Universität Hamburg

MIN Faculty

Department of Earth Sciences

Master's thesis

Creating a long data record of
Ice Water Path

Author:

Lorena Kowalczyk

Supervisors:

Prof. Dr. Stefan Bühler

Dr. Ákos Horváth

Matriculation number: 7254893

Degree program: Meteorologie (M.Sc.)

Contact details: l.kow@t-online.de

Date of submission: 21/08/2023

Climate, Radiation, Remote Sensing

Meteorological Institute

Topic: Creating a long data record of Ice Water Path

Creating a long data record of Ice Water Path

Abstract

Almost a decade ago, SPARE-ICE, the first ice water path (IWP) product that combined infrared (IR) and microwave (MW) radiances from active and passive sensors, was presented. However, one of the main advantages that SPARE-ICE provides, its potential to generate long term global IWP data records with a high spatio-temporal sampling, has not been exploited yet.

This study presents two new temporally extended global SPARE-ICE IWP data records. As a basis, the re-implementation of SPARE-ICE by Mrziglod (2018) is used. SPARE-ICE relies on a decision tree classifier for ice cloud detection and on a multilayer perceptron for IWP regression. As inputs, MW measurements from MHS and IR measurements from AVHRR/3 are used. Those are supplemented by position information, land-sea information, and surface elevation information.
The first new SPARE-ICE IWP data record relies on NOAA-18 measurements and covers the period between January 2007 an December 2018. The second one relies on MetOp-A measurements and covers the period between May 2007 and December 2019. Based on annually averaged zonal means and global distribution patterns, both IWP data records achieve reasonable results. A detailed validation still needs to be done. The new data records help to meet the need for more accurate global IWP data records and make SPARE-ICE attractive for other studies that focus on diurnal, seasonal, intra-seasonal, and intra-annual temporal and geographical variability of IWP.

To generate the new IWP data records, SPARE-ICE was re-trained on NOAA-18 data from 2007 to 2010 with IWP data from 2C-ICE as reference. The re-trained ice cloud detector achieves a F1 score of 0.889 on the testing dataset and has a lower sensitivity limit regarding IWP of 10 g/m^2. The re-trained IWP regressor achieves a R^2 score of 0.839 on the testing dataset.

This study also adds a parallelised pre-processing toolkit to SPARE-ICE that deals with the application of input data quality indicators and the removal of multiple occurrences of input data points to ensure high quality input data. The pre-processing toolkit can be used as a stand-alone toolkit.

Shell scripts, jupyter notebooks, detailed implementation descriptions, as well as workflow illustration are added to SPARE-ICE to make the product more user-friendly and the underlying algorithm more accessible.

Contents

Acronyms

2C-ICE	CloudSat and CALIPSO Ice Cloud Property Product
AAPP	ATOVS and AVHRR Pre-processing Package
AMSU-B	Advanced Very High Resolution Radiometer
ANN	Artificial Neural Network
ARTS	Atmospheric Radiative Transfer Simulator
ATOVS	Advanced TIROS Operational Vertical Sounder
AVHRR	Advanced Bery High Resolution Radiometer
CALIOP	Cloud-Aerosol Lidar with Orthogonal Polarization
CALIPSO	Cloud-Aerosol Lidar and Infrared Pathfinder Satellite Observation
CLASS	Comprehensive Large Array-data Stewardship System
CPR	Cloud Profiling Radar
CPU	Central Processing Unit
DARDAR	raDAR/liDAR project
HDF	Hierarchical Data Format
DKRZ	German Climate Computing Centre
DO-OP	Daylight Only Operations
DTC	Decision Tree Classifier
ECT	Equatorial Crossing Time
EOL	End Of Life
EPS-SG	EUMETSAT Plar System-Second Generation
EUMETSAT	European Organisation for the Exploitation of Meteorological Satellites
GAC	Global Area Coverage data
GB	Gigabyte
GMT	Generic Mapping Tools data preprocessing and display package
HIRS	High-resolution Infra Red Sounder

IASI-NG	Infrared Atmospheric Sounder Interferometer – New Generation
ICON	ICOsahedral Non-hydrostatic model
IPCC	Intergovernmental Panel on Climate Change
IR	Infrared
ISCCP	International Satellite Cloud Climatology Project
ITCZ	Intertropical Convergence Zone
IWC	Ice Water Content
IWP	Ice Water Path
LiDAR	Light Detection And Ranging
MB	Megabyte
MHS	Microwave Humidity Sounder
MLP	Multilayer Perceptron
MODIS	Moderate Resolution Imaging Spectroradiometer
MODPs	Multiple Occurrences of Data Points
MSPPS	Microwave Surface and Precipitation Products System
MW	Microwave
NOAA	National Oceanic and Atmospheric Administration
PATMOX-X	Pathfinder Atmospheres Extended product
RADAR	Radio Detection and Ranging
RFC	Random Forest Classifier
SLURM	Simple Linux Utility for Resource Management
SPARE-ICE	Synergistic Passive Atmospheric Retrieval Experiment-ICE
TIR	Thermal infrared
TIROS	Television InfraRed Observation Satellite
UINT	Unsigned Integers
UTC	Coordinated Universal Time
VIS	Visible

List of Figures

List of Tables

Chapter 1

Introduction

Upper tropospheric ice clouds have a strong effect on the radiation budget of the Earth-atmosphere system. The net effect is based on two opposing effects. On the one hand, their greenhouse effect warms the Earth-atmosphere system. On the other hand, they cool the Earth-atmosphere system by reflecting incoming solar shortwave radiation. The net effect on the radiation budget depends strongly on the horizontal and vertical extent of the cloud, cloud top temperature and ice water content (IWC), as well as on micro-physical properties which all influence the optical thickness of the cloud. (Ramanathan et al., 1989; T. Chen, Rossow & Zhang, 2000; Duncan & Eriksson, 2018; Buehler et al., 2007; Eliasson, Buehler, Milz, Eriksson & John, 2011; Hartmann, Ockert-Bell & Michelsen, 1992; Hang, L'Ecuyer, Henderson, Matus & Wang, 2019; L'Ecuyer, Hang, Matus & Wang, 2019).
Another important aspect is that through the release of latent heat during the depositional growth of ice particles and the release of heat during sublimation, ice clouds also affect the atmospheric energy budget (Buehler et al., 2007; Pakarinen, Pulido Lamas, Roudsari, Reischl & Vehkamäki, 2021). In addition, ice clouds also affect the hydrological cycle as the distribution of ice clouds strongly affects precipitation (Eliasson et al., 2011; Buehler et al., 2007; Montmessin, Forget, Rannou, Cabane & Haberle, 2004).
Several studies have already highlighted the relevance of ice clouds due to the factors mentioned above (L'Ecuyer et al., 2019; Hang et al., 2019; Hartmann et al., 1992; Ramanathan et al., 1989; Buehler et al., 2007; Waliser et al., 2009, e.g.).

Ice clouds represent a substantial fraction of ice in the atmosphere (Dou, Xiao, Huang, Yue & Han, 2020; Hirabayashi, Kanae, Motoya, Masuda & Döll, 2008). The term 'ice' includes all frozen hydrometeors in the atmosphere (Waliser et al., 2009). Components of atmospheric ice are e.g. ice clouds, snow and graupel. To quantify the amount of ice in the atmosphere, the variable ice water path (IWP) can be

used. IWP represents the vertical column integral of IWC and is usually expressed in g/m^2 (Eliasson et al., 2011). While IWP generally includes all ice particles, the components used to calculate IWP tend to differ between studies (Waliser et al., 2009; Eliasson et al., 2011).

Despite the high relevance of ice for the radiation budget, the energy budget, and the water cycle, large differences in IWP exist between individual models. Waliser et al. (2009) showed that there is a factor of 20 difference between the largest and smallest globally averaged IWP values between models that were used in the fourth IPCC assessment report. After removing the two largest outliers, a factor of about 6 between the largest and smallest globally averaged IWP values remained. Even almost a decade later, there was still no consensus on the IWP value between different models (Duncan & Eriksson, 2018). Finding out which model performs best in terms of retrieving IWP is also difficult, as retrieved IWP values based on in situ measurements by aircraft that could be used for validation are rare and not globally available (Ringel, 2023; Brath et al., 2018; Heymsfield et al., 2008; Gayet et al., 2004; Deng, Mace, Wang & Lawson, 2013). Validation is therefore only possible via satellite measurements.

For measuring atmospheric ice, sensors that operate in the visible (VIS), infrared (IR), or microwave (MW) spectral range are being used (Waliser et al., 2009; Buehler et al., 2007; Eliasson et al., 2011; Duncan & Eriksson, 2018). In general, higher sensitivity to ice clouds is achieved with high frequency sensors (VIS and IR sensors) than with low frequency sensors (MW sensors) due to the dependence of scattering properties on the relation of wavelength and particle size (Eliasson et al., 2011; Buehler et al., 2007).

VIS sensors measure the sunlight reflected by clouds. These kind of measurements only work for daylight conditions (Buehler et al., 2007; Holl et al., 2014). VIS sensors are sensitive to small particles (Cimini et al., 2023). Thin ice clods are almost transparent (Zhao et al., 2023; Ewald et al., 2021). For thick clouds, measurements are limited to the cloud top (Buehler et al., 2007).

In the IR and MW spectral range, ice clouds are detected by a modification of brightness temperatures compared to a clear-sky case (Buehler et al., 2007; Holl et al., 2014; Laviola, Levizzani, Ferraro & Beauchamp, 2020). IR sensors measure thermal emission from clouds, atmosphere and surfaces (Buehler et al., 2007). The sensors are sensitive to very small particles (Buehler et al., 2007). In the IR spectral range, ice clouds can act as nearly perfect black bodies (Duncan & Eriksson, 2018). This makes it possible to even detect thin clouds (Duncan & Eriksson, 2018). Even though IR sensors have a great sensitivity to ice clouds, their signals are subject

to strong attenuation. As a consequence, IR sensors can only obtain information from the top of clouds (Buehler et al., 2007). On the other hand, MW-based sensors can obtain information from thicker clouds and have a sensitivity to the whole atmospheric column, as the interaction of MW signals with cloud particles is not very strong. (Duncan & Eriksson, 2018; Buehler et al., 2007; G. Hong, Heygster, Miao & Kunzi, 2005). However, MW signals have difficulties detecting thin cirrus clouds. MW sensors are sensitive to large ice particles, but insensitive to very small ice particles (Eliasson et al., 2011; Waliser et al., 2009).

