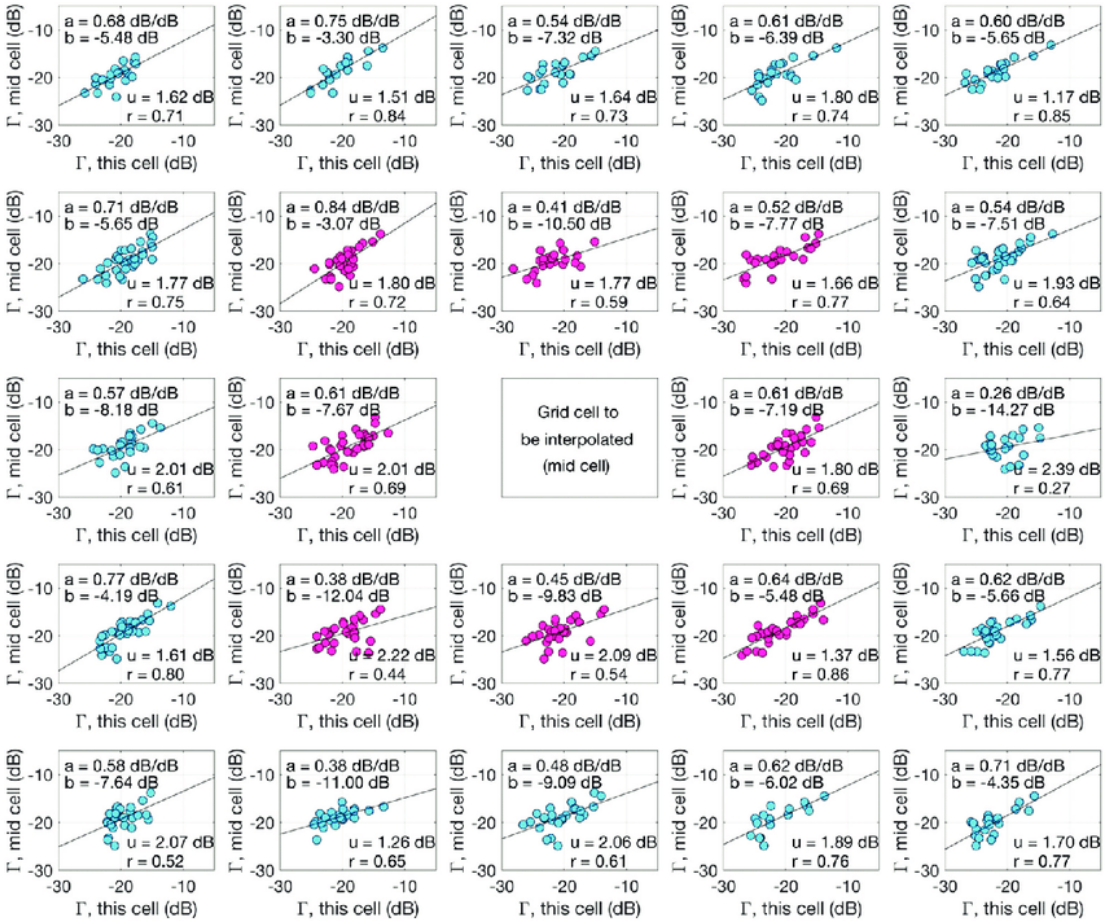


Figure 3.2: Linear regressions computed in the process of training a POBI Interpolator for a single pixel, from [Che23]

The training of a POBI interpolator for a given pixel p is conducted as follows: For each pixel q within the neighborhood of p' as defined by l_n, we search the timeseries of both pixels for all instances of concurrent valid measurements m_p and m_q in both p and q. The concurrency threshold is defined by the hyperparameter w_c. If the number of co-occurrences between p and q is equal to or greater than c_{min}, we treat all co-occurrences as data points in a two-dimensional plane, and fit a line between the measurements in p and q as illustrated in Figure 3.2. From this linear regression, we compute the slope a,

the intercept b, the correlation coefficient r and the residual standard deviation u. Thus, POBI stores four parameters per neighbor per pixel.

The total number of parameters required by POBI can thus be computed by Equation 3.2, where n_{pixels} is the total amount of pixels in the grid.

$$n_{\text{params}} = 4 \cdot n_{\text{pixels}} \cdot l_n^2 \tag{3.2}$$

In order to optimize disk usage and memory I/O, the pixels where SSM is not well defined, such as oceans or inland water bodies, can be safely ignored. This reduces the number of pixels for which we need to store parameters, albeit by no more than one order of magnitude. Strictly speaking, it is also not mandatory to store the residual standard deviation u of each pixel pair, as this parameter is not used for performing interpolation at prediction time, but rather just as an indication of the uncertainty of the relationship that has been established between the pixels in the pair. Thus, the lower bound for the number of parameters required by a POBI ensemble is given by Equation 3.3, where r_v is the ratio of valid pixels within the spatial grid used in the datacube.

$$n_{\text{params}} = 3 \cdot n_{\text{pixels}} \cdot r_v \cdot l_n^2 \tag{3.3}$$

3.3.2 POBI algorithm - Prediction and Evaluation

At prediction time, POBI estimates the value of an empty cell by applying Equation 3.4. Just like in the original POBI paper, the equation is defined in terms of $\hat{\Gamma}$, but can also be applied to SSM measurements without any changes in its structure.

$$\hat{\Gamma} = \frac{\sum_{i=1}^{n_v} w_i(a_i\Gamma_i + b_i)}{\sum_{i=1}^{n} w_i} \tag{3.4}$$

In this equation, the parameters a_i, b_i and w_i of neighbor i are derived directly from the offline parameter database computed at training time, with $w_i = r_i^2$. The index i only cycles through cells with valid data in them, and thus the number of valid neighbors n_v changes for every cell, and is bounded by l_n^2. In the case of cells with very few valid measurements available in their neighborhood, the prediction quality can potentially be compromised, especially if all of the available measurements are located in pixels whose correlation coefficients with respect to the target cell's pixel are low. Thus, it can be beneficial to define a minimum threshold for the number of valid measurements that are required to run a prediction.

If there are significant data gaps remaining in the datacube after a first pass of POBI, the algorithm can also be applied recursively. However, this may degrade the overall data quality of the interpolated datacube, as each recursion step will compound the prediction uncertainties.

The performance of the POBI interpolators can be measured with standard regression error metrics such as MSE or RMSE, with the ground truth values coming from in-situ data, remotely sensed data from other satellites, or even data from CYGNSS itself, if POBI is being used to interpolate datacubes from a subset of the CYGNSS constellation (see Chapter 5).

When using ground truths from sources other than CYGNSS, it is important to remove the bias between the two datasets, which can be done using techniques such as mean subtraction or CDF-matching [GDA+20]. A popular metric that simultaneously performs debiasing and error measurement is the unbiased RMSE (ubRMSE), shown in Equation 3.5. It essentially computes the RMSE between two signals, after mitigating sensor bias by removing the mean value of each signal.

$$\text{ubRMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[(X_i - \bar{X})(Y_i - \bar{Y})]^2} \tag{3.5}$$

3.4 Challenges and Mitigation Strategies

The main drawback of any spatiotemporal interpolation algorithm, including POBI, is that it will fail to capture atypical events whose temporal or spatial extent is short enough to be entirely contained within spatiotemporal gaps in the datacube. In the context of soil moisture, this phenomenon could materialize, for instance, in the form of a brief rainfall event that is not expected at that area or season, causing a temporary uptick in the SM timeseries of pixels in that area. If that rainfall event's spatiotemporal extent is small enough that it is completely contained within a gap in the datacube, there is no way that an interpolation algorithm can estimate it properly.

In essence, this issue is a manifestation of the Nyquist-Shannon sampling theorem, which is one of the tenets of digital signal processing theory. In general terms, this theorem states that, in order to completely capture the information of an analog signal in the form of a digital signal, the analog signal must be discretized at a sampling frequency at least twice as high as the highest frequency present among its spectral components.

While this limitation does not invalidate the usefulness of interpolation in geospatial data products, care must be taken by users whose applications require a faithful representation of those atypical events.

One particular drawback of POBI is the amount of parameters that the complete ensemble of POBI interpolators requires, as specified in Equation 3.2. While the effectiveness of POBI's location-centric approach has already been demonstrated through validation against external datasets [Che23], the algorithm's method of encoding information is not very memory-efficient. As each POBI interpolator must individually encode both the general physical dynamics of the observable and the local relationships between the pixels within the corresponding neighborhood. Thus, even though each POBI interpolator

can only be applied to its corresponding pixel, all of them are carrying some level of redundant information about the physics of the same physical quantity.

A potential strategy to mitigate those redundancies, performing spatiotemporal interpolation in a more memory efficiency manner, would be training a "global" model. This model would encode both the basic physical dynamics of the observable and the spatially varying phenomena that arise from different environmental conditions and local geographic features. Such an interpolator would be applicable to every pixel, and thus would be able to better compress information by eliminating the need to re-encode the base dynamics of the observable multiple times.

Due to their inherent capacity for modelling intricate and nonlinear functions while compressing a high volume of information derived from complex datasets, neural networks emerge as a natural candidate for achieving these goals. The convolutional family of neural network architectures is particularly well-equipped to handle spatiotemporal neighborhoods, as they are optimized for dealing with data volumes in which the volumetric pixels contain complex semantic relationships with their neighbors.

In Chapter 4, we will provide further background on neural networks. Then, in Chapter 5, we will , and propose a neural network architecture aimed at solving the problem of STI for GNSS-R data, while addressing some of the drawbacks of POBI.

CHAPTER 4

Neural Networks and Deep Spatiotemporal Interpolation

4.0.1 Artificial Neural Networks

The first studies on the bioinspired computational models that originated the modern Artificial Neural Networks (ANNs or simply NNs) date back to 1943, when McCulloch and Pitts published a paper [MP43] using propositional logic to provide a mathematical description of the behavior of neural networks. During the rest of the 20th century, a considerable amount of research was conducted in ANN theory and practice, with several novel artificial neural network architectures being proposed and implemented on the computer systems that were available.

The most prevalent of those architectures was the Multilayer Perceptron (MLP), which is composed of an ensemble of artificial neuron units individually known as Perceptrons (see Figure 4.1). In a classic MLP, the artificial neurons are grouped into layers, with each neuron in layer n receiving as inputs the output information from every neuron in layer $n-1$, but not receiving inputs from other neurons in the same layer or in the subsequent layers. Thus, as the information only flows in one direction, they are classified as Feedforward Neural Networks (FNNs).

In most cases, neural networks are trained by using variations of Stochastic Gradient Descent (SGD), an algorithm which uses local gradients to iteratively search for local minima of a loss function applied to the model's parameter vector. As it would be very costly to compute the steps of SGD analytically, they are typically approximated through an automatic differentiation algorithm called Backpropagation [Lin76] [RHW86].

In order to compute a SGD step, the Backpropagation algorithm starts from the value of the loss function with respect to the model's prediction over a subset of the training data. It then estimates the gradient of the loss function with respect to each element

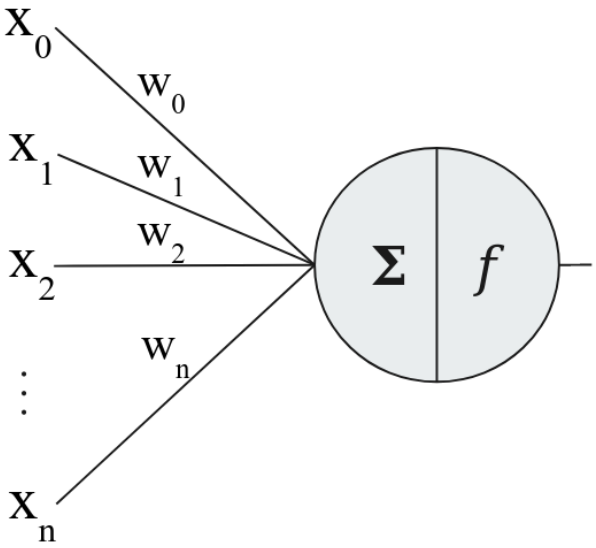


Figure 4.1: The Perceptron artificial neuron, which uses learnable weights to perform a linear combination of the components of the input vector. The result of this linear combination is used as input to an activation function f, whose output is also the final output of the neuron

of the model's current parameter vector, starting from the output nodes and iteratively progressing towards the input nodes. In this process, it efficiently re-utilizes the precomputed value of gradient with respect to the "later" neurons to speed up the application of the chain rule when computing the gradients of the "earlier" neurons.

Initially, NNs found some employment in a wide range of domains, especially in problems related to prediction and pattern recognition. Those applications of NNs, however, were often bounded in their scope and scalability by hardware limitations, as the training of NNs typically involves high computational costs, even when using backpropagation. Another obstacle for their wide adoption was high cost of collecting enough labeled training data, which remains a meaningful challenge in many problem domains.

From the 1990s onwards, steady advances of modern hardware capabilities have made the training of larger and more complex neural networks more feasible. An important driver of this process was the adoption of Graphical Processing Units (GPUs) as the hardware substrate for model training. Even though GPUs were originally developed for graphical processing, finding ample usage, for instance, in the videogame industry, the fact that they are optimized for matrix-based computations made them an ideal candidate for accelerating the linear algebra operations involved in the backpropagation algorithm.

In 2012, a Convolutional Neural Network (CNN) called Alexnet [KSH12] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), a yearly competition of

object recognition systems [RDS+15]. CNNs take as inputs data volumes with spatially distributed information, such as RGB images, and make use of the so-called convolutional layers to perform trainable feature extraction on the raw inputs. The extracted features are then fed to an inference block, which can be trained for a variety of tasks, such as classification or regression. In Section 4.1, CNNs will be described in greater detail.

While CNNs had already been successfully employed to solve small-scale image processing tasks [LBD+89], Alexnet's performance in the ILSVRC was particularly remarkable as it was the first time that a neural network had achieved such a high performance in this rather difficult challenge, and with a considerable advantage over the other competitors. That result ushered a significant surge in the usage of NNs in the fields of image processing and computer vision, with powerful convolutional models being trained on GPU hardware and redefining the state-of-the-art for tasks like image classification and segmentation.

The breakthroughs that CNNs, Transformers [VSP+17] and other contemporary neural network architectures brought to multiple fields in the 2010s and 2020s roughly coincide in time with the increasing adoption of the term "Deep Learning" (DL) in both technical and nontechnical media. While there is no formal definition for the term, Deep Learning generally refers to the usage of neural networks that contain a large number of stacked layers, performing multiple composed transformations when processing their inputs. Although there is no consensus on the number of layers that a NN needs to have in order to be considered "deep", this designation tends to be used to contrast modern architectures with the more "shallow" NNs that were commonplace in the early years of the field. Machine learning models that are not based on NNs, such as KNN or SVM classifiers, are also often referred to as shallow models. In practice, the term "Deep Learning" has become synonymous with the usage of neural networks to solve machine learning tasks.

Even though the term "neural network" remains, modern NNs have deviated a lot from their original inspirations in the behavior of biological neurons. In a stricter sense, they can be better described as a family of computational models constructed by composed and differentiable computational units, whose parameters can be trained by optimization algorithms to approximate the behavior of a wide range of non-linear functions.

4.1 Convolutional Neural Networks

Convolutional Neural Networks were first introduced in the start of the 1980s [Fuk80], finding some limited employment in simple image processing tasks which did not require excessive computational power, such as identifying handwritten postal codes [LBD+89]. After Alexnet [KSH12] won the ILSVRC, convolutional models started to gain traction in the image processing and computer vision domains, changing the state of the art for tasks like image classification, image segmentation and object detection [KSZQ20].

CNNs incorporate the concept of representation learning, which consists of using trainable feature extractors that can learn how to best convert the features of the input data into

a new representation that is more convenient for a machine learning model to process [BCV13]. This is achieved by using what are called convolutional layers (CLs). Each CL has n_c channels, a width w and a height h. Each channel has $w * h$ neurons, and they all share the same trainable weights. The weights of each channel are trained to encode spatial filters, which at prediction time are convolved with the CL's input to produce activation maps that indicate areas in the input which share a high similarity with the filter's weights. In practice, when convolutional layers are stacked sequentially, this behavior allows for the training of hierarchical filter banks, where the higher-level filters build upon the abstractions provided by the lower ones in order to detect more complex patterns.

In essence, CNNs are optimized to perform inference over inputs that manifest themselves as discretized data volumes with spatial structure, where the positional relationship between each volumetric pixel and its neighborhood carries meaningful information. The most prevalent applications of CNNs take as inputs RGB images, which consist of data volumes with a shape $(3,h,w)$, where h is the image's height and w is the image's width. This, however, is only a special case of the class of inputs a CNN can process, as convolutional layers can take as input volumes with an arbitrary number of channels.

The most simple CNN architecture for classification and regression contains one convolutional block, focused on feature extraction, followed by a fully-connected block, focused on inference. The convolutional block consists of a sequence of convolutional layers, taking as input the raw data and providing as output an alternative representation of this data in a new feature space. The fully-connected block is, in essence, a feedforward neural network, taking the transformed data as input and providing a prediction as output. Both blocks are trained simultaneously with SGD and backpropagation, in a way that the model learns to both extract convenient features from the raw data and perform satisfactory inference on those.

The feature extractors of some modern CNNs like Highway Networks [SGS15], ResNets [HZRS16] and DenseNets [HLVDMW17] include shortcut connections that can bypass convolutional layers, creating multiple possible paths through which information can flow through (see Figure 4.2). Intuitively, the shortcut connections allow the training process to focus on finding convolutional kernels that are capable of enriching the information provided by the preceding layer, instead of simply converting it into a new representation with higher levels of abstraction. In practice, this allows for the training of deeper and more powerful NNs.

4.2 DenseNets

The DenseNet [HLVDMW17] is a CNN architecture that takes inspiration in the shortcut connections used by ResNet (Figure 4.2), but replacing the identity summation approach with a concatenation of feature maps along the channel dimension. The overarching architecture of a DenseNet is similar to that of a traditional CNN, containing a feature extraction component followed by an inference component.

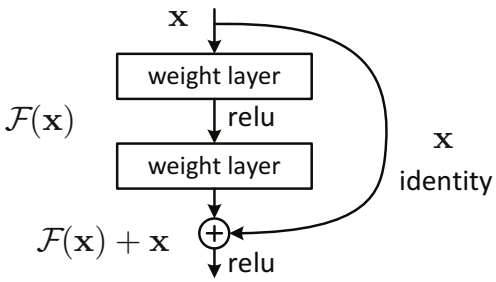


Figure 4.2: Diagram of the residual block used in ResNets, where the input tensor is combined with the output of the convolutional layers by addition. From [HZRS16]

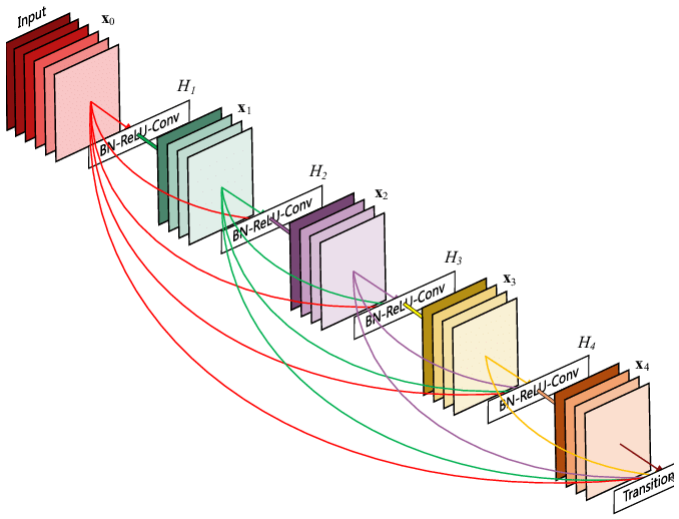


Figure 4.3: Overview of a dense block. Every dense layer takes as input a concatenation of the outputs of all previous dense layers within the block. From [HLVDMW17]

The core element of a DenseNet's feature extractor is the dense block, which contains a set of sequentially arranged convolutional layers. Within a dense block, every convolutional layer l_n receives as input the concatenation of the original input of the dense block with the outputs of all the layers between l_1 and l_{n-1}. This behavior can be modeled as shown in Equation 4.1, where $H(\cdot)$ denotes the nonlinear transformation performed by a convolutional layer, $\mathbf{x}_0$ denotes the input of the dense block, $\mathbf{x}_i$ denotes the output of the convolutional layer l_i and , denotes concatenation along the channel dimension.

$$\mathbf{x}_n = H([\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}]) \tag{4.1}$$

Dense blocks often combine the convolutions with auxiliary operations for responsible for regularization, such as batch normalization or dropout. In most cases, the convolutional layer is also succeeded by an activation function such as Sigmoid or ReLU. As suggested

in the original DenseNet paper [HLVDMW17], all of those operations can be included in the definition of $H(\cdot)$, thus facilitating notation.

An important hyperparameter to consider when designing a DenseNet is the growth rate k, which controls the number of channels in the output of all convolutional layers across the network. As those outputs get concatenated to each other within the dense blocks, it is important to choose a value of k that is large enough to provide satisfactory performance, but small enough to keep the model within reasonable bounds.

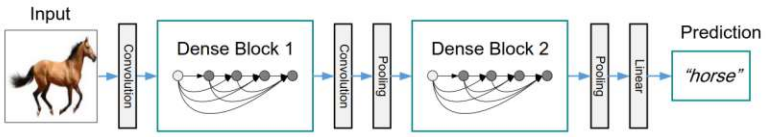


Figure 4.4: High level diagram of the DenseNet architecture, demonstrating an use case where it is employed in image classification. Adapted from [HLVDMW17]

In order to improve the scalability of the model's size, regardless of the value chosen for k, DenseNets also employ a transition block between each dense block and the subsequent one. A transition block contains two essential components: a 1x1 convolution layer, which reduces the dimensionality of the feature maps along the channel dimension, and a pooling layer, which reduces the dimensionality along the width and height dimensions. Those blocks can also include regularization elements such as batch normalization. This design allows the dimensionality of the feature maps to get progressively reduced as information flows through the DenseNet, avoiding an excessive number of parameters that could lead to unnecessary complexity.

CHAPTER 5

Deep Spatiotemporal Interpolation

5.1 General Considerations

Neural networks can also be trained to solve spatiotemporal interpolation problems, provided that there are enough ground-truth data available for the training process, either acquired from real-world sources or by employing data synthesis methods. Architectures that are optimized for performing inference on data volumes, such as those from the CNN family, are especially proficient in dealing with spatiotemporal information from the geospatial domain. This proficiency stems from the underlying structure of this type of data, which usually involves high levels of interdependence between pixels that are in close spatiotemporal proximity. Architectures designed for dealing with sequential phenomena, such as recurring neural networks or attention-based models, can also be valuable for performing STI on geospatial data, due to their proficiency in capturing dependencies along the temporal axis.

In this thesis, we present a NN-based spatiotemporal interpolation solution specifically tailored to the problem of filling gaps in GNSS-R-derived datacubes, particularly those containing soil moisture data derived from CYGNSS. With this solution, we seek to address some of the drawbacks of POBI, which is the state-of-the-art technique for performing STI on GNSS-R-derived datacubes (see Section 3.4). Mainly, we seek to have a unified regression model that is able to perform STI "globally" across the datacube, regardless of location-dependent environmental conditions and time-dependent trends such as rainfall seasonality. While this model does not have to disregard such conditions, location-specific and time-specific ancillary data regarding the target pixel must be encoded within the input vectors provided to the unified model, instead of being used as a mechanism for choosing between one of multiple location-specific models in a large

ensemble. This allows the model to somewhat emulate POBI's ability to encode location-specific knowledge, while also mitigating redundancies by exploiting a neural network's ability to compress information applicable to multiple regions of the datacube within a smaller set of parameters.

5.1.1 Related Work

Multiple studies have explored the application of neural networks to perform spatial, temporal or spatiotemporal interpolation on soil moisture data. Some works, such as [FVS+20], only conduct interpolation along the temporal axis, treating each pixel along the spatial axes as an isolated problem. This particular study, which also deals with SM data, makes use of Long Short-Term Memory (LSTM) recurrent networks to perform temporal SM interpolation in a pixel-wise fashion over a 200-hectare field, with a high spatial resolution. [ISM+23] also addresses the interpolation of soil moisture through deep learning, but employs a different approach. In this study, which focuses on the region around a railroad in eastern Canada, the interpolation of SM is only performed along the spatial axes, making use of Radial Basis Function Neural Networks (RBFN).

A large body of work on deep learning-based interpolation of geospatial data has also been developed outside the scope of Soil Moisture data. [VJF+21], for instance, uses Convolutional Autoencoders to interpolate gaps in remotely sensed sea surface sediment concentration data over a fixed region off the coast of France. [DYS+19] trains a Deep Convolutional Generative Adversarial Network (DCGAN) to perform purely spatial interpolation of remotely sensed sea surface temperature data, treating the interpolation task as an image inpainting problem. [VKT+21] developed an attention-based neural network interpolator that is capable of modelling both spatial and temporal dynamics, effectively performing spatiotemporal interpolation, and applied it to surface spectral reflectance data over Greenland and Australia.

Kirkwood *et al.* [KEPO22] employ a convolutional architecture to perform spatial interpolation of point-scale measurements of calcium concentration, producing spatially dense maps of this same variable. Their model takes two parallel inputs. The first one is a two-dimensional matrix containing spatial information about the relative elevation around the pixel of interest, which is directly supplied to a convolutional feature extractor. The second input is a vector of ancillary location information which is concatenated to the flattened output of the convolutional block before being supplied to the inference block. Thus, the model effectively learns to interpolate a point-scale dataset by learning how to estimate calcium concentration based on inputs from a different set of variables.

5.1.2 Proposed Solution - Deep Convolutional Spatiotemporal Interpolation

Much of the research in NN-based interpolation of geospatial data relies on the existence of gapless patches of data which can be used as ground-truth information for model training. In order to have corresponding input data, it is common to introduce artificial

gaps on the ground-truth patches. This approach is especially prevalent in work involving generative models such as GANs or Variational Autoencoders. In the context of CYGNSS datacube interpolation, it is trivial to acquire patches with gaps to use as input data, but we are challenged with the lack of a gap-free version of the CYGNSS datacube from which complete ground-truths could be extracted for every input.

A rather straightforward approach to overcome this issue would be obtaining ground-truth information from external data products that contain fewer or no spatiotemporal gaps. Those could be data products derived from other remote sensing systems, such as SMAP or SMOS, or even model data such as ERA-5 Land [MSDAP+21]. However, when working with data that originated from different sensors or different data retrieval pipelines, it is crucial to address the effect of the biases that exist between the products (see Section 2.3). In our specific use case, failing to correct those biases would result in an interpolation model which fills the gaps in CYGNSS datacubes with SSM predictions influenced by the biases from the external dataset. Consequently, the data in the interpolated cells would follow a slightly different distribution from that observed in the known values. Therefore, enforcing the constraint of using CYGNSS-derived SM in both the input and output vectors would significantly simplify the complexity of data preprocessing, while mitigating a potential source of additional uncertainty in our modelling.

With these considerations in mind, we propose novel strategy for performing STI on GNSS-R data, for which we have coined the name Deep Convolutional Spatiotemporal Interpolation (DCSTI). It takes inspiration from two sources: POBI's autoregression-based approach of training models using only CYGNSS data, and the CNN-based cell-wise interpolation process proposed by Kirkwood *et al.* [KEPO22].

This strategy consists of training a deep learning-based spatiotemporal interpolation model that is not focused on filling entire spatiotemporal gaps in one pass. Instead, it uses a CNN to perform interpolation across the datacube by treating each empty cell as its own individual regression problem. For each empty cell, the regression uses as predictors the valid observations that can be found within cell's spatiotemporal neighborhood (see Figure 5.1). This can be done by treating the spatiotemporal neighborhood as a data volume and providing it as input to the model's convolutional block. The specific shape of the neighborhood used as input to the model is treated as a hyperparameter.

Our proposed model can also take as a secondary input an array containing location-dependent or time-dependent ancillary data, such as the target cell's spatial coordinates, timestamp, or even land cover information. This array is concatenated with the output of the convolutional block, with the result of this concatenation serving as input to the inference block. Here, we adopt a slightly different approach from that of Kirkwood *et al.*, making the usage of such ancillary data optional. In practice, the choice of whether to use any ancillary inputs and the choice of which ancillary data to include are also treated as hyperparameters.

Thus, in the simplest case, the input vector of a DCSTI model is the spatiotemporal neighborhood around the empty cell of interest. Depending on hyperparameter choices,

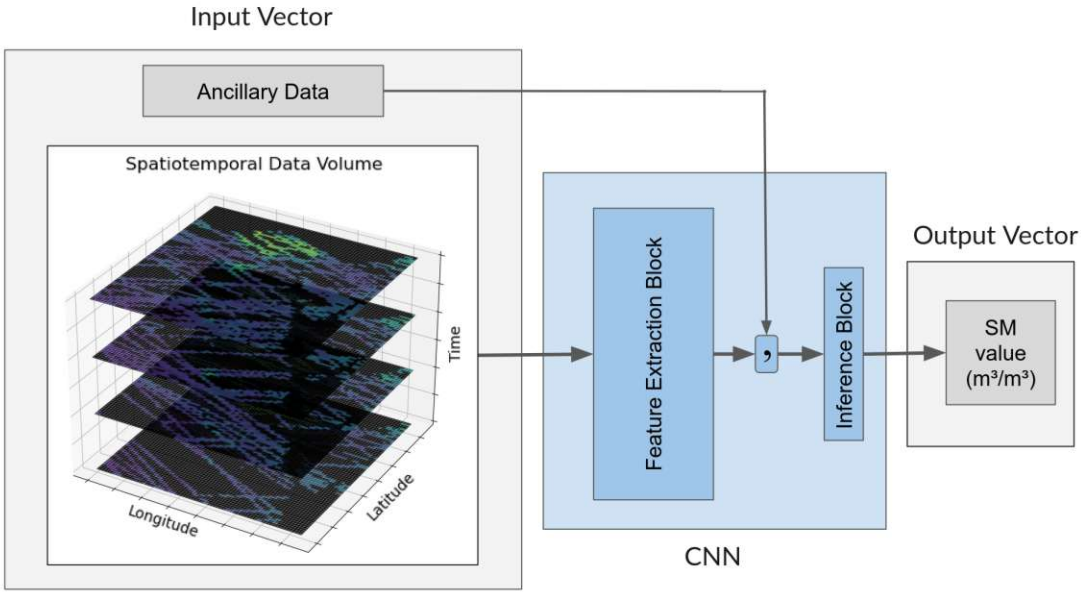


Figure 5.1: Overview of the DCSTI setup, along with its inputs and output vectors. The [,] operator denotes a concatenation of tensors

the input vector can also be a combination of two parallel inputs: the aforementioned spatiotemporal neighborhood, and an array of cell-specific ancillary data. In both cases, the output vector is simply a ground-truth value for the cell of interest.