Since VIS, IR, and MW sensors only detect a specific part of the cloud column, a combination of low and high frequency sensors is necessary for the best possible coverage of IWP. Synergies between those different types of sensors have already been successfully tested to retrieve ice cloud properties (Romano, Cimini, Rizzi & Cuomo, 2007; Islam & Srivastava, 2015; Holl et al., 2014; Mrziglod, 2018; Mastro et al., 2022; Ewald et al., 2021; Cimini et al., 2023). The results highlight the advantage of using such synergies over using individual sensors to detect ice cloud properties, including IWP (Islam & Srivastava, 2015; Holl et al., 2014; Mrziglod, 2018; Mastro et al., 2022). Nevertheless, cloud products based on synergies of high and low frequency sensors are still rare (Cimini et al., 2023; Holl et al., 2014; Duncan & Eriksson, 2018). IWP products that combine the advantages of high and low frequency sensors would therefore be beneficial.

For such a product, measurements from both active and passive sensors can be used. Passive sensors detect radiation that is emitted by an external target, for example a cloud or the Earth's surface. Examples of passive sensors are the Microwave Humidity Sounder (MHS) (Bonsignori, 2007) , the Advanced Microwave Sounding Unit - B (AMSU-B), the High-resolution Infra Red Sounder (HIRS), and the Advanced Very High Resolution Radiometer (AVHRR) (Cracknell, 1997). In general, passive sensors have a good spatial and temporal coverage. They exist on scientific and operational satellites and data based on passive sensors can be processed far back in time. This makes it possible to generate long time series. However, they provide little information about the vertical structure of the atmosphere and thus can determine IWP less accurate. (Holl et al., 2014; Waliser et al., 2009)

Active sensors, on the other hand, generate and direct energy towards a target and then detect the radiation that is coming back from the target (Carter et al., 2012; Rustamov, Hasanova & Zeynalova, 2018). Examples for active sensors are Light Detection and Ranging (LiDAR) sensors or RAdio Detection And Ranging (RADAR) sensors. In contrast to passive sensors, active sensors can resolve the vertical structure of atmospheric ice and thus are likely to determine IWP more

accurately (Waliser et al., 2009; Holl et al., 2014). However, the error is difficult to determine because in situ measurements, which could be the basis for more accurate IWP calculations, are rare. There are also drawbacks to the use of active sensors. Active sensors are more costly than passive sensors and require a lot of energy. They are also less suitable for examining geographical patterns, as they tend to have s small footprint. In addition, as they are exclusively carried on scientific platforms, their continued operation is uncertain. This can severely restrict data sets based on active sensors in terms of time. (Holl et al., 2014; Waliser et al., 2009). The benefits of both active and passive sensors are therefore complementary.

Several IWP products based on spaceborne passive sensors already exist. For example, PATMOS-X (VIS, IR) (Heidinger, Foster, Walther & Zhao, 2014), ISCCP (VIS, IR) (Rossow et al., 2016), MODIS (VIS, IR) (King, Tsay, Platnick, Wang & Liou, 1997) and MSPPS [1] (MW) provide IWP based on passive sensors.

Since the launch of CloudSat and CALIPSO on April 28, 2006, additional IWP products emerged based on active measurements from the CloudSat Cloud Profiling Radar (CPR) and the CALIPSO Clouod-Aerosol Lidar with Orthogonal Polarization (CALIOP), such as 2C-ICE (CloudSat and CALIPSO Ice Cloud Property Product) (Deng, Mace, Wang & Okamoto, 2010) and DARDAR (raDAR/liDAR project) (Delanoë & Hogan, 2008). According to the availability of the active CloudSat-CALIPSO sensors, these products are only available from 2006 onwards. Furthermore, CloudSat faced serious battery issues in 2011.[2] Since then, only daytime data is available from CloudSat's CPR. The spatio-temporal sampling of CloudSat and CALIPSO is also very limited. Those factors limit the usability of the CloudSat-CALIPSO products, especially for the analysis of IWP on diurnal, intra-seasonal, and intra-annual time scales (Duncan & Eriksson, 2018; Y. Hong & Liu, 2015).

This is unfortunate, because observation-based datasets with high accuracy to atmospheric ice from spaceborne active sensors could help to meet the need for more accurate global IWP data records and provide a valuable constraint and validation target for models if they were long enough (Eliasson et al., 2011; Buehler et al., 2007; Duncan & Eriksson, 2018). Creating such a dataset for day and night does not seem possible at the moment based on purely active spaceborne sensors. In order to obtain a long IWP time series with high accuracy based on spaceborne sensors, it thus stands to reason to combine spaceborne sensors with different frequencies from active and passive sensors. In this way, advantages of the different sensors can be

---

[1] `https://www1.ncdc.noaa.gov/pub/data/satellite/publications/podguides/N-15%20thru%20N-19/pdf/2.6%20Section%209.0%20NESDIS%20Operational%20Products.pdf`, last visited: 30.08.2023, 16:56

[2] `https://cloudsat.atmos.colostate.edu/news/CloudSat_status`, last visited: 13.08.2023, 17:06

combined and disadvantages reduced.

One way to combine the information from different sensors is through machine learning methods. For example, Artificial Neural Networks (ANNs) can receive input from both passive MW and IR measurements and can be trained using a dataset based on active measurements as reference. By taking active measurements as a reference, the ANN tries to imitate the accuracy of the active measurements during its training. By using passive measurements as input, one takes advantage of the good spatial and temporal coverage of passive data.

ANNs have found a wide range of applications in the geosciences (Minnis, Sun-Mack, Smith Jr, Hong & Chen, 2019; Mas & Flores, 2008; Gardner & Dorling, 1998), from determining chemical constituents in rocks (Ninomiya, 1995) to predicting daily total solar radiation (Elizondo, Hoogenboom & McClendon, 1994). They have been widely used to determine cloud properties like cloud optical depth, effective radius, cloud top height, cloud base height, and cloud liquid path (Minnis et al., 2016, 2019; Strandgren, Bugliaro, Sehnke & Schröder, 2017; Kox, Bugliaro & Ostler, 2014; Håkansson, Adok, Thoss, Scheirer & Hörnquist, 2018; Faure, Isaka & Guillemet, 2002; Vasudevan, Gohil & Agarwal, 2004; Brath et al., 2018, e.g.) and have been successfully applied to retrieve IWP ('successful' in this sense means with a low error compared to active measurements). For example, Mastro et al. (2022) applied an ANN on IR data from IASI-NG (Infrared Atmospheric Sounder Interferometer - New Generation) and MW data from the EPS-SG (EUMETSAT Polar System-Second Generation) Microwave Sounder sensor to retrieve cloud liquid and ice water path by using active measurements from CloudSat and CALIPSO as reference. (Wang, Wang, He & Zhang, 2022) applied an ANN on MW measurements from the microwave humidity sounder aboard FenYun-3B to retrieve IWP by using 2C-ICE as reference. In Ringel (2023), ANNs were also applied to retrieve IWP based on aircraft measurements.

The first global IWP product that combined passive IR and MW radiances and retrieved IWP based on spaceborne sensors was the Synergistic Passive Atmospheric Retrieval Experiment-ICE (SPARE-ICE) (Holl et al., 2014). SPARE-ICE relies on machine learning methods. For IWP regression, an ANN is used. Active sensors are used during training of SPARE-ICE. Thus, SPARE-ICE not only makes use of the advantages of active and passive spaceborne sensors, but also exploits the synergies of high and low measurement frequencies in order to retrieve the ice content in the atmosphere as completely as possible. Up to now, only IWP data records that cover periods between June 2006 and August 2010 have been generated with SPARE-ICE (Holl et al., 2014; S. Li, 2015; Mrziglod, 2018).

This study has two objectives:

1. To provide a SPARE-ICE based IWP data record that covers a longer period of time. The new IWP data record should start in the early/mid 2000s and approach the current year (2023) as closely as possible.

   The following is used as a basis:

   – the baseline experiment and SPARE-ICE algorithm from Mrziglod (2018) for the SPARE-ICE setup

   – IWP data from 2C-ICE for validation

   In order to generate a data record that covers a long period of time, certain factors have to be taken into account:

   – the data quality of measurements varies over time

   – the continuous availability of all measurement channels is not always given

   It turned out that in order to achieve the goal of generating a temporally extended IWP data record with SPARE-ICE, the generation of two new global IWP data records with SPARE-ICE was appropriate. The two new data records are based on passive data from MHS and AVHRR measurements. The first new data record is based on NOAA-18, the second new data record is based on MetOp-A.

2. To improve the SPARE-ICE algorithm and make it more user- and developer-friendly. This was especially important because:

   – the SPARE-ICE algorithm from Mrziglod (2018) was not yet sufficiently tested and partly non-executable. This means that improvements had to be made before SPARE-ICE could be applied.

   – the SPARE-ICE implementation was not described in sufficient detail. This made it difficult for developers to access the code.

This study is structured as follows. Chapter 2 briefly introduces SPARE-ICE with its main components and describes the historical development of SPARE-ICE. After a brief introduction to each of the main components, the main changes that I applied to them in the course of this study are explained. More detailed descriptions of the SPARE-ICE main components, including detailed implementation descriptions, can be found in the Appendix and are especially recommended for further developers.

Chapters 3 and Chapter 4 deal with the input data from SPARE-ICE and its pre-processing. Chapter 3 describes the datasets on which SPARE-ICE is based. First, an overview of the satellites available for SPARE-ICE is given and the selection process of NOAA-18 and MetOp-A as data basis for the new SPARE-ICE products is explained. Afterwards, the passive MHS and AVHRR sensors are presented. As active sensors, RADAR and LiDAR are presented. Chapter 4 introduces the pre-processing toolkit I developed and applied to ensure high quality sensor input data.

Chapter 5 deals with the actual application of SPARE-ICE. For this purpose, the setup of SPARE-ICE used for this study is presented first. In order to generate the new IWP data records, SPARE-ICE had to be re-trained. The re-training setup and its results are also addressed in Chapter 5.

Chapter 6 outlines the retrieval process that I used to generate the two new SPARE-ICE IWP data records and presents both data records through an application example that retrieves the diurnal cycle of IWP. Global maps and annual zonal means are presented. Chapter 7 provides a summary of the study and an outlook for future work.

A user guide for the generation of additional datasets using SPARE-ICE is provided in Appendix A.