In order to obtain ground-truths for training a DCSTI model, we could adopt the naive solution of introducing artificial gaps across the already gap-ridden datacube, and then using the known values of those artificial gaps as output vectors. However, in order to make the corresponding spatiotemporal inputs similar to a real-world scenario, those gaps can not have arbitrary shapes. Rather, they would have to resemble the gaps naturally caused by a lack of enough satellite tracks to cover the entire datacube. This gap shape is not trivial to simulate, especially in areas where a large amount of tracks are overlapped.

In order to streamline this process, a more elegant solution is to extract the input and output vectors from two separate datacubes. The output datacube DC_{out}, contains data retrieved from all 8 satellites that comprise CYGNSS. The input datacube DC_{in}, on the other hand, consists of a datacube that only contains SM data retrieved from a small subset of the CYGNSS constellation, which we will refer to as a subconstellation. Both DC_{in} and DC_{out} should be three-dimensional, with the same temporal and spatial coverage, in such a way that there is a 1-to-1 spatiotemporal correspondence between any pair of cells that have the same indices in both datacubes.

While DC_{out} still contains significant spatiotemporal gaps, those are much smaller than the ones found in DC_{in}. Thus, there will be a large number of cells which contain valid SM information in DC_{out}, while being empty in DC_{in}. By querying input and

output vectors corresponding to all of those cells, we are able to build a dataset whose input vectors contain spatiotemporal volumes with realistic satellite coverage gaps, and whose output vectors contain ground-truth knowledge derived from the same satellite constellation. The process by which we obtain those datacubes will be detailed in Section 6.2.1.

By splitting both datacubes along the temporal axis into a training set, a validation set and a testing set, it is possible to conduct a classic supervised learning training and evaluation process with the DCSTI model. In Chapter 6, we describe an experiment in which we train both DCSTI and POBI on the same CYGNSS-derived dataset. In Chapter 7, the results of this experiment are analyzed.

CHAPTER 6

Experiment Design

6.1 Preliminaries

In this chapter, we will describe an experiment whose primary goal is to evaluate the performance of the proposed DCSTI algorithm in the task of performing STI on CYGNSS-derived datacubes. In this experiment, we will treat STI as a regression problem, and train both DCSTI neural networks and a POBI ensemble to solve it.

As POBI is considered the state-of-the-art solution for performing STI on GNSS-R datacubes, our aim is to use it as a benchmark against which DCSTI can be evaluated. While both algorithms have fundamental differences in their training and prediction processes, they essentially tackle the same problem. Thus, we aim to establish comparable experiment setups for both algorithms, despite the constraints that arise from their inherent differences.

The experiment will be divided into two phases. In Phase 1, we will train both POBI and DCSTI on the same training set and evaluate them against the same training set using RMSE as an error metric. In Phase 2, we will use both the trained POBI ensemble and the trained DCSTI model from Phase 1 to fully interpolate the gaps of the same CYGNSS-derived datacube. Then, as a secondary evaluation, both of those interpolated datacubes will be validated against ground-truth values derived from external data products derived from other remote sensing systems. In order to do this, we will use metrics that are recommended in the SM validation literature for comparing data from different sources, such as the ubRMSE and the Pearson Correlation Coefficient [GDA+20].

6.1.1 Choice of Observable

Beyond evaluating the performance of DCSTI, our experiment also has a secondary goal, which is producing a trained interpolation model that can more easily be adapted for usage with upcoming GNSS-R missions such as HydroGNSS by means of transfer learning.

Unfortunately, the assessment of this goal will only be possible once the HydroGNSS constellation has become operational and provided users with enough data. Thus, even though we will make some preliminary preparations for this purpose, concrete evaluations of model transferability will only be conducted in future studies, once the required data is available.

Transferring a regression model between two data domains is not a trivial task, and gets progressively more difficult based on how disjoint those domains are. Thus, one of the challenges that could arise when trying to adapt an interpolation model trained on CYGNSS reflectivity data to a new domain, such as HydroGNSS reflectivity data, is that electromagnetic backscatter observables such as the Γ measurements that were used in the original POBI paper [Che23] are highly coupled to individual sensor characteristics.

Conversely, volumetric SSM estimates retrieved from reflectivity observations made by two different sensors will be the result of different data processing pipelines, but always refer to the same physical quantity. While there are certainly a multitude of sensor-dependent and modelling-dependent uncertainties and biases embedded in SSM retrievals, the domain difference between CYGNSS-derived and HydroGNSS-derived SSM data will be much smaller than the one observed in electromagnetic backscatter data.

Thus, building upon our inference that POBI's high performance in interpolating reflectivity data is partially driven by the physical dynamics of SSM (see Section 3.2), we have decided not to perform our interpolation experiments with electromagnetic reflectivity observables such as Γ, but rather to interpolate SSM retrievals directly.

6.1.2 Spatial Resolution Scenarios

We have decided to run both phases of the experiment twice, using two different spatial resolution scenarios. The first one will employ a datacube with the spatial dimensions discretized along a 36 km regular grid. This resolution is close to the one used by coarser-resolution SM products such as those derived from SMAP and SMOS, facilitating external validation against such products in Phase 2. The second scenario will employ a datacube with a 9 km regular grid, which is closer to CYGNSS' theoretical along-track resolution of either 3.5 km or 7 km (see Section 6.2.2), but still coarse enough to prevent the spatiotemporal data volumes in DCSTI's input vectors from becoming too sparse. Additionally, this resolution is also coarse enough to ensure that the training processes of both POBI and DCSTI do not incur prohibitive computational costs.

Furthermore, this dual-grid approach allows us to analyze DCSTI's performance in interpolating SSM along two different spatial scales, which are both characterized by their own distinct spatial SM dynamics. For simplicity's sake, those two versions of the experiment will henceforth be denoted as the 9 km experiment and the 36 km experiment.

6.2 IFAC-TW Data Product

Among the several CYGNSS-derived data products that are currently available, one that is particularly well suited for our experiment setup is a secondary version of the "IFAC CYGNSS ANN SM" data product [SPP+20][SCC+22], known as "IFAC CYGNSS ANN SM Trackwise", which we will refer to as IFAC-TW for simplicity.

Unlike the main version of the "IFAC CYGNSS ANN SM" product, IFAC-TW is not provided as a datacube, but rather in a tabular format, consisting of a list of geolocated CYGNSS-derived soil moisture retrievals. Each entry in this list contains an individual volumetric SSM retrieval, as well as data on the time and location of the observation. Furthermore, each entry also contains a track identification number that indicates which of the 8 CYGNSS satellites performed the observation, hence the name "Trackwise". Table 6.1, provides a detailed description of the variables contained in each entry of this data product.

Variable	Contents
Latitude	Approximate latitude of the measurement, with a precision of up to decimals of degrees
Longitude	Approximate longitude of the measurement, with a precision of up to decimals of degrees
Date	Date in which the measurement was taken
Second of Day (SoD)	Number of seconds since the start of the day in which the measurement was taken
Satellite ID	ID of the CYGNSS satellite from which the measurement originated, ranging from 1 to 8
Soil Moisture	Volumetric soil moisture measurement in m^{-3}/m^{-3}, ranging from 0 to 1

Table 6.1: Description of the variables in the initial CYGNSS SM dataset

The IFAC-TW data product aligns well with our specific requirements. First, it contains data on SSM retrievals instead of reflectivity, which would improve the transferability of pre-trained DCSTI models, as argued in Section 6.1.1. Furthermore, the satellite identification flags allow us to extract subsets of IFAC-TW that only include observations derived from specific subconstellations of CYGNSS. Provided that we have a mechanism for converting the tabular data into SSM datacubes (see Section 6.2.1), it is trivial to obtain two subsets of IFAC-TW that can be readily converted into an input datacube DC_{in} and an output datacube DC_{out} following the criteria established in Section 5.1.2.

The IFAC-TW product has a temporal coverage ranging from August 2018 to September 2021, amounting to a total of approximately three years. In terms of spatial coverage, the observations are restricted to latitudes between $-35°$ and $35°$, as the CYGNSS orbital path was designed to cover only the intertropical regions (see Section 2.4.1).

The SSM estimates contained in IFAC-TW were retrieved from CYGNSS reflectivity data through a machine learning model. Specifically, Santi *et al.* trained a feedforward neural network to predict SSM estimates from input vectors containing a combination of Γ, SNR and location-specific ancillary data such as elevation land cover information. The output vectors were SMAP-derived SM estimates obtained from the SMAP L3 Radiometer Global Daily dataset [OC23b]. For a complete description of the training process of this model, as well as evaluation results, please refer to [SPP+20] and [SCC+22].

6.2.1 Preprocessing

As discussed in Section 5.1.2, the training of DCSTI requires two tridimensional SM datacubes, denoted as DC_{in} and DC_{out}, from which we will extract input vectors and output vectors, respectively. Both of them should have two spatial dimensions and one temporal dimension, using the same spatial grid and same timestamp quantization, and having the same extent in each dimension. Specifically, every cell from DC_{in} must have identical spatial and temporal coordinates to those of the cell with corresponding indices in DC_{out}. The key difference between them is that while DC_{out} contains SSM estimates retrieved from all 8 CYGNSS satellites, DC_{in} only contains data collected by a smaller subconstellation.

In order to make the extraction of tridimensional subconstellation datacubes flexible, we have decided to initially convert the tabular data of IFAC-TW into a four-dimensional datacube with the shape (channel, time, latitude, longitude), with the channel determining which satellite originated the measurement. In practical terms, this four-dimensional datacube can be understood as an array of eight tridimensional datacubes. Each of those contains the observations performed by a single CYGNSS satellite. Thus, a tridimensional datacube with the observations performed by an arbitrary subconstellation of CYGNSS, can be obtained by simply filtering the channel dimension to only include the indices of the desired satellites, and then computing a mean along the channel dimension. In practice, this is simply an element-wise averaging of the tridimensional datacubes corresponding to the satellites of interest.

Thus, to convert IFAC-TW to a datacube format, all entries have to be quantized along four dimensions. The quantization of the channel dimension is straightforward, as its coordinates already consist of 8 discrete values. With respect to the spatial dimensions, we have opted to bin the measurements according to the the 9 km and 36 km versions of the EASE 2.0 regular grid [BBH+12], depending on which spatial resolution scenario we are working with (see Section 6.1.2).

The time dimension has been quantized into daily bins. This decision stems from CYGNSS' data collection pattern, which involves observations that can be made at arbitrary times during the day. As SSM measurements at a given location can potentially fluctuate a lot between daytime and nighttime, the usage of 24h bins increases the chances that a pixel's timeseries contains less noisy representations of daily SSM patterns.

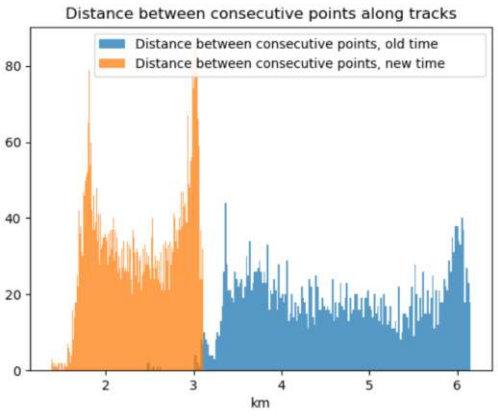
Thus, in practical terms, our 4-dimensional datacube consists of an array of 8 identically-shaped three-dimensional tensors. Every cell of each of those tensors is either empty or contains a simple mean of all SSM measurements that were collected by the corresponding CYGNSS satellite within a 9 km × 9 km or a 36 km × 36 km window, over the course of an entire day. In order to obtain a tridimensional datacube that encompasses the observations made by an arbitrary subconstellation of CYGNSS, we must simply perform an element-wise averaging of the relevant tridimensional tensors.

6.2.2 Data issues

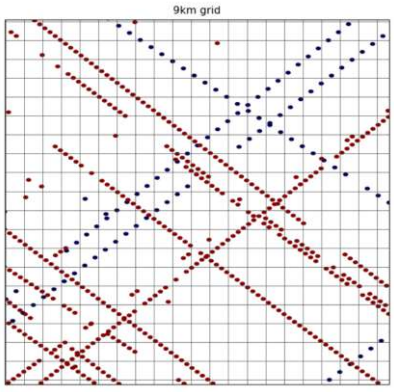
In this section, we will delve into some of the data issues, caveats and challenges that we have encountered over the course of the work with the IFAC-TW data. Some of them required additional processing steps after the conversion of IFAC-TW into a datacube format, and some of them required special precautions to be taken during the split of the data for the training of POBI and DCSTI (see Section 6.3.1).

Integration Time

When a microwave receiver embarked on a remote sensing satellite makes an observation, it collects electromagnetic radiation during a specific time period known as integration time. Due to the satellite's relative movement with respect to the observed surface, this integration time has an impact on the shape of the satellite's footprint, changing its length in the along-track direction and consequently impacting the along-track spatial resolution.



(a) Histogram of the distances between consecutive observations within a track, aggregating data from two separate days. In orange, a day with the old integration time. In blue, a day the new integration time.



(b) An example of the sampling rates provided by both integration times over the same area. The blue tracks are from a day with the old integration time, while the red tracks are from a day with the new integration time.

Figure 6.1: Changes in the spatial distribution of CYGNSS observations between the two integration times

In July 2019, the integration time of the CYGNSS satellites was halved from 1 s to 0.5 s. As a consequence, the theoretical along-track footprint of the sensors was reduced from 7 km to 3.5 km (see Figure 6.1). This alteration of the average along-track sampling rate of CYGNSS can impose a challenge on the training process of DCSTI, as it affects the overall spatiotemporal distribution of valid pixels within the data volumes used as input vectors, making the problem domain harder to model. In order to mitigate this issue, we have carefully designed the training, validation and testing sets in a way that they include data collected under both integration time regimes.

Constellation Size

In November 2022 an unexpected communication failure of satellite FM06 has reduced the number of active CYGNSS satellites from eight to seven. This has directly affected the average temporal resolution of CYGNSS data, as well as the constellation's spatial coverage patterns. As the current version of the IFAC-TW product only covers the period between August 2018 and September 2021, this disruption has no impact on our experiments. Thus, we only mention this incident here for the sake of completeness.

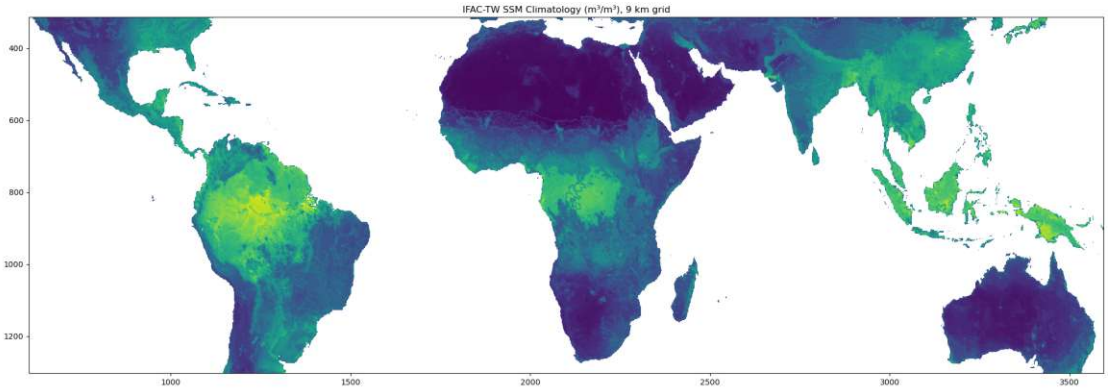
Filtering of Invalid Pixels

In order to mitigate several data quality issues, we have developed a pixel validity mask that filters out observations from certain pixels based on latitude, hydrology or climatic conditions. In practice, it just replaces any non-empty cells in those locations with empty values.