Chapter 2

SPARE-ICE

This chapter introduces SPARE-ICE with its main components and lists the main modifications in the implementations of the SPARE-ICE components that have been conducted within the scope of this study. More detailed explanations of the components and implementation details were generated for this study and included in Appendix B and Appendix C.

2.1 What is SPARE-ICE?

SPARE-ICE was introduced by Holl et al. (2014) as the first IWP retrieval product that benefited from both active and passive MW and IR spaceborne sensors. SPARE-ICE considers the full atmospheric column, including cloud ice, graupel and snowfall (Holl et al., 2014). As the product is not based on measurements from the VIS frequency range, it provides IWP data during daytime and nighttime.

The combination of MW and IR in SPARE-ICE proved to be useful for deriving plausible IWP from tropical to polar regions. However, one has to keep in mind that reference in situ measurements for IWP that could be used for assessing the accuracy of the retrieval product are still almost non-available. (Deng et al., 2013; Holl et al., 2014; S. Li, 2015)

What makes SPARE-ICE particularly interesting is that SPARE-ICE is in principle applicable to global passive MW and IR measurement data covering the period from 1998 to the present. Thus, SPARE-ICE offers the possibility to generate global IWP over decades. So far, however, only data between June 2006 and August 2010 has been generated with SPARE-ICE (Holl et al., 2014; S. Li, 2015; Mrziglod, 2018).
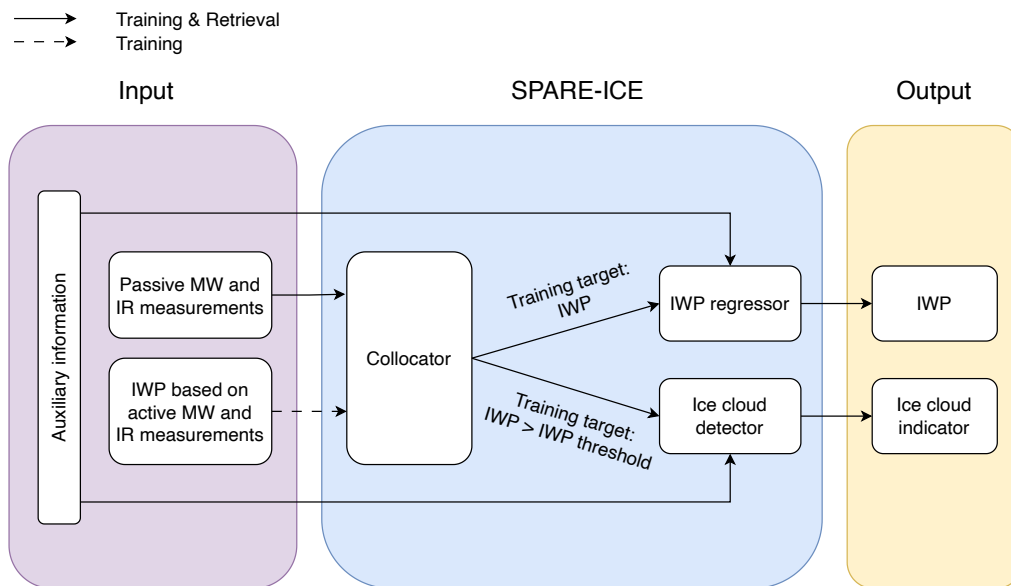
Figure 2.1: Schematic representation of SPARE-ICE including its required inputs and retrieved outputs. SPARE-ICE consists mainly of three components: a collocator, an IWP regressor and an ice cloud detector. The figure was adopted from Figure 2.8 from Mrziglod (2018, p.17) and partly modified.

SPARE-ICE retrieves IWP by using machine learning methods, starting from mainly passive MW and IR measurements from spaceborne sensors. As reference during the training of the machine learning methods, IWP from the 2C-ICE product is used. The 2C-ICE product contains IWP data that is retrieved from RADAR and LiDAR measurements. Thus, SPARE-ICE benefits from the measurement accuracy of active sensors by trying to imitate their accuracy (Mrziglod, 2018).

Even though the setup of SPARE-ICE evolved over time (Table 2.1), SPARE-ICE can still be reduced to three major components: a collocator, an IWP regressor and a cloud detector (Figure 2.1). The collocator is used to combine measurements that observe almost the same place at almost the same time (Mrziglod, 2018).

As illustrated in Figure 2.1, the IWP regressor and the ice cloud detector both undergo a training and a retrieval process. During the training process, IWP based on active MW and IR measurements (reference dataset) is compared to retrieved SPARE-ICE IWP values. During the retrieval process, the final outputs of SPARE-ICE are generated. These are: IWP values and ice cloud indicators (*ice cloud flags*). The ice cloud flags indicate whether a cloud is actually present or not.

For the generation of the original SPARE-ICE product, Holl et al. (2014) used an ANN (type: Multilayer Perceptron) for the regression of IWP and another ANN that for the prediction of ice cloud probability (Table 2.1). By training both ANNs based

SPARE-ICE	Holl et al. (2014)	Mrziglod (2018)	current version
Programming language	MATLAB	Python	Python
Spatial collocation tool	Interval subdivision	Ball tree	Ball tree
IWP regressor	Multilayer Perceptron	Multilayer Perceptron	Multilayer Perceptron
Ice cloud detector	Multilayer Perceptron	Decision Tree Classifier	Decision Tree Classifier

Table 2.1: Main SPARE-ICE characteristics for different SPARE-ICE versions.

on collocated MW and IR radiances from active and passive sensors and applying them to IR and MW collocated radiances from passive sensors only, Holl et al. (2014) were able to retrieve IWP on a global scale with an accuracy that approaches the one from active sensors (Holl et al., 2014).

The original SPARE-ICE retrieval performed well in general, had a consistently good performance for different cloud types, and showed great potential, especially in regions were IWP retrievals are difficult (for example near the poles) (Holl et al., 2014; Mrziglod, 2018; S. Li, 2015). However, some essential improvements to SPARE-ICE were still pending: the original toolkit was slow on large datasets and based on experimental code (Mrziglod, 2018). In addition, Li (2015) noticed that, at least for the year 2008, SPARE-ICE strongly underestimated IWP at mid-latitudes when compared to 2C-ICE, its reference dataset used for training.

Mrziglod (2018) addressed these known problems and set the next step for SPARE-ICE by translating SPARE-ICE from MATLAB into Python, implementing more efficient workflows, and by finding a way to reduce its mid-latitudinal bias: SPARE-ICE was redesigned by using object-oriented programming principles and restructured to allow parallel processing support. The previous spatial collocation search tool of SPARE-ICE was replaced by a more efficient ball tree method and the distance metric simplified. Instead of using an ANN, Mrziglod (2018) decided on a much faster to train decision tree classifier for ice cloud detection. For reducing the mid-latitude bias, SPARE-ICE was re-trained with more data and expanded input information.

This study builds on SPARE-ICE from Mrziglod (2018). The main components and their setup on SPARE-ICE have not been changed. However, I made additional improvements to the SPARE-ICE code from Mrziglod (2018) and connected SPARE-ICE with a new pre-processing toolkit.

2.2   Adjustments of the SPARE-ICE components

The SPARE-ICE package from (Mrziglod, 2018) was already complete, fast, and well structured. However, SPARE-ICE still contained bugs in its implementation, was partly inconsistent between similar methods and and could not yet handle some anomalies of specific files (e.g. empty files, missing collocations). Also not all example python scripts (.py files) were executable and properly structured. This made the use of SPARE-ICE unnecessarily difficult and unhandy.

In the course of this study, all errors found in the implementation were corrected. The classes that use the scikit-learn package were adjusted, so that the code is now executable with the more recent scikit-learn version 1.1.1 (see Section 5.1). Where necessary, I redesigned the example python scripts to be more user-friendly and better executable. In addition, I created shell (.sh) scripts to simplify the execution of SPARE-ICE and to automate tasks, and created jupyter notebooks to make it easier to examine the underlying processes step-by-step.

Detailed descriptions of the basic SPARE-ICE components, as well as a description of their configuration as used in this study, can be found in Appendix B. The appendix also contains detailed implementation descriptions and simplified (Appendix B.1, B.2 and B.3) as well as detailed (Appendix C.1, C.2, C.3) work flow illustrations that I created in order to make the SPARE-ICE underlying algorithm more accessible to future users and developers.

The following subsections are limited to a brief introduction to the respective SPARE-ICE components and to the main modifications in their implementations that I conducted within the scope of this study. Since this study passes data from MHS, AVHRR, and 2C-ICE to SPARE-ICE, this section uses data from MHS, AVHRR, and 2C-ICE as input examples.

2.2.1   Collocator

The purpose of the collocator component of SPARE-ICE is to find collocations between measurements from different sensors. In the context of SPARE-ICE, *collocations* are events, where two or more spaceborne sensors measure almost the same place at almost the same time (Mrziglod, 2018). The process of finding collocations is called *collocating* and is performed by a *collocation algorithm*. The collocation component can also be used independently of SPARE-ICE.

Whether two measurements from different sensors form a collocation or not, depends on a given maximum time limit (Δt) and a maximum spatial distance (Δs) which can be set individually. For small thresholds, the collocation search can be more

efficient. However, one then risks not finding any collocations at all or only in specific regions. For example, due to different equator-crossing times of the NOAA-18 and MetOp-A satellites, one finds only collocations between the NOAA-18 and MetOp-A MHS measurements between 70° - 80° North and South between the December 10 and December 20, 2013, if a Δt of 5 min and a Δs of 5 km are set (Mrziglod, 2018). On the other hand, when the limits are set too loosely, it can happen that the collocation algorithm relates two observations to each other which, according to the objectives of the study, differ too much from each other. For example, for atmospheric variables like IWP or for the inter-calibration of measurements, it is important that the measured atmospheric state has not changed significantly between collocated measurements (Mrziglod, 2018).

The following paragraphs briefly describe the role of collocations in SPARE-ICE and list the main changes in the implementation of the collocator component that were introduced to SPARE-ICE within the scope of this study. The methods mentioned can be found in Appendix B and Appendix C, if not stated otherwise.