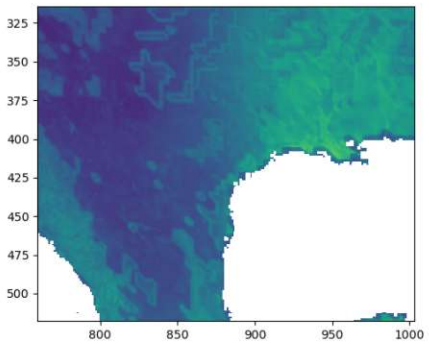
Due to physical limitations, microwave remote sensing systems are unable to reliably measure the water content of soil obstructed by snow or ice cover. Thus, in order to ensure the overall data quality and validity, cells that contain snow or ice cover should always be masked out of the SM datacubes. As the data we are working with is restricted to the latitudes between $-35°$ and $35°$, the vast majority of cells obstructed by snow or ice are located in regions with high elevation, such as the Andes or the Himalayas. In order to simplify the snow and ice masking process, we have opted to filter out from the 4-dimensional CYGNSS datacube any observations made at locations with an elevation higher than 3000 m.

In a similar vein, the concept of soil moisture is inherently ill-defined for observations collected over inland water bodies, as the observed surface does not in fact constitute soil. In practice, CYGNSS observations made over inland water bodies such as rivers and lakes tend to have a very high reflectivity value, which data retrieval models can erroneously interpret as elevated soil moisture levels. In order to mitigate the amount of those problematic retrievals in our data, we have filtered out from the datacube all the cells corresponding to pixels where more than 20% of the area is covered by permanent water bodies. This same filtering rule also effectively drops any pixels over oceans, or coastline pixels predominantly covered by the sea.

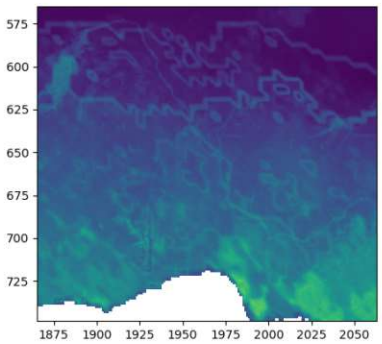
Retrieval Artifacts



(a) Global IFAC-TW SSM climatology, computed from the entire time span of a datacube with a 9 km grid and data from the entire CYGNSS constellation



(b) Close-up of artifacts in North America



(c) Close-up of artifacts in Western Africa

Figure 6.2: The IFAC-TW SSM climatology contains multiple artifacts that do not match with known real-world SSM spatial patterns, with sharp discontinuities that manifest as polygons or straight lines spanning hundreds of kilometers

When performing studies involving environmental variables such as soil moisture, it is standard practice to compute long-term statistical moments of the variable. As part of a preliminary exploratory data analysis of IFAC-TW, we have computed a pixelwise temporal mean of SSM across the entire dataset, which in the context of Earth sciences is often referred to as a "climatology" of the variable. The main purpose of computing this climatology was to perform an informal visual examination of the SSM patterns across the globe, aiming to verify if they match the patterns observed in the climatologies of well-established SM data products.

Through this visual analysis of the IFAC-TW climatology, we have ascertained that most regions exhibited spatial SSM patterns that matched our expectations. However, we have encountered multiple instances of anomalous and abrupt spatial fluctuations of average

SSM that, when plotted on a map, ultimately manifest as large straight lines or arbitrary geometric shapes spread across vast swaths of land, not resembling any natural features that can influence SSM patterns (see Figure 6.2).

This type of artifact is commonly caused by biases introduced by incorporating categorical data in the data retrieval algorithm, such as discretized land cover information. In our specific case, this is also likely to be the origin of the artifacts, as the SSM estimates contained in IFAC-TW were retrieved from reflectivity data through a neural network that includes land cover data in its input vectors. Due to the black-box nature of this retrieval algorithm, isolating and mitigating this bias in the data would pose a significant challenge.

Therefore, we have decided to treat those artifacts as inherent errors in the data, acknowledging them, within the limited bounds of our experiment, as parts of the ground-truth. Thus, we will refrain from any attempts to steer the models away from fitting to those errors, as that would stray the learning away from what we consider to be the ground-truth.

6.3 Experiment Design - Phase 1

In Phase 1, we will perform a classic machine learning experiment to compare the performance of two algorithms, namely POBI and DCSTI, in the execution of the same spatiotemporal interpolation task. In essence, this task consists of solving an individual regression problem instance for each empty pixel of a tridimensional CYGNSS SSM datacube containing spatiotemporal gaps.

In order to evaluate the regression performance of both algorithms, we have split the four-dimensional IFAC-TW datacube along the temporal dimension into a training slice, a validation slice and a test slice. Both algorithms have been trained exclusively on data from training slice, and had their performance measured on data from the test slice according to the RMSE criterion. The validation slice has been used for hyperparameter tuning. In the case of DCSTI, it was also used as a tool for assessing convergence and overfitting during the training of neural networks

Both POBI and DCSTI operate in an analogous fashion when performing inference, having the same output domain and effectively solving the same problem of producing a fully interpolated tridimensional datacube. However, those two algorithms have fundamental differences in their input vectors at both training and prediction time.

The training of POBI only requires a single datacube, from which the historical spatiotemporal relationships between all relevant pair of pixels are extracted. This datacube can also be the source of the testing data used to evaluate the POBI ensemble, provided that training and testing data are extracted from disjoint temporal slices of the datacube.

Conversely, the training of DCSTI requires the usage of two datacubes DC_{in} and DC_{out}, with input vectors being extracted from DC_{in} and output vectors being extracted from

DC_{out}. As outlined in Section 5.1.2, DC_{in} must be derived from a subconstellation with a lower cardinality than the one from which DC_{out} is derived. This requirement of two separate datacubes applies to training, validation and testing data.

In order to run a single inference at prediction time, the only input that POBI requires is a two-dimensional array containing the known values surrounding the cell of interest at its exact temporal index. DCSTI, on the other hand, utilizes a tridimensional tensor extracted from the vicinity of the cell of interest.

As part of the effort to keep the experiment setup as consistent as possible between POBI and DCSTI, we seek to provide both algorithms with the same amount of ground-truth knowledge at training time. Therefore, we train the POBI models using the same DC_{out} as the one used to extract the output vectors used in the training of DCSTI. Specifically, we will use all eight CYGNSS satellites to generate this DC_{out}, thus maximizing the ground-truth information available to both algorithms. This specific datacube will henceforth be denoted as DC_{full}.

While the training of DCSTI could in principle be conducted with a DC_{in} derived from a CYGNSS subconstellation with any cardinality between 1 and 7, our focus will be on the scenario where the subconstellation has a cardinality of 2. The rationale behind this decision is that we seek to closely align our experiment with the use case of the HydroGNSS constellation, which will be comprised of a total of 2 satellites.

As mentioned in Section 6.1.2, the entire experiment will be conducted twice, using two separate datacubes derived from IFAC-TW, using two different spatial grids for discretizing the spatial dimensions. They are, respectively, the 9km and the 36km versions of the EASE 2.0 grid.

In both resolution scenarios, the evaluation of the trained POBI and DCSTI models has been conducted by having them fully interpolate an IFAC-TW datacube derived from a subconstellation with a cardinality of 2. Then, the RMSE of the testing slice was computed with respect to the same slice from DC_{full}.

The subconstellation used in this evaluation process is specifically composed by the CYGNSS satellites FM03 and FM04. Those satellites were selected randomly, as a preliminary analysis of statistical moments from several IFAC-TW datacubes with a cardinality of 2 revealed that all satellite pairs yield data with the same overall distribution. The tridimensional CYGNSS SSM datacube derived from FM03 and FM04 will henceforth be denoted as $DC_{3,4}$

6.3.1 Data Split

The IFAC-TW dataset encompasses approximately three years of data, ranging from August 2018 to September 2021. When temporally partitioning an IFAC-TW-derived datacube into training, testing and validation sets, a common strategy would be to define a fixed proportion of IFAC-TW's time span to be assigned to each subset. Subsequently, the time indices could be randomly allocated according to those proportions, employing

a very fine or even individual granularity. However, for this particular experiment, we cannot adopt such a strategy for a variety of reasons.

Firstly, the training of POBI involves establishing correlations between long-term SSM timeseries of neighboring pixels, and thus the training set must consist of a long sequence of consecutive days.

Secondly, the input vectors of DCSTI contain spatiotemporal volumes that must cover several days along the time dimension, meaning that the splitting granularity must be coarse enough that such input volumes can be entirely contained within slices of either subset.

Thirdly, we want all three slices to be equally distributed across all seasons of the year, as SSM patterns are known to exhibit a high degree of seasonality.

Lastly, due to the changes that happened to the CYGNSS integration time in 2019 (see Section 6.2.2), we have two distinct spatial distributions of observations along the IFAC-TW time period, and it would be important to account for this variability in all three data subsets.

Therefore, while recognizing that the usage of a fixed data split can introduce some level of bias to the experiment results, we have opted for a compromise where we manually construct a fixed. This decision stems from an effort to find a balance between potential bias and the need to address all of the constraints listed above.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2018								O	O	O	O	O
2019	O	O	O	O	O	O	N	N	N	N	N	N
2020	N	N	N	N	N	N	N	N	N	N	N	N
2021	N	N	N	N	N	N	N	N	N			

Key:	
Train	O: Old Integration Time
Validation	N: New Integration Time
Test	
Dropped	

Figure 6.3: Diagram of the data split employed in Phase 1. The data was split temporally, with a monthly granularity, seeking to ensure a balanced distribution of integration time and temporal seasonality across all three subsets.

This split, shown in Figure 6.3, allocates the entirety of 2019 and 2020 as the training set. The remaining time has been partitioned at a monthly granularity between the testing and validation sets, following a ratio of 2:1. The testing and validation months are alternated in order to ensure that both of those subsets have samples spread across all seasons. All three subsets have approximately a quarter of their time indices derived from the period characterized by the old CYGNSS integration time. The months of

August and September from 2019 have been dropped from the data in order to facilitate the construction of this manual split.

6.3.2 Phase 1 - POBI

As outlined in Section 3.3.1, the POBI algorithm has three main hyperparameters: the Neighborhood Length (l_n), the Concurrency Window (w_c), and the Minimum Concurrency Threshold ($c_{\min}$). As training POBI with a single hyperparameter configuration is already computationally expensive, especially in the case of the 9 km experiment, we have opted for using a fixed value of 1 day for w_c and a minimum of 3 points for $c_{\min}$.

Then, in both the 9 km and 36 km experiments, the POBI ensemble was trained according to the procedure described in Section 3.3.1, using the following values for l_n: {3, 5, 7, 9, 11, 13, 15, 17}. As specified in Sections 6.3 and 6.3.1, the training data for POBI was derived from 2019 and 2020, using DC_{full} as a data source.

In order to optimize the value of l_n, those 8 hyperparameter configurations of the POBI ensemble were used to fully interpolate the validation months of CYGNSS SSM datacubes derived from 2 satellites. We then computed the average RMSE of the interpolated validation set with respect to the corresponding dates in DC_{full}. In both the 36 km and 9 km cases, this analysis revealed $l_n = 9$ to be the most favorable choice within the examined range. Higher values of l_n yielded no significant improvement with respect to RMSE on validation data, and substantially increased the number of parameters required by the POBI ensemble.

Finally, in both the 9 km and 36 km cases, the $l_n = 9$ version of the POBI ensemble was used to interpolate $DC_{3,4}$. Then, the RMSE between the test slice of those interpolated datacubes and the test slices of the corresponding versions of DC_{full} were computed. For detailed analysis of those results, please refer to Section 7.2.

6.3.3 Phase 1 - DCSTI

As mentioned in Section 6.1.1, this study has a secondary goal of producing an STI model that can be adapted for the upcoming HydroGNSS mission by means of transfer learning. Thus, in order to approach the observation distribution of HydroGNSS, we have focused both the 36 km and 9 km DCSTI experiments on the case where DC_{in} is derived from a subconstellation with cardinality 2, while DC_{out} is derived from the entire CYGNSS constellation.

In order to increase the amount of data for the training and validation sets, we extract input vectors from multiple parallel realizations of DC_{in}, employing all possible pairs of CYGNSS satellites. Thus, each non-empty cell of DC_{full} might be used as an output vector in combination with multiple different input vectors extracted from different realizations of DC_{in}.

As shown in Figure 6.4, the DCSTI experiment pipeline in Phase 1 starts with querying all cases of empty cells in any realization of DC_{in} with a corresponding non-empty cell

in DC_{full}. From those cells, we extract input vectors comprised of their spatiotemporal surroundings and output vectors comprised simply of the known value form DC_{full}. If the appropriate hyperparameters have been toggled, the input vectors also include an array of ancillary information.

Those input/output pairs are then split into training, testing and validation sets according to the criteria established in Section 6.3.1. Once the data split is completed, we proceed to train multiple CNN regressors to map from the input vectors to the output vectors, with each of those models having its own unique hyperparameter configuration.

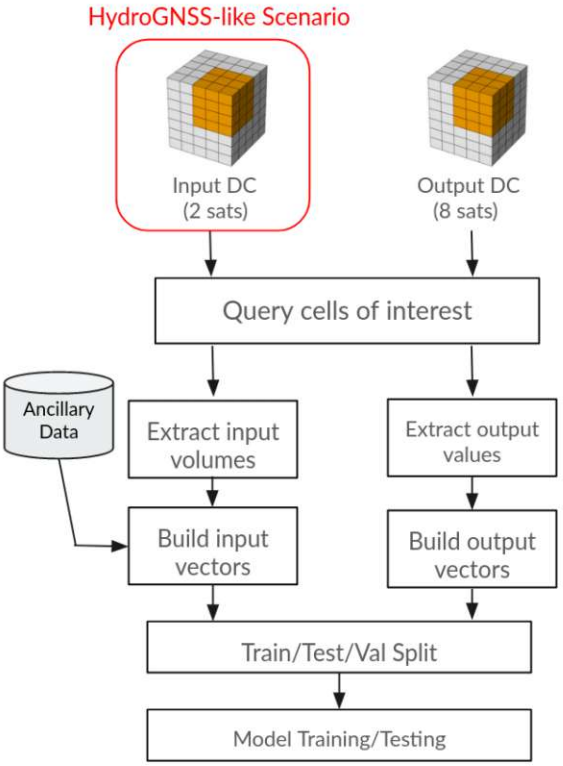


Figure 6.4: Flowchart of the DCSTI modelling pipeline in Phase 1

All models were trained from scratch, without employing transfer learning from pre-trained models designed for general-purpose image processing tasks. This decision stems from the fact that the target domain of this study is significantly different from general imaging domain in terms of underlying data semantics. Moreover, in most cases, our input data volumes have more than three channel dimensions, which in the imaging domain usually refer to RGB channels, and in our domain correspond to time indices.

To evaluate the performance of each hyperparameter configuration, we have employed a classic supervised learning setup for neural network training: after each model reached convergence on periodic evaluations against subsets of the validation set, they were

evaluated against the entirety of the validation set according to the RMSE.

Finally, just like we did for POBI, the DCSTI models that achieved the highest performance in each resolution scenario were used to fully interpolate the corresponding version of $DC_{3,4}$. Then, the RMSE between the test slice of those interpolated datacubes and the corresponding test slices in both versions of DC_{full} were computed. For a detailed analysis of those results, please refer to Section 7.2.

In the following subsections, we will outline some considerations about the hyperparameter tuning process for DCSTI. For a complete rundown of the hyperparameters that yielded the best-performing DCSTI model for each spatial resolution scenario, please refer to Section 7.1.

Hyperparameter Tuning - Input Vectors

The shape of the spatiotemporal volume around the cell of interest in each input vector is an important hyperparameter, characterized by split six different values. Those values represent the number of cells to be included around the cell of interest along each direction of the three spatiotemporal dimensions. During our hyperparameter search, we focused on configurations where no cells from timestamps posterior to that of the cell of interest were included. This choice stems from an attempt to produce an interpolation model that does not rely on knowledge of the future at prediction time, and thus can be used for applications requiring near-real-time data delivery.

The inclusion of ancillary data in the input vectors, and the choice of which ancillary data sources to use, are also hyperparameters that require tuning. We have included three possible sources of ancillary data in our hyperparameter space:

- Spatial coordinates of the cell of interest, encoded as latitude and longitude values;
- Temporal coordinates of the cell of interest, encoded as the day of the year;
- Land cover information of the cell of interest, obtained from the MODIS/Terra Land Cover dataset [BK11].

Hyperparameter Tuning - Architecture

The hyperparameter space of convolutional neural networks is challenging to explore, as it contains a high number of dimensions, many of which have their existence depend on the values taken by other dimensions. For instance, it would only make sense to optimize the length of the third hidden layer of a CNN's fully-connected block if said block has a number of hidden layers greater than or equal to three. Thoroughly exploring such a hyperparameter space requires a substantial amount of computational resources, especially in cases where training a single model takes a long time. Moreover, the nondeterministic nature of SGD might require multiple training runs with a single hyperparameter configuration in order to obtain an appropriate grasp of that configuration's performance.

With the computational resources that were made available for this study, training a neural network to convergence requires approximately one full day for the 9 km experiment, and approximately 5 hours for the 36 km experiment. Thus, in light of the challenges outlined above, we have defined a limited hyperparameter space to explore for each experiment, and manually adjusted the ranges and granularity of those spaces according to improvements or setbacks that were empirically encountered along the search process.

Specifically, we have narrowed down the neural network architecture search to three options: standard CNNs, ResNets and DenseNets. For each architecture, a few different values were attempted for hyperparameters such as the number of layers and growth rate. The model architectures we used in this study were either implemented from scratch using the PyTorch library [PGM+19] or adapted from the open-source architecture implementations provided by the same library, modifying them to better suit our specific needs.

6.4 Experiment Design - Phase 2

In Phase 2 of the experiments conducted in this study, we seek to validate the four fully-interpolated versions of $DC_{3,4}$ against external soil moisture data products. In order to establish a baseline, we have also validated both versions of DC_{full} against the same data. Thus, Phase 2 involves a total of six datacubes, including:

- The 36 km version of $DC_{3,4}$, fully interpolated by the best-performing 36 km POBI model;
- The 36 km version of $DC_{3,4}$, fully interpolated by the best-performing 36 km DCSTI model;
- The 36 km version of DC_{full}, without any interpolation;
- The 9 km version of $DC_{3,4}$, fully interpolated by the best-performing 9 km POBI model;
- The 9 km version of $DC_{3,4}$, fully interpolated by the best-performing 9 km DCSTI model;
- The 9 km version of DC_{full}, without any interpolation.

For this phase, we no longer make the distinction between training, validation and testing slices, but rather evaluate the entire interpolated datacubes against external data.

The rationale for also validating those datacubes against external data lies in the still experimental and evolving nature of GNSS-R technology for monitoring soil moisture. Although our primary focus is improving the quality of GNSS-R data products by means of spatiotemporal interpolation, we also seek to contribute to a deeper understanding of

the inherent issues contained in the GNSS-R data itself. We acknowledge, for instance, that the data we regard as ground-truth within the bounds of Phase 1 might contain an unexpectedly high level of error with respect to the real-world observables, as hinted by the artifacts described in Section 6.2.2.

Thus, in order to provide a secondary means of model evaluation, we resort to external data derived from other more mature remotely sensed SM data products, which have already undergone more extensive modeling, calibration and validation studies, and are thus likely to carry a lower level of uncertainty and deviation from the real-world values. This comparison could, in theory, allow us to better assess the degree to which POBI and DCSTI improve or degrade the quality of the CYGNSS data, and how the uncertainties found in the interpolated data compare with the uncertainty that is already embedded in the ground truths used to train those models.

All six of the validation experiments of Phase 2 have been conducted within QA4SM [GDA+20], an online platform for validation and quality assurance of SM data products which is available at `https://qa4sm.eu` (last accessed on March 20, 2024). With the aid of QA4SM, we can compute time-oriented validation metrics for each pixel, the corresponding timeseries between multiple datacubes. We are mainly concerned with two metrics, which are commonplace in the SM validation literature [GDA+20].

The first one is the ubRMSE (Equation 6.1), which seeks to compute the RMSE between two corresponding timeseries derived from different sensors after removing the bias that exists between both sources.

$$\text{ubRMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[(X_i - \bar{X})(Y_i - \bar{Y})]^2} \tag{6.1}$$

The second one is the Pearson correlation coefficient (Equation 6.2), which seeks to quantify the strength of a correlation between two timeseries.

$$r = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} = \sum_{i=1}^{N} \frac{(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{N}(X_i - \bar{X})^2 \sum_{i=1}^{N}(Y_i - \bar{Y})^2}} \tag{6.2}$$

For a complete summary of the results of Phase 2, please refer to Section 7.3.

CHAPTER 7

Results and Discussion

7.1 Final Hyperparameter Configurations

In this section, we will briefly describe the hyperparameter configurations of the models that were picked during the hyperparameter search of Phase 1 to be evaluated on test data. Those models are not necessarily the ones that displayed the best possible RMSE on validation data, but rather those that have struck the most balanced compromise during our hyperparameter exploration, combining a validation RMSE on par with the overall best-performing models while requiring a reasonably small number of parameters.

7.1.1 POBI Hyperparameters

After conducting the hyperparameter search procedure described in Section 6.3.2, with fixed values for w_c and $c_{\min}$, we reached the conclusion that a neighborhood length of 9 provides an ideal balance between validation RMSE and model size. While higher values of l_n provide marginal improvements in the validation RMSE, they unfortunately lead to a dramatic increase in model size, as an increase in l_n leads to a quadratic increase in the number of parameters required by the POBI ensemble. This trend of diminishing returns at $l_n = 9$ was observed in both the 9 km and 36 km experiments, and therefore the incumbent POBI hyperparameter configuration for both cases is the one shown in Table 7.1.

Variable	Value
l_n	9 pixels
w_c	1 day
$c_{\min}$	3 points

Table 7.1: Incumbent hyperparemeter configuration for POBI

7.1.2 DCSTI Hyperparameters

Similarly to the POBI case, our exploration indicated that some regions of the DCSTI hyperparameter space yield a good balance between validation RMSE and model size in both the 9 km and 36 km experiments. Although marginal improvements could be achieved by fine-tuning the architecture hyperparameters of each resolution scenario, we have opted for maintaining a certain consistency between the values used in both cases, keeping them identical unless a specific hyperparameter change yielded significant improvements in validation RMSE.

Ultimately, as shown in Tables 7.3 and 7.2, the main difference between the incumbent architectures that were selected for the 9 km and 36 km cases lies in the shape of the data volumes contained in the input vectors. This also has an indirect impact on the dimensions of the feature maps used throughout the dense blocks, as their widths and heights are not defined explicitly, but rather as a function of the input volume's width and height.

In Tables 7.3 and 7.2, the shape of the input volume is described by the number of cells that are extracted from around the cell of interest in each direction. They are listed in the following order: number of cells in the past, number of cells in the future, number of cells to the west, number of cells to the east, number of cells to the north, number of cells to the south.

For both resolution scenarios, we have focused our hyperparameter exploration on input volume shapes that do not include any pixels from the future, thus allowing the trained models to be used for applications that require near-real-time data delivery. In the case of the 9 km experiment, we sought to use input volumes with longer widths and heights along the spatial dimensions, as the pixels in the 9 km grid correspond to a smaller land area than the pixels in the 36 km grid.

Parameter	**Value**
Input Volume Shape	[14,0,8,8,8,8]
Ancillary Data	None
Architecture	DenseNet
Dense Block Sizes	[2,4,8,4]
Growth Rate	32
Fully Connected Layers	[512, 1]

Table 7.2: Incumbent hyperparameter configuration for DCSTI: 36 km Experiment

A noteworthy finding of our hyperparameter exploration was that ResNets can achieve a validation RMSE on par with that of the DenseNet configurations listed above, but only when using input vectors that include ancillary data such as spatial coordinates and land cover information. Thus, even though ResNets usually require fewer parameters than DenseNets, in the context of this problem this model size advantage would come at the cost of including additional data streams. Ultimately, we opted for using an architecture

Parameter	Value
Input Volume Shape	[14,0,14,14,14,14]
Ancillary Data	None
Architecture	DenseNet
Dense Block Sizes	[2,4,8,4]
Growth Rate	32
Fully Connected Block Sizes	[512, 1]

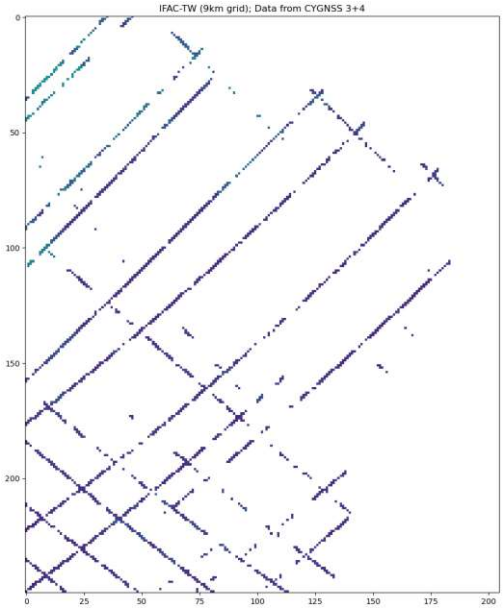
Table 7.3: Incumbent hyperparameter configuration for DCSTI: 9 km Experiment

that requires a higher number of parameters, but offers a simpler data pipeline that minimizes data dependencies.

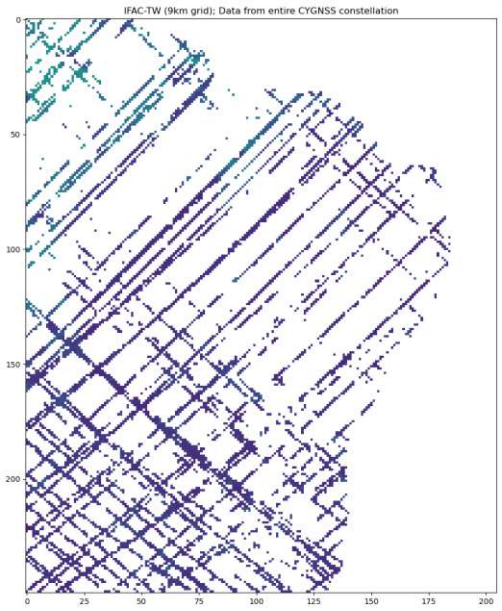
7.2 Phase 1 Results - RMSE on test data

In this section we will discuss the performance of the trained models described in Section 7.1 when performing spatiotemporal interpolation of cells in the test slice of $DC_{3,4}$, which is derived exclusively from the IFAC-TW observations collected by the CYGNSS satellites FM03 and FM04, as outlined in Section 6.3. We measure this performance by computing the element-wise RMSE between the fully interpolated $DC_{3,4}$ and a ground-truth datacube DC_{full} which is derived from the entire constellation. Since DC_{full} has not undergone any interpolation, it still retains some spatiotemporal gaps, and thus the RMSE is only computed at cells for which there is valid ground-truth information available.

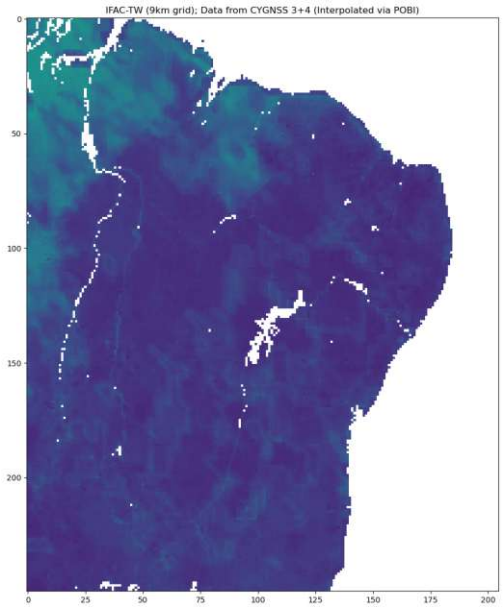
Tables 7.5 and 7.4 present the test RMSE results for the 9km and 36km experiments, respectively. We also include the number of parameters required by each model, along with the order of magnitude of this same number in order to simplify the table's readability. In the case of POBI, the number of parameters is computed according to Equation 3.3, which defines the lower bound for the number of parameters required by a POBI ensemble. In the case of DCSTI, we directly report the number of trainable parameters contained within the neural networks selected for evaluation. In the histograms contained in Figure 7.3, we display the pixel-wise RMSE distributions of both methods.