Collocations in SPARE-ICE

Collocations are important for the SPARE-ICE training and retrieval process. For the retrieval process, collocations between MHS measurements and AVHRR measurements are required to provide all input measurements needed for the retrieval. For the training, collocations between MHS measurements, AVHRR measurements, and IWP retrieval values from 2C-ICE are required. This is to ensure that the model tries to match only MHS and AVHRR measurements with 2C-ICE IWP values that are related to the same scene. As the collocation tool of SPARE-ICE allows only two sensors to be collocated at the same time, the collocation procedure has to be executed twice in a row to generate the raw training dataset. First, the collocation algorithm is performed on MHS and 2C-ICE IWP. Subsequently, the MHS and 2C-ICE collocations (*MHS_2C-ICE*) are further collocated with AVHRR to generate the final collocations (*MHS_2C-ICE_AVHRR*) that are needed for the training of SPARE-ICE. When the collocation algorithm has to be executed twice in a row to obtain all required collocations, the resulting collocations are referred to as *co-collocations* in this study.

Modifications in the collocator component

The collocation algorithm tries to find collocations between two input datasets, *primary* and *secondary*, which were taken from a primary and a secondary file respectively. For example, a MHS file represents a primary file and an AVHRR file represents a secondary file. During the collocation procedure, collocations between

primary and secondary are searched for, taking into account Δt and Δs. First, a temporal pre-selection is applied during the file search for primary and secondary files. Then spatial collocations are searched for, taking into account Δs. Afterwards, the temporal threshold Δt is applied on the spatially pre-selected data points and the resulting collocations are saved to disk.

Some changes were made to the collocator component algorithm by me in SPARE-ICE.

Adjustments of the file search process:

- Inclusion of previously missed files:
  To collect file information (i.a. file paths) of all primary and secondary files which could potentially, according to the date and times as stated in their file name, lead to collocations, the method *find* is used. During its search, *find* only relies on the time period as provided by the user via *start* (start time) and *end* (end time), and on time information that are contained in the directory names and file names. In the version by Mrziglod (2018), the search of *find* was restricted to the semi-open interval [*start*, *end*). Assume, *start* is set to '2008-01-01 13:00' and *end* is set to '2008-01-01 14:00', and two files exist that overlap with the time period given by *start* and *end*. The first file contains the time stamp '2008-01-01 13:00 - 2008-01-01 13:59' and the second file the time stamp '2008-01-01 14:00 - 2008-01-01 14:59' in their file names. Then, the second file would be missed by *find*. This implies that even though the user expects to find all files that contain data points that lie within the time period '2008-01-01 13:00 - 2008-01-01 14:00', *find* only provided them with files that covered data points that lie within the time period '2008-01-01 13:00 - 2008-01-01 13:59'. Hence, data points that could lead to collocations could be missed. In the adjusted version of SPARE-ICE, *find* searches on the closed interval [*start*, *end*]. The use of a closed interval in *find* is now consistent with the use of a closed interval in the method *_get_common_time_period*, which selects a time period based on the time stamps from the data points from the primary and secondary files, *start*, *end*, and Δt.

- Optimising the usage of parallel resources:
  The search interval given by *start* and *end* is inserted into *find* via the generator *match*. In the previous version, *match* extended the search interval for primary files by Δt. This was unnecessary, because the collocation algorithm only needs to check whether data points from the secondary files are centred around a point from a primary file under consideration of Δt. In the worst case the extended search interval for primary files led to two additional search

results from *find* which initiated two additional child processes in one of the parallelised parts of the collocation algorithm. The additional child processes did not return any collocations. By restricting the search interval, computing resources were saved.

- Improved handling of empty files:
  In the original version by Mrziglod (2018), the collocation search would break when no files were found during the file search. Even though a parameter (*no_files_error*) existed in *find* that was supposed to handle this exception, the parameter was not transferred correctly in the collocation algorithm. This caused the program to stop every time no file was found. To solve this issue, I created the parameter *skip_file_errors* and incorporated it into *match*, so that *match* could pass on the file handling parameter accordingly. The parameter *no_files_error* was also added to the method *retrieve_from_collocations*[1] which is used to retrieve SPARE-ICE IWP from collocations between MHS and AVHRR files. For implementation reasons, the program continues when no files were found when *skip_file_errors* is set to true or when the parameter *no_files_error* in *find* is set to false.

Adjustments of the collocation process:

- Handling of empty collocations:
  If files were found in the search directories, but no potential collocations resulted from those files, the collocation algorithm stopped executing and made an exit. This problem was caught by forwarding no value (*None*) instead.

- Prevention of artificial duplicates:
  Major changes were implemented between the reading and spatial collocation of the primary and secondary files. In the original version by Mrziglod (2018), the threads for reading the files were initiated in the background. Every time a secondary was available, the primary was collocated with this single secondary. This made it possible to start collocating while other secondaries were still being read in the background. What sounds like an efficient approach, however, had the disadvantage that a point from primary, which could be collocated to points from two different secondaries according to the collocation criteria, created two separate collocations instead of just one. When the coordinates from primary are chosen as representatives for the collocated datasets (meaning that points from secondary are *collapsed* (Section 5.2) onto the points from

---

[1]`https :// radiativetransfer .org / misc / typhon / doc -trunk / _modules / typhon / retrieval/spareice/common .html#SPAREICE .retrieve _from _collocations`, last visited: 12.07.2023, 13:07
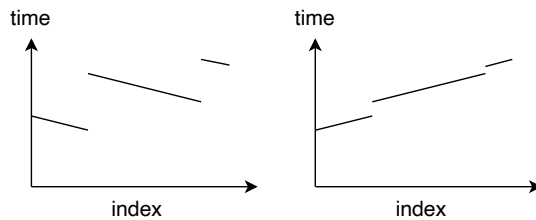
Figure 2.2: Exaggerated sketch of the time stamps of a concatenated file. Before the adjustment (left), the file shows individual sections, each descending in time. The time stamps of the file are non-monotonous over the file index. After the bugfix (right), the time stamps are monotonically increasing.

primary), this lead to artificial duplicates according to the latitude and longitude coordinates and time stamps of the primary. Those artificial duplicates over-represented data points from the primary and contained incomplete collocations with the data points from the secondaries. This made the artificial duplicates unusable for further use. By concatenating the secondaries first before collocating them with primary, the generation of those artificial duplicates is now prevented. However, the disadvantage of the current approach is that all secondaries have to be read in first, before the spatial collocation search can start. This slightly increases the runtime.

- Fixing monotonicity problems:
  In the version by Mrziglod (2018), the collocation algorithm could not handle unsorted input data. This was especially a problem, when the primary input dataset was non-monotonous due to concatenated sections that were each descending in time. As a result of the missing monotonicity, some sections within the input datasets were neglected by the collocation algorithm and fewer co-collocations than possible were found. This bug didn't raise an error and was therefore well hidden from the user. In addition, the resulting collocations were returned unsorted. This means that if the unsorted collocations were further used to create co-collocations, the error occurred again.

  To fix this issue, I sorted the primary and secondary datasets first before inserting them into the spatial and temporal collocation methods. The resulting collocations are now returned in an ascending order sorted by time. An example of an input dataset before and after the correction is illustrated in Figure 2.2.

2.2.2   IWP regressor

For the regression of IWP, SPARE-ICE uses a Multilayer Perceptron (*MLP*). A MLP is a fully connected feed-forward artificial neural network (da Silva, Hernane Spatti, Andrade Flauzino, Liboni & dos Reis Alves, 2017). A description of the MLP prepared for this study can be found in Appendix B.2

IWP regression in SPARE-ICE

Before the MLP can be used to retrieve IWP, it has to be trained and tested. This happens during the training process of SPARE-ICE. Due to the high dynamic range of IWP (10^{-2} to 10^4 g/m^2), the training of the MLP takes place in log space.

For the training of the MLP, a dataset consisting of input fields (*features*) and outputs (*targets*) is required. The targets are the expected, or rather the desired, outputs of the MLP that one wants to get for the input data when the input data is inserted into the MLP. This only makes sense when data points from the input data and data points from the target data can be attributed to the same scene. To ensure this, the collocator is used to create collocations for the training of the MLP.

The collocation datasets created for training are referenced as MHS_2C-ICE_AVHRR and contain collocations between passive measurements from MHS and AVHRR, and retrieved IWP data from 2C-ICE that is based on active measurements. The IWP data from 2C-ICE is used as target data for the MLP during training. Information from MHS and AVHRR are used as input features.

The collocations created for training are then split into three datasets: a training dataset, a validation dataset, and a testing dataset.

Based on the training dataset, the MLP adjusts its parameters in such a way that its output matches the IWP target as best as possible. This tunes the MLP according to its task. To prevent over-fitting to the target data of the training dataset, the performance of the MLP is assessed not only on the training dataset, but also on the validation dataset during the training phase. A trade-off of the errors on the performance on both datasets then fixes the final parameter choice. The testing dataset is used to calculate the final score of the MLP.

After training, the MLP is configured and ready for retrieving IWP based on collocated passive data from MHS and AVHRR without the need of a target.

In order to provide the user with a better understanding of the MLP and its training and retrieval phase, I have compiled a summary about the MLP and created a documentation of its training and retrieval process in Appendix B.2.

Modifications in the IWP regressor component

No major adjustments have been made to the IWP regressor component algorithm in SPARE-ICE within the scope of this study. The component, as well as the setup for the training of the MLP have been adopted from Mrziglod (2018) without further changes. The setup of the MLP is summarised in Appendix B.2.

Mrziglod (2018) and Holl et al. (2014) conducted several experiments with SPARE-ICE in which they varied the selection of the input features. The feature selection for the MLP as used for this study was adopted from the baseline experiment from Mrziglod (2018), with the exception of one feature which was excluded due to a lack of continual availability (refer to Chapter 5 and Section 3.2.2 for more details). By orientating to the baseline experiment, an attempt was made to achieve a similar accuracy as in Mrziglod (2018).

If one re-trains the MLP, one should consider that even the same MLP setup with the same input data points and target data points will not necessarily lead to the same final MLP setup, because the MLP is re-optimised each time it is trained and contains random aspects during its training. This implies that changes in the final MLP configuration do not indicate changes in the SPARE-ICE code and even with the same training setup, the same accuracy cannot be guaranteed.

2.2.3 Ice cloud detector

For the ice cloud detection, SPARE-ICE uses a Decision Tree Classifier (DTC). DTCs are maximum likelihood classifiers that belong to the tree-based supervised learning techniques (Hauska & Swain, 1975; Wilkinson, 2004). For this study, I created a description of the architecture of a DTC in Appendix B.3.

Ice cloud detection in SPARE-ICE

As the IWP regression is performed in log space, IWP values of 0 g/m^2 must be excluded from the training of the IWP regressor. This implies that to check for cases of IWP = 0 g/m^2, an additional method is needed. This is the purpose of the ice cloud detector. The ice cloud detector checks whether the case of IWP = 0 g/m^2 (*clear sky*) or IWP > 0 g/m^2 (*cloudy*) exists. The classification into cloudy or clearsky scenes is performed by the DTC.

Similar to MLPs, DTCs also have a training and a retrieval phase. During the training phase, the tree is constructed and evaluated on a training dataset. In SPARE-ICE, this training dataset is represented by MHS_2C-ICE_AVHRR collocations. Constructing the tree mainly means formulating questions against which

attributes from the input data are tested. As inputs, information from MHS and AVHRR from the training dataset are used. As reference (*target*) for the training, two classes based on the IWP data provided by 2C-ICE were created: cloudy and clearsky. IWP values that were larger than $0 \ g/m^2$ were classified as cloudy, IWP values that contained $0 \ g/m^2$ were classified as clearsky. During the construction phase, the questions for the attributes are modified so that the tree can predict the classes from the target as well as possible.

After the training, the tree is configured and can be queried with passive data from MHS and AVHRR to classify the input data into cloudy and clearsky scenes without information from 2C-ICE.

The implementation of the DTC in SPARE-ICE is closely connected to the implementation of the MLP. For further developers I created workflow diagrams of the training and retrieval phase of the DTC and MLP in Appendix C.2 and C.3 and added a simplified documentation of the implementation details of both phases in Appendix B.2 and in Appendix B.3.

Modifications in the ice cloud detector component

No major adjustments have been made to the ice cloud detector component algorithm in SPARE-ICE for this study. The ice detector component was adopted from the SPARE-ICE package from Mrziglod (2018).

Uncertainties still remained in the exact setup of the DTC: according to the setup description in Mrziglod (2018), a Random Forest Classifier (RFC) was used to predict IWP. A RFC trains several DTCs and returns a final classification prediction by combining all predictions from all trained DTCs. This can help to improve the predictive accuracy and to prevent over-fitting (Appendix B.3). Mrziglod (2018) stated that he trained a RFC with 20 trees. No configuration details were specified for those trees. However, in the parameter file forwarded by Mrziglod (2018) that describes the final setup of his best experiment (baseline experiment), only a single DTC was mentioned. The default settings of the handed over SPARE-ICE algorithm also contained a single DTC.

Since the default specifications and the parameter file specified a single DTC and the parameter file was the only indication of the baseline experiment with complete training information and setup information, I decided to use a single DTC for the default settings. For setting up the DTC, I used the setup information from the baseline experiment parameter file.

As with the MLP, each re-training of the DTC can produce different results.

Chapter 3

Data basis

In order to create a long IWP data record with SPARE-ICE, a data basis for SPARE-ICE must be selected. For this study, MHS and AVHRR/3 of NOAA-18 and MetOp-A are selected as passive sensors. IWP data from the 2C-ICE product is used as IWP reference. 2C-ICE is based on active LiDAR and RADAR measurements from CloudSat and CALIPSO. The sensor measurements were supplemented by auxiliary information, such as land-sea information and elevation information.

In the following, the selection of satellites and sensors for this study is justified and retrieval periods are selected. Subsequently the passive MHS and AVHRR/3 sensors are presented. This is followed by a presentation of the 2C-ICE product with a focus on the underlying active RADAR and LiDAR sensors. Afterwards, the auxiliary datasets required by SPARE-ICE according to the setup used in this study are then presented.

3.1 Data record frameworks

To to generate an IWP data record using SPARE-ICE that starts in the early/mid 2000s and ends as close as possible to the current year, 2023, it must first be decided which sensors are to serve as a basis for SPARE-ICE and, based on the sensor choice, which satellites can serve as a source. To provide continuity within the IWP retrieval data records, sensor and satellite changes should be avoided as far as possible throughout the whole retrieval period. Peculiarities in the orbits of the various satellites, e.g. changes in orbits or differences between different satellite orbits, have to be considered.

In the following, the sensor and satellite choices made for this study will explained and some peculiarities of the satellites considered for the selection will be pointed out. Afterwards the retrieval time periods used in this study are selected.

Sensor and satellite choices

According to Holl et al. (2014), SPARE-ICE can in principal rely on MHS, as well as on AMSU-B data for its MW channel inputs. For the IR measurements, only AVHRR and HIRS have been mentioned as sources so far. However, as HIRS footprints are highly noncontinuous, Holl et al. (2014) has considered the use of HIRS as less suitable. Therefore, HIRS was not considered further for the instrument selection and the focus was placed on AVHRR data. The first and second generation of AVHRR only cover the period until May 2007.[1] Thus, only AVHRR/3 is interesting for the time period of interest. As such, only satellites with MHS, AMSU-B, and AVHRR/3 are considered in the following.

Figure 3.1 lists the satellites on which AMSU-B, MHS, and AVHRR/3 have been installed. Figure 3.2 shows the mission period of each satellite, based on the launch date and official end of life date (EOL). A list with the exact dates has been compiled in the Appendix (Table D.5).

Out of MHS and AMSU-B, AMSU-B was discarded for the retrieval, because AMSU-B does not provide data far enough into the current time period (cf. Figure 3.1 and Figure 3.2): NOAA-16 and NOAA-17 officially stopped operating on June 09, 2014, and April 10, 2013 respectively. NOAA-15 is still operational, however, after channel failures and scan motor problems, AMSU-B operations on NOAA-15 stopped in 2011.[2] Thus, the choice between MHS and AMSU-B therefore fell on MHS.
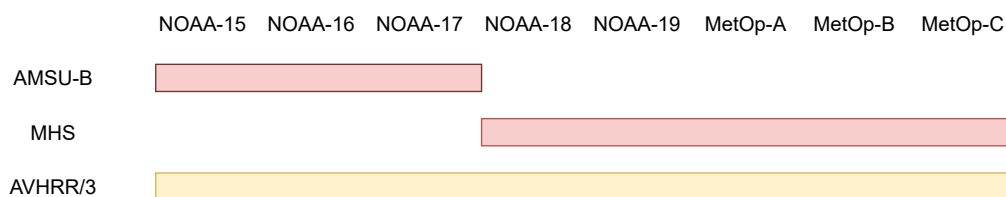
Figure 3.1: Overview of the satellites on which the AMSU-B, MHS, and AVHRR/3 sensors have been installed respectively.

---

[1] http://eodg.atm.ox.ac.uk/eodg/instrument_AVHRR, last visited: 17.07.2023, 22:30 and Wunderle and Neuhaus (2020)

[2] https://space.oscar.wmo.int/instruments/view/amsu_b, last visited: 24.07.2023, 15:23, and https://www.ospo.noaa.gov/Operations/POES/NOAA15/amsu_b.html, last visited: 24.07.2023, 15:24

[3] https://space.oscar.wmo.int/satellites, last visited: 30.07.2023, 17:49
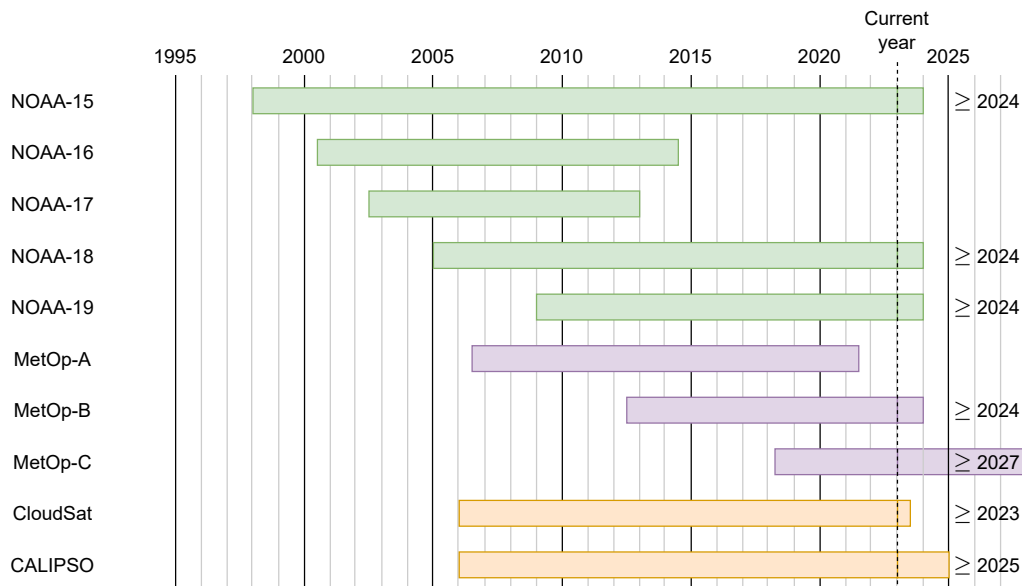
Figure 3.2: Mission phases of the satellites using MHS or AMSU-B and AVHRR/3, and of CloudSat and CALIPSO, which provide the basic data for 2C-ICE. The phases start on the launch date and end on the EOL date. The EOL data for satellites that were still operational in July 2023 are based on estimates as of July 2023. The data was compiled from the World Meteorological Organisation Observing Systems Capability Analysis and Review Tool's website[3].

Apart from covering the period of interest very well, MHS also has the advantages that it has a better sensitivity compared to AMSU-B and has already been used by Holl et al. (2014), Li (2015), and Mrziglod (2018) in connection with SPARE-ICE. This simplifies future comparisons of the SPARE-ICE retrieval products between this and other studies. By choosing MHS, the satellite selection was restricted to NOAA-18, NOAA-19, MetOp-A, MetOp-B, and MetOp-C.

Holl et al. (2014), Li (2015), as well as Mrziglod (2018) used data from NOAA-18 for their SPARE-ICE retrievals, and, if carried out, also for the training of SPARE-ICE. This has the advantage that the results from those studies can be compared with each other. Although a comparison of SPARE-ICE with other studies was not carried out in this work, the choice of NOAA-18 would facilitate future comparisons. In addition, NOAA-18 has proven to be advantageous for the analysis, as data from NOAA-18 has the most collocations with the SPARE-ICE reference dataset 2C-ICE for the retrieval periods used in those studies, which ranges from June 2006 to August 2010 (see Appendix D.4 and Eliasson et al. (2011)).