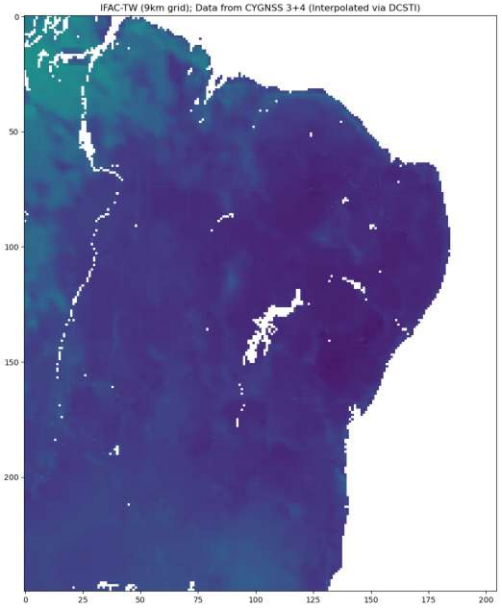
(a) Timeslice of $DC_{3,4}$ (9 km)



(b) Timeslice of DC_{full} (9 km)



(c) Timeslice of $DC_{3,4}$ (9 km), interpolated via POBI



(d) Timeslice of $DC_{3,4}$ (9 km), interpolated via DCSTI

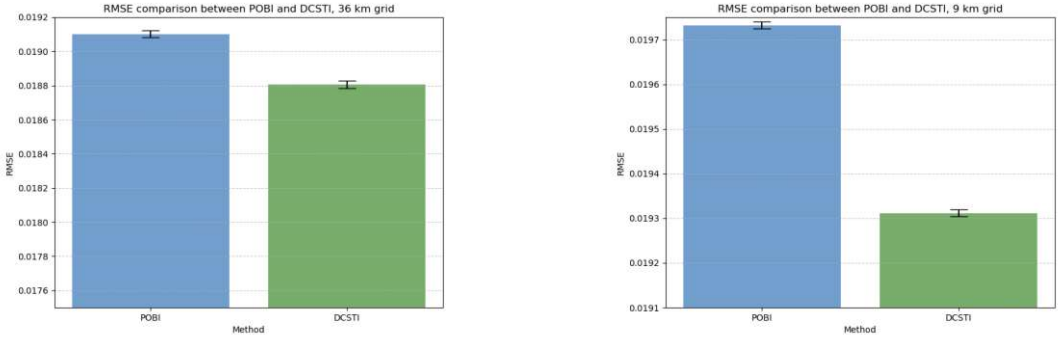
Figure 7.1: Comparison of the same timeslice across all four datacubes involved in the 9 km scenario of Phase 1. In order to facilitate the visualization of the non-interpolated tracks, we have zoomed into a specific region in northeastern Brazil

Method	RMSE	Number of Model Parameters, Total	Number of Model Parameters, Order of Magnitude
POBI	0.017318	13,909,920	10^7
DCSTI	0.016842	1,802,913	10^6

Table 7.4: Performance and Model Size Comparison for 36 km Experiment

Method	RMSE	Number of Model Parameters, Total	Number of Model Parameters, Order of Magnitude
POBI	0.018894	226,498,320	10^8
DCSTI	0.018156	2,581,153	10^6

Table 7.5: Performance and Model Size Comparison for 9 km Experiment



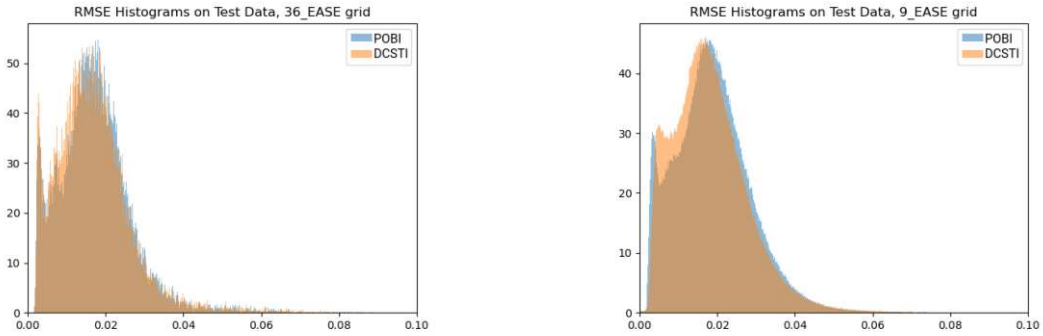
(a) RMSE on test data, 36 km scenario

(b) RMSE on test data, 9 km scenario

Figure 7.2: Comparison of the RMSE of both POBI and DCSTI on test data, in both spatial resolution scenarios

Through an analysis of the values in Tables 7.4 and 7.5 and the distributions depicted in the histograms in Figure 7.3 it becomes evident that, in both experiments, the chosen DCSTI model was able to outperform the chosen POBI ensemble with respect to the RMSE on test data. Figure 7.2 further illustrates this point, showing non-overlapping 95% confidence intervals for the RMSE in both experiments, indicating that there is a statistically significant difference in the performance of both models according to this criterion. The statistical significance of this difference was further confirmed through a paired t-test, with p-values smaller than 0.001.

Nevertheless, it is important to recognize that the performance of both DCSTI and POBI can be very sensitive to hyperparameter choices and data quality issues, and there is a possibility that the most favorable regions of the hyperparameter space have not been explored in either method. Thus, despite observing a statistically significant performance difference between both methods in this specific experiment, we consider this difference



(a) Pixel-wise RMSE on test data, 36 km scenario

(b) Pixel-wise RMSE on test data, 9 km scenario

Figure 7.3: Histogram of the pixel-wise RMSE of POBI and DCSTI on test data, in both spatial resolution scenarios

to be small enough that it could potentially be bridged or even reversed by means of a more thorough hyperparameter optimization process.

Consequently, while our findings suggest that DCSTI can achieve a lower RMSE than that offered by POBI in this problem instance, we opt for exercising caution in making strong assertions about this performance difference. Therefore, we conclude that both methods have demonstrated the potential to achieve a comparable performance in addressing the spatiotemporal interpolation problem with the IFAC-TW dataset.

However, despite having only achieved an equivalent performance to that of POBI with regards to the test data RMSE, DCSTI was remarkably successful in mitigating one of the major drawbacks of POBI, which is the requirement of a high number of parameters in order to model the spatiotemporal dynamics of SSM. Specifically, DCSTI requires 1 order of magnitude fewer parameters than POBI in the 36 km scenario, and 2 orders of magnitude fewer parameters in the 9 km scenario. As the number of parameters required by a POBI ensemble depends on both the width and height of the spatial grid that is used in the datacube of interest, this gap in model size between the two methods is likely to increase even further if we apply them to datacubes that employ spatial grids with higher resolutions. Thus, we conclude that DCSTI is able to provide a significant advantage over POBI in terms of model size and resource requirements.

Moreover, as DCSTI makes use of a neural network to solve the STI problem, it carries the potential for a quick adaptation to novel, albeit similar, problem domains by means of transfer learning. This could, for instance, be exploited in the development of data products derived from HydroGNSS SSM retrievals, once the satellites have become operational. POBI, in contrast, would require the accumulation of a long data record from the sensor before it can be trained to interpolate datacubes derived from it. Regardless, while the HydroGNSS data is still not available for transfer learning experiments, this

assertion remains a conjecture.

7.2.1 Spatial Distribution of Errors

While we refrain from making strong statements about DCSTI generally outperforming POBI, we hypothesize that DCSTI may be more robust to fitting to some of the inherent errors that are observed in the ground-truth data. This conjecture stems from the notion that, since POBI's modeling of spatiotemporal SSM patterns is heavily location-centric, it would be more prone to encode pixel-specific artifacts such as the ones described in Section 6.2.2. DCSTI, on the other hand, seeks to model a more location-agnostic representation of the spatiotemporal SSM dynamics, and thus could be less likely to be affected by pixel-specific issues. It should also be noted, however, that neural networks are very prone to overfitting, and thus DCSTI could still be vulnerable to this problem.

To investigate this hypothesis, we have computed the pixel-wise RMSE values for both POBI and DCSTI. This process consists of computing the RMSE between the interpolated timeseries and the ground-truth timeseries at each pixel in the datacube's spatial grid, obtaining a two-dimensional array of pixel-wise error levels. These arrays can then be projected on a map for better visibility, as shown in Figure 7.4. In this text, we will only display the maps corresponding to the 9 km experiment, as their higher resolution allows for a clearer depiction of spatial patterns than that offered by their 36 km counterparts.

In both the POBI and DCSTI error maps, the artifacts described in Section 6.2.2 are easily visible, suggesting that both algorithms are somewhat robust to the inclusion of such errors in their modeling of SSM dynamics. However, as shown in Figure 7.5, the artifacts are more clearly visible in the DCSTI error map, indicating that the DCSTI-interpolated datacubes exhibit a greater departure from the ground-truth values of those locations. This suggests that DCSTI has been more effective in avoiding the inclusion of such undesirable artifacts in its modeling of SSM dynamics.

It should be noted that the error intensities depicted in pixel-wise RMSE maps should not be interpreted at face value as a measure of the model's performance across different macro-regions. This is because regions with low SSM variance will lead the models to make small prediction errors, whereas regions with higher SSM variance may lead the model to make more significant errors. It would not be meaningful, for instance, to compare a model's performance over the Sahara desert, which is characterized by a low SSM variance, with its performance in more humid regions such as the Amazon rainforest, where SSM patterns can fluctuate intensely and rapidly due to rainfall events and other environmental phenomena. Nevertheless, despite their limitations, pixel-wise RMSE maps remain valuable for discerning which are the most problematic areas within the scope of bounded regions with mostly homogeneous environmental conditions.

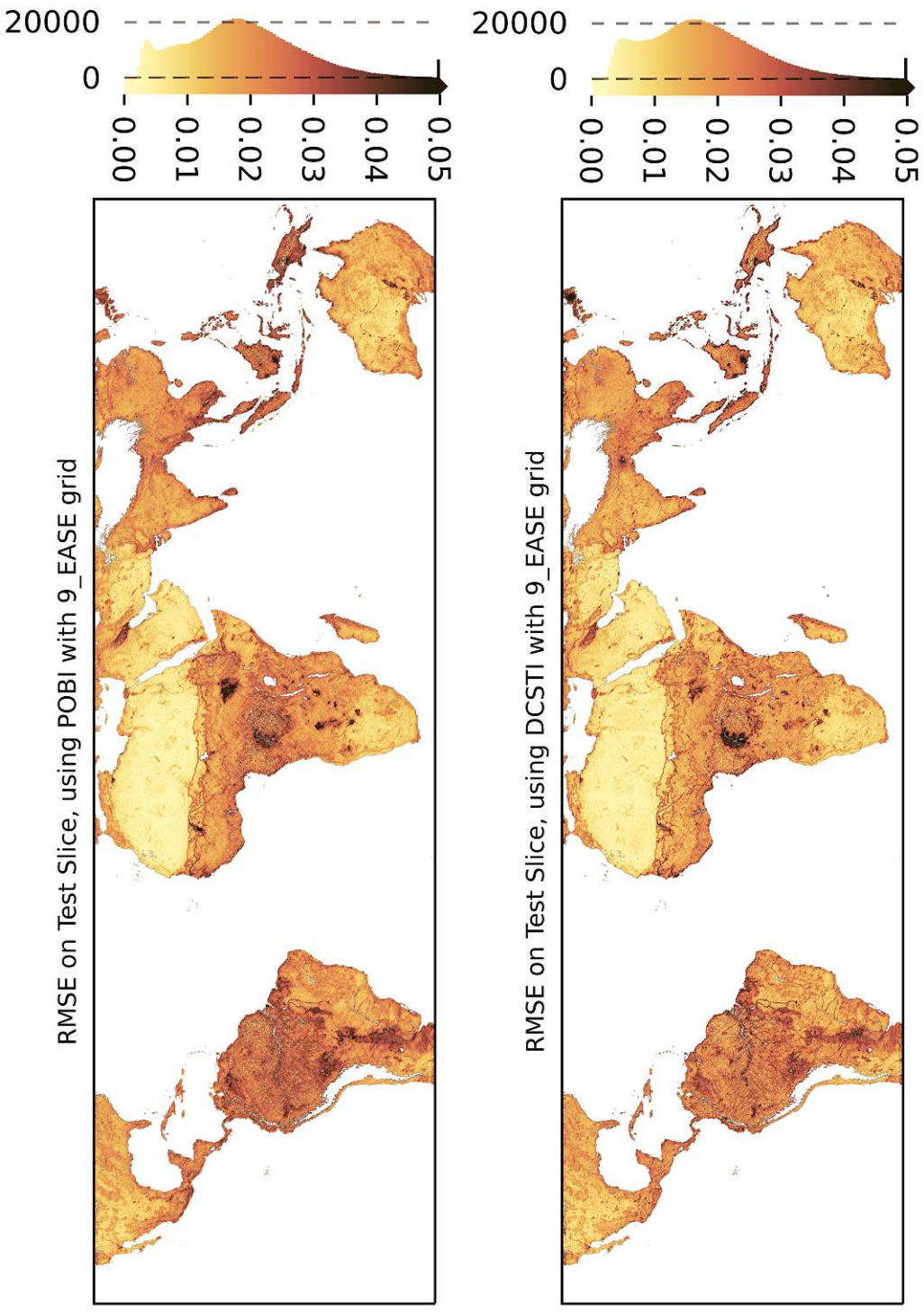
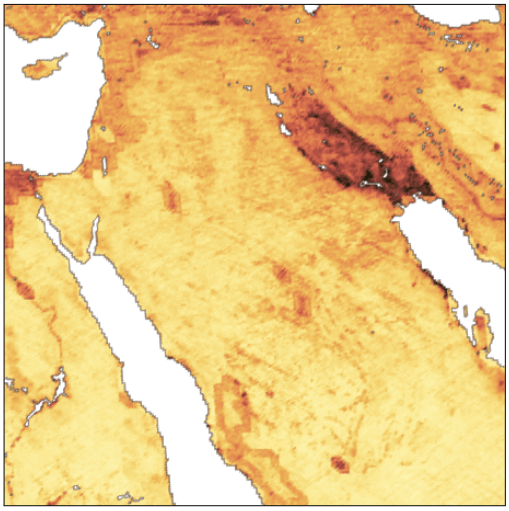
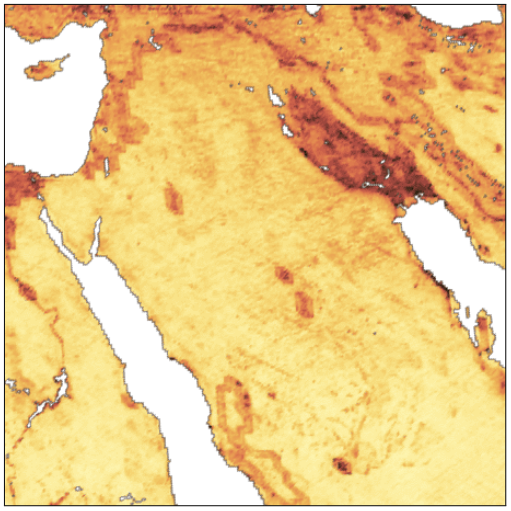


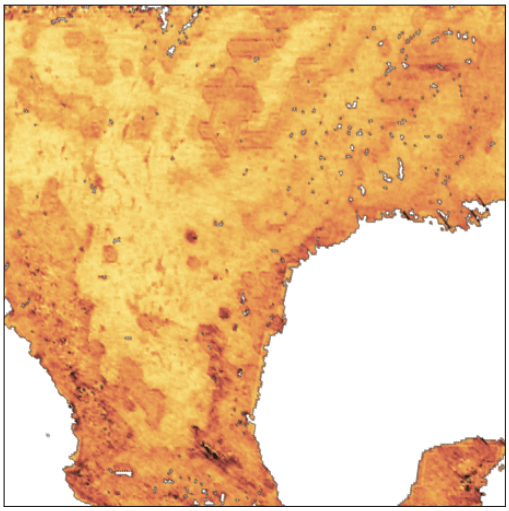
Figure 7.4: Pixe-lwise RMSE of POBI-interpolated (left) and DCSTI-interpolated (right) versions of $DC_{3,4}$ with respect to DC_{full}



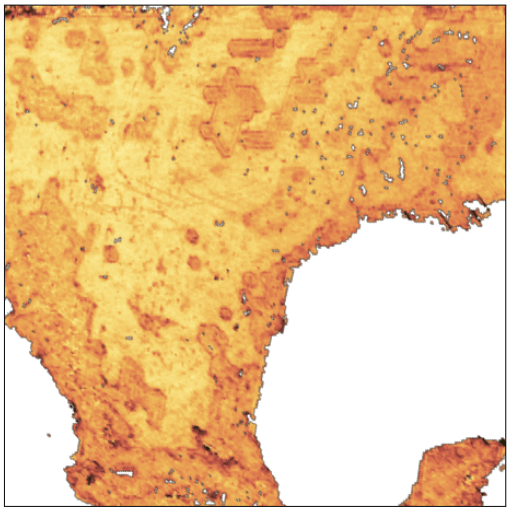
(a) Zoomed-in pixel-wise POBI RMSE on test data over parts of the Middle East



(b) Zoomed-in pixel-wise DCSTI RMSE on test data over parts of the Middle East



(c) Zoomed-in pixel-wise POBI RMSE on test data over parts of North America



(d) Zoomed-in pixel-wise DCSTI RMSE on test data over parts of North America

Figure 7.5: Zoomed in pixel-wise RMSE maps over two regions that contain a particularly high density of the ground-truth errors described in Section 6.2.2. The left column displays the pixel-wise RMSE maps for the POBI-interpolated datacubes. The right column displays the pixel-wise RMSE maps for the DCSTI-interpolated datacubes.