The reason for this is that NOAA-18 had a similar orbit during this time period as CALIPSO and CloudSat, which serve as the data basis for 2C-ICE. This is demonstrated by the relatively similar local Equatorial Crossing Times (ECT) of

the three satellites between 2006 and 2010 (Figure 3.3). During this time period, CALIPSO and CloudSat were still part of the A-train, a satellite constellation in sun-synchronous orbit.[4] The A-train satellites fly in close proximity to each other and cross the equator at about 1.45 pm local time (ascending node). As a representative for CALIPSO, CloudSat is shown in Figure 3.3.
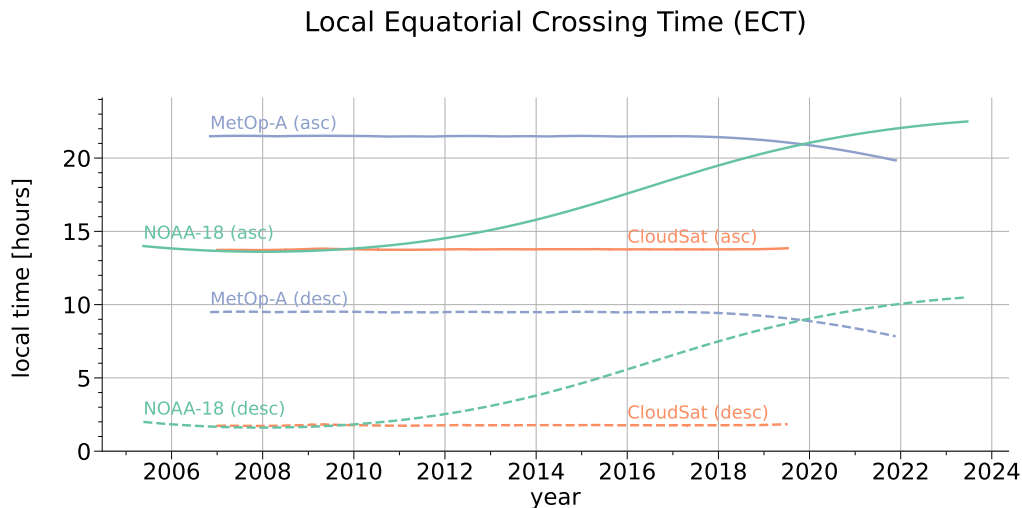
Figure 3.3: Equatorial crossing times (ECTs) for the ascending and descending nodes of NOAA-18, MetOp-A, and CloudSat. Dashed lines represent the local ECT during the descending node, solid lines represent the local ECT during the ascending node. All satellites displayed are in sun-synchronous orbits. For CloudSat, ECTs were calculated based on the 2C-ICE P1_R05 files. The data on which the ECTs for NOAA-18 and MetOp-A are based were calculated with data from the NOAA Product Polar Navigation page[5].

A good amount of collocations provides a good starting point for a possible re-training of SPARE-ICE, but also sufficient references for a comparison between SPARE-ICE retrieval products and 2C-ICE IWP. It was therefore decided to use NOAA-18 for this study as well. Thus, it was decided to generate a SPARE-ICE IWP data record based on passive date from NOAA-18.

However, the use of NOAA-18 raises two problems in particular: on the one hand, data from NOAA-18 can be used for SPARE-ICE only up to October 2018. Since October 21, 2018, MHS is not longer operational on NOAA-18 due to a scan motor

---

[4]CloudSat left the A-Train on February 22, 2018. CALIPSO joined CloudSats's new orbit. Together, CALIPSO and CloudSat form the C-train. The C-Train is located about 16.5 km below the A-Train. The relocation of CloudSat had very little effect on the ECT of CloudSat's orbit. The ground track of the A-train and C-train satellites intersect about every 20 days. (Braun, Sweetser, Graham & Bartsch, 2019)

[5]https://www.ospo.noaa.gov/Products/ppp/navpage.html, last visited: 30.07.2023, 13:40

failure[6]. This prevents an extension of SPARE-ICE up to recent years. On the other hand, NOAA-18 experienced significant shifts in its orbit over the years (cf. Figure 3.3). Thus, varying retrieval results due to the diurnal cycle are expected. In contrast to that, MetOp-A, MetOp-B, and MetOp-C are better stabilised.

In order to provide IWP retrieval data for which no or only marginal effects of orbit variations on the retrieval data have to be expected, and in order to test a possible extension of the retrieval period based on data from MetOp satellites, an additional SPARE-ICE IWP data record based on MetOp-A was generated during this study.

MetOp-A was chosen because the mission phases of NOAA-18 and MetOp-A have the largest overlap. MetOp-A was launched on October 19, 2006. Its operational phase started on May 14, 2007. MetOp-A was decommissioned in November 2021. The ECT of MetOp-A for the ascending node is roughly 09:30 pm over the whole period of interest.

As all MetOp satellites are equipped with MHS and AVHRR/3 sensors, a comparison with data from NOAA-18 is facilitated. Due to the continuity of the sensors between NOAA-18 and MetOp-A, it is expected that differences between the SPARE-ICE IWP data records that originate from both satellites are mainly due to differences in sensor performances and orbital characteristics.

Selection of the retrieval periods

As a starting point for the retrieval period, the year 2007 was selected for both the MetOp-A and the NOAA-18 based IWP data records. 2007 is the first fully covered year, in which data from 2C-ICE is available. This facilitates the comparison with 2C-ICE IWP. As MetOp-A is only operational from May 2007 onwards, the start of MetOp-A based IWP data record is May 2007.

The end of the NOAA-18 IWP data record was set to October 2018. No more MHS data from NOAA-18 is available after October 2018. The upper limit of the retrieval period is therefore reached.

The end of the MetOp-A IWP data record was set for December 2019. On the one hand, no more 2C-ICE data are available after 2019, and on the other hand, this limit is based on current storage capacity limits. An overview of the retrieval periods can be found in Table 3.1.

---

[6]`https://database.eohandbook.com/database/instrumentsummary.aspx?instrumentID=304`, last visited: 30.07.2023, 18:03, and `http://tropic.ssec.wisc.edu/real-time/amsu/amsu_change_log.html`, last visited: 04.08.2023, 15:15

Satellite basis	Retrieval period		
NOAA-18	January 2007	-	October 2018
MetOp-A	May 2007	-	December 2019

Table 3.1: Time span of the SPARE-ICE IWP data records generated during this study.

If desired, it is possible to generate IWP retrieval data based on NOAA-18 back to May 2005. For periods further back in time, NOAA satellites with AMSU-B have to be used. An extension of retrieval IWP data records until November 2021 based on MetOp-A is possible in principle. For an extension of the retrieval product to time periods after November 2021, sensors from MetOp-B and MetOp-C satellites are recommended.

3.2   Passive sensors

SPARE-ICE relies on data from passive sensors during its training and retrieval process. Resulting SPARE-ICE IWP data records are solely based on passive sensors.

Passive sensors measure the natural emissions coming from the Earth's surface and its atmospheric components without transmitting a signal themselves. A common example for a passive remote sensing instrument are radiometers. Passive sensors have long been established in satellite-based remote sensing and thus provide the possibility of processing their data over a long period of time (Holl et al., 2014; Waliser et al., 2009). This enables SPARE-ICE to generate IWP data over several decades.

For the generation of the new SPARE-ICE IWP data records during this study, data from two radiometers, MHS and AVHRR/3, are used. An overview of both sensors is given in the following. Since MHS and AMSU-B are very similar and both can in principle be used by SPAR-ICE (Holl et al., 2014), AMUS-B will be described together with MHS.

3.2.1   MHS / AMSU-B

MHS is an across-track scanning, self-calibrating radiometer, operating in five frequency channels in the millimetre wave band from 89 GHz to 190 GHz (*NOAA KLM User's guide*, 2014; Bonsignori, 2007). MHS was first launched into space with NOAA-18 on May 20, 2005. Launches with NOAA-19, MetOp-A, MetOp-B, and MetOp-C followed.

MHS samples the Earth by means of a rotating reflector at 90 equidistant positions (*scan positions*) with a scanning rate of 3 scans every 8 seconds (*NOAA KLM User's guide*, 2014). One rotation is also referred to as *scan line*. At a nominal orbital altitude of 870 km, a measurement (*pixel*) at nadir covers an area that corresponds roughly to a circle with a diameter of 17 km. At the edge of the scan, a scan position covers approximately 52 km x 27 km (Robel & Graumann, 2009).

At each scan line, radiometric calibration is provided by reference measurements against an on-board blackbody target and a view on free space (*NOAA KLM User's guide*, 2014; Bonsignori, 2007).

MHS is the successor of AMSU-B, which was installed on NOAA-15, NOAA-16, and NOAA-17. NOAA-15 was launched on May 13, 1998.[7] As MHS, AMSU-B covers an area that corresponds roughly to a circle with a diameter of 17 km at nadir (*NOAA KLM User's guide*, 2014). Compared to AMSU-B, MHS constitutes a significant improvement in terms of radiometric sensitivity and calibration accuracy (Bonsignori, 2007). Another difference between MHS and AMSU-B is their different frequencies on channel 2 and channel 5. MHS provides measurements at 157 GHz and 190 GHz, while AMSU-B provides measurements at 150 GHz and 183 ± 7 GHz. The channels used by MHS are listed in Table 3.2.

Channel	Spectral region	Central Frequency (GHz)	Bandwidth (GHz)
1	MW	89.0	2.8
2	MW	157.0	2.8
3	MW	183.311 ± 1.0	2 x 0.5
4	MW	183.311 ± 3.0	2 x 1.0
5	MW	190.311	2.2

Table 3.2: MHS channels with their central frequencies and bandwidths. All channels operate in the microwave regions. Channel 3-5 are centred around the water vapour absorption line at 183 GHz. (*NOAA KLM User's guide*, 2014)

The frequencies of MHS are only sensitive to large ice particles and can penetrate even thick ice clouds (Mrziglod, 2018; Waliser et al., 2009). Thin clouds are (almost) completely transparent (Buehler et al., 2007).

Channel 1 and channel 2 are surface sensitive channels under all conditions (Holl et al., 2014). The remaining channels are around the water vapour absorption line at 183 GHz (Bonsignori, 2007). At those frequencies, large ice particles can be detected due to the scattering of radiation emitted by water vapour below the ice

---

[7] https://space.oscar.wmo.int/satellites/view/noaa_15, last visited: 27.07.2023, 14:59

particles (Holl et al., 2014). If there is enough water vapour in the atmosphere, those channels are not surface sensitive. If the atmosphere is very dry, in particular channel 5 becomes surface-sensitive (Holl et al., 2014).