7.3 Phase 2 Results - Validation against external data

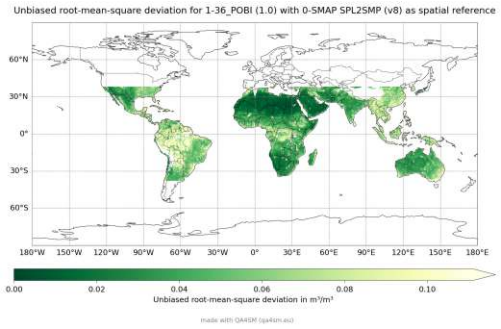
For Phase 2 of this study, we have conducted a secondary evaluation experiment in which we have validated a total of six CYGNSS SSM datacubes derived from external data sources. Specifically, for both spatial resolution scenarios, we have performed an external validation of the ground-truth datacube DC_{full}, as well as the fully-interpolated datacubes produced by both POBI and DCSTI. In Phase 2, we perform a validation along the entirety of the time span covered by IFAC-TW, rather than restricting our analysis to the test slices defined in Section 6.3.1.

We have validated all three 36 km datacubes against SM data derived from SMAP. Specifically, we have used the SMAP SPL2SMP data product [OC23a], which employs the same 36 km version of the EASE 2.0 Grid that has is employed by our 36 km CYGNSS SSM datacubes. For the 9 km case, we have validated all three datacubes against SSM estimates from ERA5-Land [MSDAP+21], a dataset produced by a meteorological reanalysis model. ERA5-Land also offers a nominal spatial resolution of 9 km, but employs a different grid from the one we have used in the IFAC-TW datacubes.

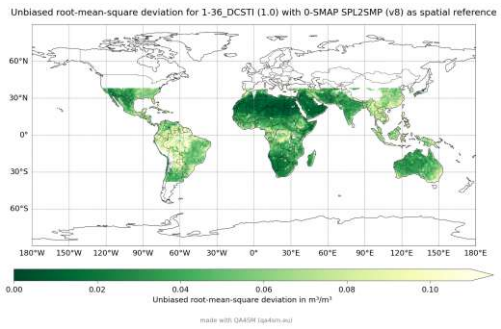
Figures 7.6 and 7.7 show the pixel-wise Pearson correlation coefficient and unbiased RMSE maps obtained in the validation of the fully-interpolated datacubes from the 36 km and 9 km spatial resolution scenarios, respectively. In both of those scenarios, the validations of POBI and DCSTI have yielded maps with very similar error intensities and spatial error patterns. This provides further evidence that a DCSTI model is able to encapsulate the same problem domain knowledge as a POBI ensemble, but in a better compressed and more resource-efficient format.

In Figure 7.8, we display the global ubRMSE and Pearson correlation coefficients of all six datacubes with respect to the external data they were validated against. In both spatial resolution scenarios, neither POBI nor DCSTI was able to provide a statistically significant improvement over the ground-truth datacube with respect to those metrics. Nevertheless, it is important to note that both methods were able to provide fully-interpolated datacubes with a negligible amount of spatiotemporal gaps, while still maintaining the same level of uncertainty that DC_{full} carries with respect to the "higher-order" ground-truths derived from more mature data products.

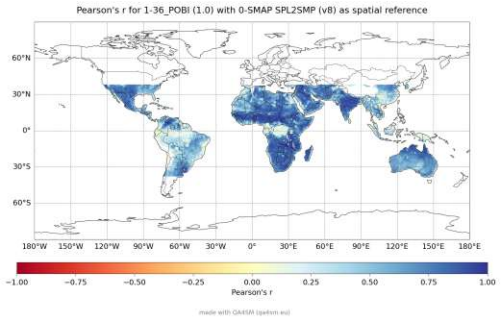
Additionally, such improvement of spatial coverage and temporal resolution, which was achieved without compromising data quality, has been obtained by applying POBI and DCSTI to $DC_{3,4}$, which only contains observations made by 25% of the CYGNSS constellation. While it is important to acknowledge that the development of the models that are able to generate such high-quality interpolated datacubes from $DC_{3,4}$ relies on ground-truth knowledge extracted from the entire constellation, such ground-truth knowledge could also be derived from external sources, as long as inter-sensor bias issues are properly addressed during data preprocessing. Thus, this result suggests that we are well-positioned for developing HydroGNSS interpolation models once the HydroGNSS satellites becomes operational.



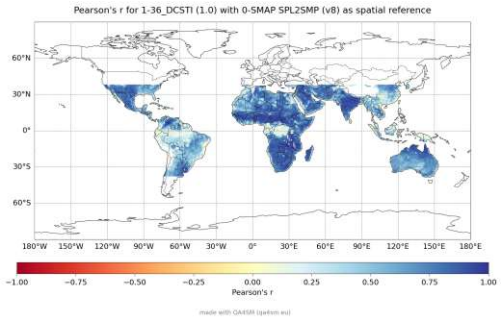
(a) Pixel-wise ubRMSE (POBI vs SMAP)



(b) Pixel-wise ubRMSE (DCSTI vs SMAP)

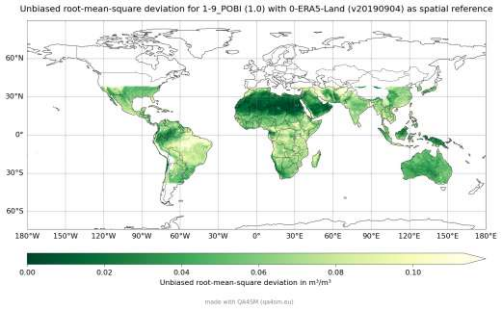


(c) Pixel-wise R (POBI vs SMAP)

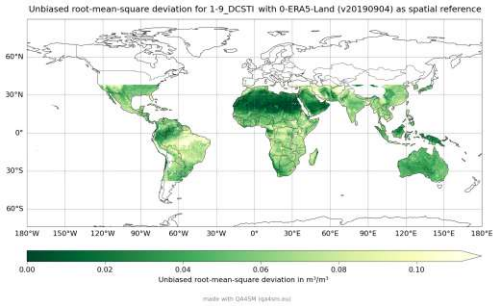


(d) Pixel-wise R (DCSTI vs SMAP)

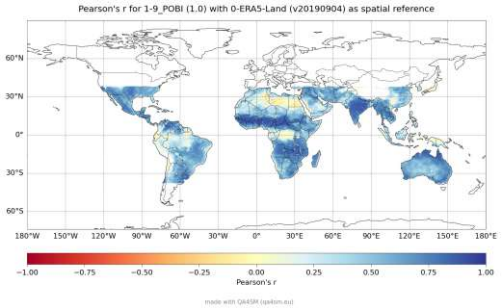
Figure 7.6: Results of the external validation of the POBI-interpolated and DCSTI-interpolated $DC_{3,4}$ datacubes in the 36 km spatial resolution scenario. On the first row, we display the pixel-wise unbiased RMSE (ubRMSE). On the second row, we display the pixel-wise Pearson correlation coefficient (R). For this resolution, we have validated the IFAC-TW datacubes against data from SMAP



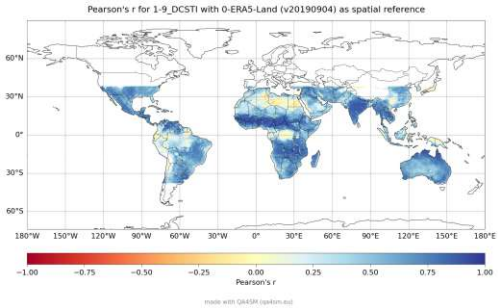
(a) Pixel-wise ubRMSE (POBI vs ERA5-Land)



(b) Pixel-wise ubRMSE (DCSTI vs ERA5-Land)

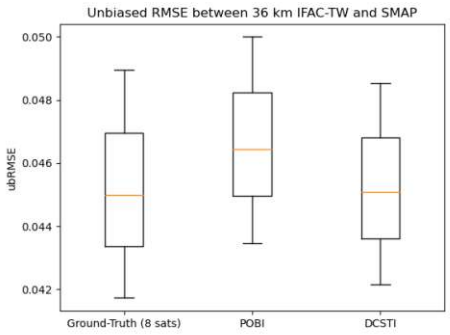


(c) Pixel-wise R (POBI vs ERA5-Land)

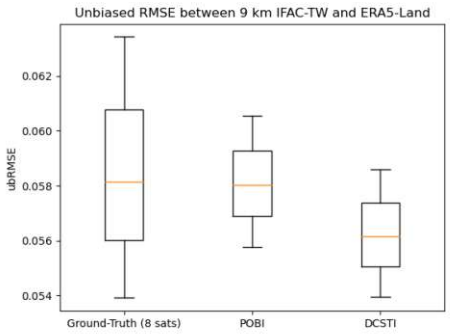


(d) Pixel-wise R (DCSTI vs ERA5-Land)

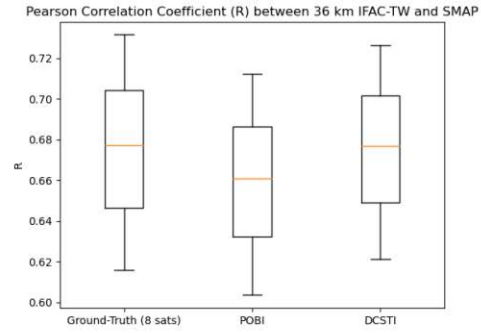
Figure 7.7: Results of the external validation of the POBI-interpolated and DCSTI-interpolated $DC_{3,4}$ datacubes in the 9 km spatial resolution scenario. On the first row, we display the pixel-wise unbiased RMSE (ubRMSE). On the second row, we display the pixel-wise Pearson correlation coefficient (R). For this resolution, we have validated the IFAC-TW datacubes against data from ERA5-Land



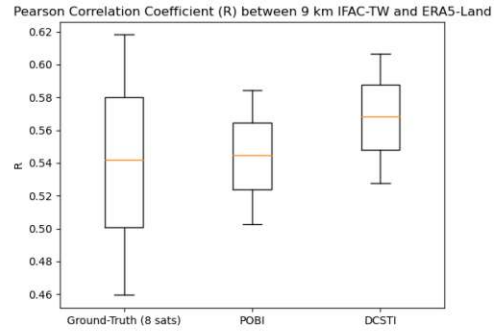
(a) ubRMSE between all three 36 km datacubes and SMAP data



(b) ubRMSE between all three 9 km datacubes and ERA5-Land data



(c) R between all three 36 km datacubes and SMAP data



(d) R between all three 9 km datacubes and ERA5-Land data

Figure 7.8: Results of the external validation of DC_{full}, as well as the POBI-interpolated and DCSTI-interpolated versions of $DC_{3,4}$ in each spatial resolution scenario. In the first row, we display the global unbiased RMSE (ubRMSE) of all three datacubes with respect to external data in each spatial resolution scenario. In the second row, we display the global Pearson correlation coefficient (R) of all three datacubes with respect to external data in each spatial resolution scenario.

CHAPTER 8

Conclusion

In this thesis we propose DCSTI, a novel solution for the problem of spatiotemporal interpolation of GNSS-R-derived soil moisture data. DCSTI employs deep learning to produce a unified regression model that encapsulates the overall spatiotemporal dynamics of soil moisture, while also being flexible enough to deal with location-specific patterns.

DCSTI has shown comparable performance to that of POBI, the current state-of-the-art algorithm for addressing this problem. Notably, it does so while also mitigating one of POBI's main limitations: the requirement of a large model size. DCSTI achieves this by employing a single neural network which is able to compress considerable domain knowledge about spatiotemporal SM patterns within a relatively small number of parameters. This contrasts with POBI's approach, which consists of building an ensemble of location-specific regression models designed to encode the relative SM patterns between neighboring pixels. Although every model within the POBI ensemble seeks to capture the behavior of the same environmental variable, each one of them must be able to do so independently. Thus, the ensemble makes an inefficient usage of computational resources by storing redundant information across multiple regressors.

These conclusions stem from an experiment that evaluated DCSTI in two steps, which we denote as Phase 1 and Phase 2. In Phase 1, both DCSTI and POBI have been trained to perform spatiotemporal interpolation of CYGNSS SSM datacubes derived from the IFAC-TW dataset. After training, both models were used to interpolate the spatiotemporal gaps of a datacube derived from a small subconstellation of CYGNSS. The quality of those two fully-interpolated datacubes was evaluated by comparing them to a third CYGNSS SSM datacube derived from the entire CYGNSS constellation, which, for the purposes of this experiment, is considered to contain ground-truth values. This evaluation was based on the RMSE of the interpolated cells within a temporal slice of the datacube that was reserved as a holdout test set, and thus was not included in the training of either POBI or DCSTI. This process was carried out twice, using both a 36 km and a 9 km spatial grid.

The analysis of the results of Phase 1 shows us that DCSTI was able to reach a lower test data RMSE than POBI by a statistically significant, yet very narrow margin. Given the uncertainties involved in the hyperparameter tuning of both algorithms, we refrain from making definitive statements about DCSTI outperforming POBI with respect to the RMSE, and instead conclude that both methods are capable of achieving comparable error levels. It is worth noting, however, that DCSTI is able to achieve such a performance while requiring significantly fewer parameters - specifically, one order of magnitude fewer parameters than POBI on the 36 km scenario, and two orders of magnitude fewer parameters on the 9 km scenario.

In Phase 2, fully-interpolated datacubes produced by the models trained in the previous phase were validated against data from external sources. The ground-truth datacubes used in Phase 1 were also included in this validation procedure, in order to provide a baseline for the other two. In both the 36 km and 9 km scenarios, we have found no statistically significant difference between the errors observed in the three datacubes with respect to the external data. We have assessed this through two metrics: the unbiased RMSE and the Pearson correlation coefficient.

The findings of this thesis open up several avenues for further research. From the results of Phase 2, we have evidence that both POBI and DCSTI have a good potential to improve the spatial coverage and temporal sampling rate of GNSS-R-derived SM datacubes, while also maintaining acceptable data quality standards. Nevertheless, the extent to which these gap-filling algorithms are actually able to enhance the informational content of the datacubes still remains an open question. Investigating this issue would require the development of a theoretical framework that offers a clear definition of information gain within the context of spatiotemporal interpolation of Earth observation data, along with a methodology on how to properly quantify it.