In this study, MHS measurements from NOAA-18 and MetOp-A are used. The MHS data was downloaded as level 1b data from the Comprehensive Large Array-data Stewardship System (CLASS)[8] from NOAA. After downloading, the level 1b data was converted to level 1c data in HDF format by our working group using the AAPP software version AAPP 8.10 from the ATOVS and AVHRR Pre-processing Package (AAPP)[9]. The resulting HDF files were then saved. The HDF files contain i.a. brightness temperatures for each channel, quality indicators, and geolocation data information. Each HDF file contains the sensing start and end time in UTC of its enclosed measurements in its file name.

3.2.2 AVHRR

AVHRR is a radiometer which was first launched in 1978 with TIROS-N. The latest generation of AVHRR, AVHRR/3, was first launched on NOAA-15 in 1998 and has since been installed on NOAA-16 to NOAA-19, and MetOp-A , MetOp-B, and MetOp-C. Via MetOp-C, AVHRR/3 is expected to provide measurements until at least 2025 (Wunderle & Neuhaus, 2020). Thus, apart from some data gaps due to e.g instrument or transmission problems, AVHRR/3 has been providing data for more than two decades. This makes AVHRR/3 a very valuable data source for long-term studies of the Earth's atmosphere.

Apart from proving long data records and twice-daily global coverage, AVHRR/3 turned out to be well suited for SPARE-ICE as AVHRR/3 is installed on all satellites that also carry MHS. AVHRR/3 footprints overlap with MHS footprints on those satellites. It is therefore to be expected that numerous collocations exist between both sensors. Those collocations can be used for the long-term retrieval of IWP via SPARE-ICE.

AVHRR/3, is comprised of six channels out of which five channels can operate simultaneously. The channels are located in the visible, near-infrared (NIR), and thermal infrared (TIR) spectral regions (Wunderle & Neuhaus, 2020). Table 3.3 lists all channels of AVHRR/3 with their respective spectral bandpass. Channel 1 and channel 2 measure reflected solar radiation (*NOAA KLM User's guide*, 2014). Channel 3A is located in the short wave infrared region (SWIR) (Wunderle & Neuhaus, 2020) and measures reflected solar radiation, while channel 3B is located in the TIR region

---

[8]`https://www.avl.class.noaa.gov/`, last visited: 10.07.2023, 17:04
[9]`https://nwp-saf.eumetsat.int/site/software/aapp/`, last visited: 10.07.2023, 17:05

(*NOAA KLM User's guide*, 2014) and measures a mix of reflected solar and emitted terrestrial radiation (Holl et al., 2014). Channel 4 and channel 5 are located in the TIR region and measure terrestrial infrared radiation.

Channel 3A and 3B can not operate simultaneously and are switched for day/night operations according to operational requirements (*NOAA KLM User's guide*, 2014). Channel 3A was intended for operation during the daylight part of the orbits, while channel 3B was intended for operation during the night time part of the orbit ("AVHRR Level 1b Product Guide", 2011). In the AVHRR datasets, channel 3A and channel 3B are grouped together as channel 3. This means that channel 3 corresponds to either channel 3A or channel 3B. Indicators to distinguish between channel 3A and channel 3B for channel 3 are available in the AVHRR/3 datasets.

Channel	Spectral region	Spectral bandpass (μm)
1	VIS	0.58-0.68
2	NIR	0.725-1.00
3A	NIR	1.58-1.64
3B	TIR	3.55-3.93
4	TIR	10.3-11.3
5	TIR	11.5-12.5

Table 3.3: AVHRR/3 channels with their spectral bandpasses. The channels operate in the visible (VIS), near-infrared (NIR), and terrestrial infrared (TIR) spectral regions. (*NOAA KLM User's guide*, 2014)

In the course of this study, AVHRR/3 data from NOAA-18 and MetOp-A was processed. According to the channel switch indicators in the AVHRR files, channel switches between channel 3A and channel 3B on NOAA-18 occurred between April 2016 and May 2019. Between January 2007 and March 2016, only channel 3B was active. On MetOp-A, AVHRR channel 3A and channel 3B switched throughout the period from January 2007 to December 2019 between 2007 and 2019.

The AVHRR/3 data from NOAA-18 and MetOp-A was downloaded as global area coverage (GAC) data (level 1b) from the Comprehensive Large Array-data Stewardship System (CLASS)[10] from NOAA. The GAC data contain a reduced resolution of the original AVHRR measurements. For the generation of the GAC data, four out of every five samples along the scan line are averaged and only every third scan line is processed ("AVHRR Level 1b Product Guide", 2011). This results in an effective discontinuous footprint of 1.1 x 4.4 km^2 and 409 scan positions (pixels) per scan line ("AVHRR Level 1b Product Guide", 2011). The scan lines are orientated

---

[10]`https://www.avl.class.noaa.gov/`, last visited: 10.07.2023, 17:04

perpendicular to the respective spacecraft orbit track (*NOAA KLM User's guide*, 2014).

After downloading, the GAC files were converted to level 1c data in HDF format by our working group using the AAPP software version AAPP 8.10.[11] As for MHS, the HDF files contain i.a. brightness temperatures for each channel, quality indicators, and geolocation data information and have information about their sensing start and end time in UTC in their file name included.

In the further course of this study, the term AVHRR is used to describe AVHRR/3 measurements from NOAA-18 and MetOp-A.

3.3   Active sensors

SPARE-ICE relies on data from active sensors during its training process. During training, SPARE-ICE requires reference values of IWP. These reference values are taken from 2C-ICE. 2C-ICE is based on CRP from CloudSat and CALIOP from CALIPSO. As CPR and CALIOP are active sensors, SPARE-ICE benefits from active sensors through 2C-ICE.

Active sensors generate energy towards a target and detect the radiation that comes back from the target. The first active sensors that were specifically designed for examining clouds and precipitation were on CloudSat and CALIPSO (G. L. Stephens et al., 2002). CloudSat and CALIPSO were launched together on April 28, 2006 and joined the A-train constellation (G. Stephens et al., 2018). Since the measurements from CloudSat and CALPISO were supposed to complement each other (Winker, Pelon & McCormick, 2003), both satellites mostly flew in close proximity to each other. CloudSat-CALIPSO records have been available since June 2006.

In the following, 2C-ICE and CPR and CALIOP are presented.

2C-ICE

2C-ICE is a CloudSat and CALIPSO based product that uses an optimal estimation framework to determine cloud properties (Mace & Deng, 2019). The 2C-ICE product combines measured RADAR reflectivities from CPR with measured attenuated backscatter coefficients from CALIOP. The combination of CPR and CALIOP measurements allows for a more complete cover of the atmospheric ice column. The 2C-ICE product used in this study is 2C-ICE_P1_R05. Compared to the previous 2C-ICE release, 2C-ICE_P1_R04, the RADAR reflectivity factor parameterization in the LiDAR only regions is improved.

---

[11]`https://nwp-saf.eumetsat.int/site/software/aapp/`, last visited: 10.07.2023, 17:05

The 2C-ICE data (level 2c) was downloaded from the CloudSat Data Processing Center webpage[12] and has not been further modified. The 2C-ICE data is provided in HDF-EOS format[13], based on HDF4. The files contain i.a. IWP data, one quality indicator and geolocation data information. The file name of each 2C-ICE file includes the time in UTC of its first data point[14].

Apart from being used during the training and retrieval process of SPARE-ICE, this study also uses 2C-ICE IWP for accuracy estimation of the new SPARE-ICE IWP data records. Comparisons between the SPARE-ICE records and 2C-ICE IWP are only possible for parts of the SPARE-ICE retrieval period, as CloudSat has been operating in Daylight Only Operations (DO-OP) mode since November 2011 (see CPR description). In DO-OP mode, most of the satellite subsystems are switched off during the 38-minute night portion of CloudSat's orbit. (Braun et al., 2019)

In order to make a comparison between 2C-ICE IWP and the SPARE-ICE IWP record nevertheless, some of the results are subdivided into ascending and descending nodes. This makes a comparison with 2C-ICE possible even after 2011, at least during daytime. At the time of this study, the 2C-ICE product ends in July 2019.

CPR (CloudSat)

CPR is a nadir-viewing radar operating at 94 GHz that measures back-scattered power as a function of distance (G. L. Stephens et al., 2002; L. Li, Durden & Tanelli, 2017). The power measured is converted into radar reflectivity (G. L. Stephens et al., 2002).

The launch of CloudSat on April 28, 2006 made CPR the first millimeter wavelength cloud radar in space (G. L. Stephens et al., 2002). Due to the inclination of CloudSat's orbit, CPR provides only measurements between 82°N and 82°S. Measurements of CPR provide vertical cross-sections of the atmosphere with a vertical resolution of 500 m, a cross-track resolution of 1.4 km and an along-track resolution of 1.7 km. (Waliser et al., 2009)

Although CPR was mainly constructed for the investigation of clouds, it also resolves precipitation systems, like light rain and snow (Ellis, L'Ecuyer, Haynes & Stephens, 2009; G. L. Stephens et al., 2002). Results from Waliser et al. (2009) suggest that CloudSat provides an estimate of the total IWC field, including cloud, snow and

---

[12]`https://www.cloudsat.cira.colostate.edu/`, last visited: 15.08.2023, 16:38

[13]`https://nsidc.org/data/user-resources/help-center/what-hdf-eos`, last visited: 13.08.2023, 21:57

[14]`https://www.cloudsat.cira.colostate.edu/data-products`, last visited: 15.08.2023, 16:45

graupel. Due to its frequency, CPR is only sensitive to large large particles and can penetrate thick clouds. (Waliser et al., 2009; Eliasson et al., 2011). Hence, with a CPR based only product, the mass contribution from small particles would be underestimated (Waliser et al., 2009).

In April 2011, CPR suffered a serious spacecraft battery anomaly. CPR could be put back into operation again. However, due to the battery anomaly, it was decided in November 2011 to operate CloudSat in DO-OP mode only. (Braun et al., 2019)

CALIOP (CALIPSO)

CALIOP is a nadir-viewing LiDAR that measures backscattered intensity at 1064 nm and two orthogonal polarization components at 532 nm from aerosols and cloud tops (Winker et al., 2003). CALIOP is installed on CALIPSO. CALIPSO and Cloud-Sat were launched together on April 28, 2006 and joined the A-train constellation. Due to the inclination of CALIPSO's orbit, CALIOP provides only measurements between 82°N and 82°S (Winker, Hunt & McGill, 2007).

In the upper troposphere, CALIOP has a horizontal resolution of approximately 1 km and a vertical resolution of around 60 m (Winker et al., 2003).