Additionally, there are also prospects for data fusion research. This includes, for instance, investigating extensions of the DCSTI framework that incorporate spatiotemporal data from different satellites into the model's input vectors. These extended versions of DCSTI could potentially be even more effective in addressing the spatiotemporal interpolation problem. Furthermore, the framework could also be adapted to tasks like short-term forecasting or enhancing the spatial resolution of datasets devoid of spatiotemporal gaps.

Looking ahead, once the HydroGNSS constellation becomes operational and starts delivering SM data, there will be a compelling opportunity to explore the feasibility of utilizing transfer learning to address the spatiotemporal interpolation of SM datacubes. This would involve fine-tuning a DCSTI model, initially trained on CYGNSS data, to address the problem domain presented by HydroGNSS.

Ultimately, we anticipate that the findings and perspectives presented in this thesis will be valuable in the development and refinement of spatiotemporal interpolation techniques, further empowering the Earth observation community's ability to extract value from emerging soil moisture data sources.

List of Figures

List of Tables

Bibliography

[BA12] Jessica Blunden and Derek S. Arndt. State of the climate in 2011. *Bulletin of the American Meteorological Society*, 93(7):S1 – S282, 2012.

[BBH+12] Mary J. Brodzik, Brendan Billingsley, Terry Haran, Bruce Raup, and Matthew H. Savoie. Ease-grid 2.0: Incremental but significant improvements for earth-gridded data sets. *ISPRS International Journal of Geo-Information*, 1(1):32–45, 2012.

[BCM+14] Luca Brocca, Luca Ciabatta, Christian Massari, Tommaso Moramarco, Sebastian Hahn, Stefan Hasenauer, Richard Kidd, Wouter Dorigo, Wolfgang Wagner, and Vincenzo Levizzani. Soil as a natural rain gauge: Estimating global rainfall from satellite soil moisture data. *Journal of Geophysical Research: Atmospheres*, 119(9):5128–5141, 2014.

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[BK11] M. J. Brodzik and K. Knowles. Ease-grid 2.0 land-ocean-coastline-ice masks derived from boston university modis/terra land cover data, version 1, 2011.

[BMPH+18] Bernhard Bauer-Marschallinger, Christoph Paulik, Simon Hochstöger, Thomas Mistelbauer, Sara Modanesi, Luca Ciabatta, Christian Massari, Luca Brocca, and Wolfgang Wagner. Soil moisture from fusion of scatterometer and sar: Closing the scale gap with temporal filtering. *Remote Sensing*, 10(7), 2018.

[BSJ+19] Ebrahim Babaeian, Morteza Sadeghi, Scott B. Jones, Carsten Montzka, Harry Vereecken, and Markus Tuller. Ground, proximal, and satellite remote sensing of soil moisture. *Reviews of Geophysics*, 57(2):530–616, 2019.

[Che23] Clara Chew. Spatial interpolation based on previously-observed behavior: a framework for interpolating spaceborne gnss-r data from cygnss. *Journal of Spatial Science*, 68(1):155–168, 2023.

[CPP+16] Adriano Camps, Hyuk Park, Miriam Pablos, Giuseppe Foti, Christine P. Gommenginger, Pang-Wei Liu, and Jasmeet Judge. Sensitivity of gnss-r spaceborne observations to soil moisture and vegetation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(10):4730–4742, 2016.

[CS18] C. C. Chew and E. E. Small. Soil moisture sensing using spaceborne gnss reflections: Comparison of cygnss reflectivity to smap soil moisture. *Geophysical Research Letters*, 45(9):4049–4057, 2018.

[CS20] Clara Chew and Eric Small. Description of the ucar/cu soil moisture product. *Remote Sensing*, 12(10), 2020.

[CSZ+16] Clara Chew, Rashmi Shah, Cinzia Zuffada, George Hajj, Dallas Masters, and Anthony J. Mannucci. Demonstrating soil moisture remote sensing with observations from the uk techdemosat-1 satellite mission. *Geophysical Research Letters*, 43(7):3317–3324, 2016.

[dD16] Richard de Jeu and Wouter Dorigo. On the importance of satellite observed soil moisture. *International Journal of Applied Earth Observation and Geoinformation*, 45:107–109, 2016. Advances in the Validation and Application of Remotely Sensed Soil Moisture - Part 1.

[DYS+19] Junyu Dong, Ruiying Yin, Xin Sun, Qiong Li, Yuting Yang, and Xukun Qin. Inpainting of remote sensing sst images with deep convolutional generative adversarial network. *IEEE Geoscience and Remote Sensing Letters*, 16(2):173–177, 2019.

[EKBG19] Orhan Eroglu, Mehmet Kurum, Dylan Boyd, and Ali Cafer Gurbuz. High spatio-temporal resolution cygnss soil moisture estimates using artificial neural networks. *Remote Sensing*, 11(19), 2019.

[ENO+10] Dara Entekhabi, Eni Njoku, Peggy O'Neill, K.H. Kellogg, Wade Crow, W.N. Edelstein, Jared Entin, Shawn Goodman, Thomas Jackson, Joel Johnson, J. Kimball, Jeffrey Piepmeier, Randal Koster, Neil Martin, Kyle Mcdonald, Mahta Moghaddam, Susan Moran, Rolf Reichle, Jiancheng Shi, and Jakob van Zyl. The soil moisture active and passive (smap) mission. *Proceedings of the IEEE*, 98:704 – 716, 06 2010.

[Fuk80] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr 1980.

[FVS+20] Conrad James Foley, Sagar Vaze, Mohamed El Amine Seddiq, Alexey Unagaev, and Natalia Efremova. Smartcast: Predicting soil moisture interpolations into the future using earth observation data in a deep learning framework, 2020.

[GDA+20] A. Gruber, G. De Lannoy, C. Albergel, A. Al-Yaari, L. Brocca, J.-C. Calvet, A. Colliander, M. Cosh, W. Crow, W. Dorigo, C. Draper, M. Hirschi, Y. Kerr, A. Konings, W. Lahoz, K. McColl, C. Montzka, J. Muñoz-Sabater, J. Peng, R. Reichle, P. Richaume, C. Rüdiger, T. Scanlon, R. van der Schalie, J.-P. Wigneron, and W. Wagner. Validation practices for satellite soil moisture retrievals: What are (the) errors? *Remote Sensing of Environment*, 244:111806, 2020.

[HLVDMW17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[HUD+85] Martti Hallikainen, Fawwaz Ulaby, Myron Dobson, Mohamed El-Rayes, and Lil-Kun Wu. Microwave dielectric behavior of wet soil-part 1: Empirical models and experimental observations. *Geoscience and Remote Sensing, IEEE Transactions on*, GE-23:25 – 34, 02 1985.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[ISM+23] Hanifeh Imanian, Hamidreza Shirkhani, Abdolmajid Mohammadian, Juan Hiedra Cobo, and Pierre Payeur. Spatial interpolation of soil temperature and water content in the land-water interface using artificial intelligence. *Water*, 15(3), 2023.

[JASC23] Bhargavi Janga, Gokul Prathin Asamani, Ziheng Sun, and Nicoleta Cristea. A review of practical ai for remote sensing in earth sciences. *Remote Sensing*, 15(16), 2023.

[JWD22] Shuanggen Jin, Qisheng Wang, and Gino Dardanelli. A review on multi-gnss for earth observation and emerging applications. *Remote Sensing*, 14(16), 2022.

[KEPO22] Charlie Kirkwood, Theo Economou, Nicolas Pugeault, and Henry Odbert. Bayesian deep learning for spatial interpolation in the presence of auxiliary information. *Mathematical Geosciences*, 54(3):507–531, Apr 2022.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[KSZQ20] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial intelligence review*, 53:5455–5516, 2020.

[KWW+10] Yann H. Kerr, Philippe Waldteufel, Jean-Pierre Wigneron, Steven Delwart, François Cabot, Jacqueline Boutin, Maria-José Escorihuela, Jordi Font, Nicolas Reul, Claire Gruhier, Silvia Enache Juglea, Mark R. Drinkwater, Achim Hahne, Manuel Martín-Neira, and Susanne Mecklenburg. The smos mission: New tool for monitoring key elements ofthe global water cycle. *Proceedings of the IEEE*, 98(5):666–687, 2010.

[LBD+89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, 12 1989.

[Lin76] Seppo Linnainmaa. Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics*, 16(2):146–160, Jun 1976.

[LML+11] David R. Legates, Rezaul Mahmood, Delphis F. Levia, Tracy L. DeLiberty, Steven M. Quiring, Chris Houser, and Frederick E. Nelson. Soil moisture: A central and unifying theme in physical geography. *Progress in Physical Geography: Earth and Environment*, 35(1):65–86, 2011.

[LSGD23] Preet Lal, Ankit Shekhar, Mana Gharun, and Narendra N. Das. Spatiotemporal evolution of global long-term patterns of soil moisture. *Science of The Total Environment*, 867:161470, 2023.

[MN93] Manuel Martin-Neira. A passive reflectometry and interferometry system (paris): Application to ocean altimetry. *ESA Journal*, 17:331–355, 01 1993.

[MP43] Warren Mcculloch and Walter Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[MSDAP+21] Joaquín Muñoz Sabater, Emanuel Dutra, Anna Agusti-Panareda, Clement Albergel, Gabriele Arduini, Gianpaolo Balsamo, Souhail Boussetta, Margarita Choulga, Shaun Harrigan, Hans Hersbach, Brecht Martens, Diego Miralles, Maria Piles, Nemesio Rodriguez-Fernandez, Ervin Zsótér, Carlo Buontempo, and J.-N Thépaut. Era5-land: A state-of-the-art global reanalysis dataset for land applications. *Earth System Science Data*, 13:4349–4383, 09 2021.

[MZKE00] D. Masters, V. Zavorotny, S. Katzberg, and W. Emery. Gps signal scattering from land for moisture content determination. In *IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium. Taking the Pulse of the Planet: The Role of Remote Sensing in Managing*

the Environment. Proceedings (Cat. No.00CH37120), volume 7, pages 3090–3092 vol.7, 2000.

[NZS+17] Son V. Nghiem, Cinzia Zuffada, Rashmi Shah, Clara Chew, Stephen T. Lowe, Anthony J. Mannucci, Estel Cardellach, G. Robert Brakenridge, Gary Geller, and Ake Rosenqvist. Wetland monitoring with global navigation satellite system reflectometry. *Earth and Space Science*, 4(1):16–39, 2017.

[OC23a] S. Chan E. G. Njoku T. Jackson R. Bindlish O'Neill, P. E. and J. Chaubell. Smap l2 radiometer half-orbit 36 km ease-grid soil moisture, version 9, 2023.

[OC23b] S. Chan E. G. Njoku T. Jackson R. Bindlish O'Neill, P. E. and J. Chaubell. Smap l3 radiometer global daily 36 km ease-grid soil moisture, version 9, 2023.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[RCL+18] Christopher S. Ruf, Clara Chew, Timothy Lang, Mary G. Morris, Kyle Nave, Aaron Ridley, and Rajeswari Balasubramaniam. A new paradigm in earth environmental monitoring with the cygnss small satellite constellation. *Scientific Reports*, 8(1):8782, Jun 2018.

[RCLC+22] Christopher Ruf, Hugo Carreno-Luengo, Clara Chew, Mahta Moghaddam, Derek Posselt, Juan Crespo, and Zhaoxia Pu. The nasa cyclone global navigation satellite system smallsat constellation. 08 2022.

[RCS+21] T. Maximillian Roberts, Ian Colwell, Rashmi Shah, Stephen Lowe, and Clara Chew. Gnss-r soil moisture retrieval with a deep learning approach. In *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pages 147–150, 2021.

[RDS+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec 2015.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, Oct 1986.

[RUD+13] C. Ruf, M. Unwin, J. Dickinson, R. Rose, D. Rose, M. Vincent, and A. Lyons. Cygnss: Enabling the future of hurricane prediction [remote sensing satellites]. *IEEE Geoscience and Remote Sensing Magazine*, 1(2):52–67, 2013.

[SCC+22] Emanuele Santi, M.P. Clarizia, Davide Comite, L. Dente, Leila Guerriero, Nazzareno Pierdicca, and Nicolas Floury. Combining cygnss and machine learning for soil moisture and forest biomass retrieval in view of the esa scout hydrognss mission. pages 7433–7436, 07 2022.

[SCD+10] Sonia I. Seneviratne, Thierry Corti, Edouard L. Davin, Martin Hirschi, Eric B. Jaeger, Irene Lehner, Boris Orlowsky, and Adriaan J. Teuling. Investigating soil moisture–climate interactions in a changing climate: A review. *Earth-Science Reviews*, 99(3):125–161, 2010.

[SGS15] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks, 2015.

[SPP+20] E. Santi, S. Pettinato, S. Paloscia, M.P. Clarizia, L. Dente, L. Guerriero, D. Comite, and N. Pierdicca. Soil moisture and forest biomass retrieval on a global scale by using cygnss data and artificial neural networks. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 5905–5908, 2020.

[UPC+21] Martin J. Unwin, Nazzareno Pierdicca, Estel Cardellach, Kimmo Rautiainen, Giuseppe Foti, Paul Blunt, Leila Guerriero, Emanuele Santi, and Michel Tossaint. An introduction to the hydrognss gnss reflectometry remote sensing mission. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:6987–6999, 2021.

[VJF+21] J-M. Vient, F. Jourdin, R. Fablet, B. Mengual, L. Lafosse, and C. Delacourt. Data-driven spatio-temporal interpolation of sea surface sediment concentration from satellite-derived data: An osse case-study in the bay of biscay. In *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pages 2576–2579, 2021.

[VKT+21] Sumit Kumar Varshney, Jeetu Kumar, Aditya Tiwari, Rishabh Singh, Venkata M. V. Gunturi, and Narayanan C. Krishnan. Deep geospatial interpolation networks. *CoRR*, abs/2108.06670, 2021.

[VPDSBV85] G. Vachaud, A. Passerat De Silans, P. Balabanis, and M. Vauclin. Temporal stability of spatially measured soil water probability density function. *Soil Science Society of America Journal*, 49(4):822–828, 1985.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach,

R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[VVH+12] Karl Vanderlinden, Harry Vereecken, Horst Hardelauf, Michael Herbst, Gonzalo Martinez, Michael Cosh, and Yakov Pachepsky. Temporal stability of soil water contents: A review of data and analyses. *Vadose Zone Journal*, 11, 11 2012.

[WHK+13] W Wagner, S Hahn, R Kidd, T Melzer, Z Bartalis, S Hasenauer, J.de Rosnay Figa-Saldaña, A Jann, S Schneider, J Komma, G Kubu, K Brugger, C Aubrecht, J Züger, U Gangkofner, Stefan Kienberger, L Brcca, Y Wang, G Blöschl, J Eitzinger, K Steinnocher, P Zeil, and F Rubel. The ascat soil moisture product: A review of its specifications, validation results, and emerging applications. *METEOROLOGISCHE ZEITSCHRIFT*, page 5–33, 2013. 22 (1).

[WLR99] Wolfgang Wagner, Guido Lemoine, and Helmut Rott. A method for estimating soil moisture from ers scatterometer and soil data. *Remote Sensing of Environment*, 70(2):191–207, 1999.

[WPD+08] Wolfgang Wagner, Carsten Pathe, Marcela Doubkova, Daniel Sabel, Annett Bartsch, Stefan Hasenauer, Günter Blöschl, Klaus Scipal, José Martínez-Fernández, and Alexander Löw. Temporal stability of soil moisture and radar backscatter observed by the advanced synthetic aperture radar (asar). *Sensors*, 8(2):1174–1197, 2008.