The LiDAR provides profiles that provide information on the vertical distribution of aerosol and clouds and allows for a qualitative classification of aerosol size via the wavelength dependence of the backscatter. Cloud and ice and water phases can be determined by the ratio of signals of the two orthogonal polarization channels. Due to its measurement frequencies, CALIOP is sensitive to small particles and can detect even thin cirrus clouds. For thick clouds, measurements are constrained to the cloud tops. (Eliasson et al., 2011; Winker et al., 2003; Mrziglod, 2018)

3.4   Auxiliary inputs

For the retrieval of IWP and for its training, SPARE-ICE supplements the information from passive sensors with land-sea information and elevation information. In the following the land-sea mask and the elevation file used during this study are introduced.

3.4.1   Land-sea mask

A land-sea mask mask is used to inform the IWP regressor and ice cloud detector whether a measurement was taken over land or over sea. As land-sea mask, I chose the file land_water_mask_5min.png from the ARTS (Atmospheric Radiative Transfer

Simulator) atmlab-data directory[15] from the Meteorological Institute of Universität Hamburg. The file contains a land-sea mask in a 5-arc minute resolution where land is indicated by a value of 0, water is indicated by a value of 255, and pixels close to the coast are indicated by values between 1 and 254.

In the experiments undertaken by Mrziglod (2018), the land-sea mask had the strongest effect on reducing the mid-latitudinal bias and was therefore crucial for the good performance in the mid-latitudes of his baseline experiment.

3.4.2 Elevation

An elevation dataset is passed to SPARE-ICE to improve its performance in regions with higher terrain (Mrziglod, 2018). The elevation information are passed to the IWP regressor as well as to the ice cloud classifier in SPARE-ICE.

For this study, the elevation information were taken from the 1-arc minute resolution dataset ETOPO1_Bed_g_gmt4_1m.grd (Amante & Eakins, 2009) from the National Oceanic and Atmospheric Administration (NOAA) product website[16].

The elevation file contains 21601 by 10801 cells that are centred on lines of latitude and longitude. The elevation data is vertically referenced to sea level and considers the bedrock below ice sheets (Amante & Eakins, 2009). The original file was modified by renaming its vertical variable to 'z' and processed with the the Generic Mapping Tools data processing and display software package (GMT) version 4.5.1 to create the file ETOPO1_Bed_g_gmt4_1m_rn.grd.

Elevation information from ETOPO1 were already provided to SPARE-ICE by Mrziglod (2018). However, it is not clearly stated by Mrziglod (2018), what his source file was and how he modified it. Based on discussions within our working group, it can be assumed that the same file ETOPO1_Bed_g_gmt4_1m.grd from the NOAA product website was also used and modified as for this work.

In the baseline experiment from Mrziglod (2018), the elevation information had the smallest feature importance for the ice cloud classifier.

---

[15]`https://radiativetransfer.org/svn/rt/atmlab-data/trunk/land_water_mask/` 30.09.2022, 14:10

[16]`https://www.ncei.noaa.gov/products/etopo-global-relief-model`, last visited: 15.08.2023, 11:02

Chapter 4

Pre-processing toolkit

When generating a long time series, it must also be taken into account that the selected sensors may deteriorate over time. This can negatively affect the quality of the measurements. Also, as for any other data analysis, special properties of the data (e.g. frequency changes, outliers) have to be considered.

For SPARE-ICE, these data considerations are crucial, as there is no internal data control and no check for physical meaningfulness of the input data integrated in SPARE-ICE, even though SPARE-ICE is heavily built on machine learning methods which rely solely on information provided to them through data.

This means that the well-known mantra 'garbage in - garbage out' from data-intensive research applies, which refers to the acknowledgement that poor-quality input generates unreliable data output (Monique F. & Kerin, 2018).

To ensure high data quality input for SPARE-ICE, I created pre-processing functions and compiled them into a pre-processing toolkit. The functions are applicable on MHS, AVHRR, and 2C-ICE data and deal with the application of quality indicators and the removal of multiple occurrences of data points.

The current SPARE-ICE implementation offers the opportunity of performing the pre-processing and the collocation steps simultaneously by a single function call (see Appendix A). This combines the previous collocation procedures with the newly implemented pre-processing package. Pre-processing and collocating can be each done in parallel. The user can decide whether they want to have the pre-processed data saved as an intermediate result or not. By default, the pre-processed MHS, AVHRR, and 2C-ICE files are only stored temporarily and deleted after they have been used for collocating. This saves a lot of storage space.

In the following, the two types of quality checks offered by the pre-processing toolkit, the removal of multiple occurrences of data points and the application of quality indicators, are explained in more detail and their relevance in relation to SPARE-ICE is highlighted. Implementation details are outlined.

4.1   Removal of multiple occurrences of data points

In this study, multiple occurrences of the data points (*MODPs*) are defined as data points that share exactly the same latitude and longitude coordinate as well as the same time stamp with another data point.

MODPs are not handled separately in SPARE-ICE. Nonetheless, SPARE-ICE is confronted with them through its input datasets. Those MODPs have to be removed, as they introduce a bias into the training data of SPARE-ICE and generate additional IWP data during the retrieval process, which biases e.g. zonal or regional averages. Therefore, the removal of those occurrences is relevant for both short and long time spans.

The part of the pre-processing toolkit that deals with the removal of MODPs adresses the problem of MODPs in the input datasets. Out of the pre-processing toolkit, the functions for the removal of MODPs are the most time-consuming functions.

MODPs in the input data

MODPs exist in the MHS and AVHRR input data from NOAA-18 and MetOp-A. The MODPs are caused by files that overlap in time.

Figure 4.1 shows an extract of MHS files converted to HDF from NOAA-18 for January 01, 2008. The 42-character long names of the files are structured in the following way:

```
<data_set_creation_site>.<data_type>.<spacecraft_id>.
Dyyddd.Shhmm.Ehhmm.<processing_block_id>.<ground_station_source>.h5
```

```
NSS.MHSX.NN.D08001. S0000.E0155 .B1347475.SV.h5
NSS.MHSX.NN.D08001. S0153.E0348 .B1347577.SV.h5
NSS.MHSX.NN.D08001. S0347.E0542 .B1347778.WI.h5
NSS.MHSX.NN.D08001. S0541.E0734 .B1347879.WI.h5
```

Figure 4.1: Extract of the MHS files for January 01, 2008. The files were converted to HDF using the AAPP 8.10 software package. The files have time stamps that overlap in time.

'yy' identifies the year of the century, 'ddd' identifies the day of the year on which the spacecraft began recording the data set. 'S' identifies the group as a start time delimiter and the following 'hhmm' denotes hours and minutes in UTC to the nearest minute. This is the time at which the spacecraft started recording. 'E' identifies the group as and end time delimiter and the following 'hhmm' denotes the hours and minutes in UTC to the nearest minutes at which the spacecraft recorded the last usable data in the data set. (*NOAA KLM User's guide*, 2014)
As can be deduced from the time stamps in Figure 4.1, the files indicate a temporal overlap between one and two minutes.

Random checks of the subfolders showed that the MHS files from NOAA-18 mainly overlap by one, two, and six minutes. The same applies to AVHRR files from NOAA-18. MHS files from MetOp-A also indicate temporal overlaps according to their filenames. In this case, random checks of the subfolders revealed overlaps of one minute. The same applies to AVHRR files from MetOp-A. For the 2C-ICE samples analysed, no temporal overlaps existed between files.

Even though there were cases in the input files in which one file was completely overlapped by another file and also overlapped with a third file, those cases were very rare. Therefore, the functions of the pre-processing package that deal with the removal of MODPs have been called 'functions for the removal of duplicates' in the pre-processing toolkit. Despite their names, these functions are also suitable for removing MODPs.

The functions created for the removal of duplicates always process only a single file each time and remove duplicates within this one file. This implies that in order to remove duplicates between separate files that overlap, the files must be combined first before the functions can be used. For this purpose, the functions were embedded into wrapper functions. The wrapper functions (see Appendix A) are marked with the suffix '_mfilesets' and are also available as parallelised versions. Those wrapper functions primarily require the input of a time period and a dataset path reference and then combine all files that fit within the time period over a given time interval before passing the files into the functions that deal with the removal of duplicates. The removal of the duplicates is then based on the latitude, longitude and time stamps of each data point.

4.2   Consideration of quality indicators

With increasing time of instrument usage, instrument abrasion becomes a more and more relevant factor. This can have an increasingly negative impact on the

quality of e.g. measurements and calibration procedures and thus lead to quality abnormalities. Therefore, special attention should be paid to data quality for longer time series. At the same time, even transmitted data from relatively newly installed instruments can be subject to faults, for example due to calibration problems. It is therefore always recommended to check the quality of one's data.

How much data is affected by quality abnormalities, be it due to calibration problems, GPS problems or the like, depends strongly on the respective file. For example, an analysis I conducted with the default quality flag settings of the AVHRR data from NOAA-18 for 2008 showed that between 0.21 % and 100 % of the data from one file can be affected by potential quality abnormalities. On average, 0.31 % of the NOAA-18 AVHRR data from 2008 were affected by quality abnormalities. The problem of quality abnormalities is therefore not always relevant, but can have a major effect if the file selection is unfortunate.

In the case of AVHRR data from NOAA-18, there are also unphysical brightness temperature values of 0 K and around -323 K included in the level 1c HDF files between January 2008 and December 2010. If one wants to achieve that SPARE-ICE establishes connections between its inputs and its reference data, which are supposed to replace physical regularities purely based on the input and reference data during its training, then the inputs and the references should also be in a physically sensible range of values in order to establish this connection. A physical sound data basis is therefore desirable. The quality indicators help to avoid many of these unphysical values in the input and reference data by marking them as abnormal. However, it has to be mentioned that, according to three dataset samples, the quality indicators did not successfully mark the negative brightness temperature values.

For the application of the quality indicators, pre-processing functions were developed in this study and added to the newly developed pre-processing toolkit. The functions for applying the quality indicators can also be used separately from SPARE-ICE. For SPARE-ICE, the quality of the data has to be checked before [1] calling SPARE-ICE.

Even though a quality abnormality does not necessary mean that the data point affected is no longer usable for the analysis, caution is advised and the case should be examined more closely. As this is not feasible in SPARE-ICE due to the large amount of data, all points flagged by the selected quality indicators are deleted from the respective datasets for the re-training (see Section 5.2) and from the datasets for the retrieval (see Section 6.1) by using the pre-processing toolkit.

---

[1]There is no filter integrated into the IWP regressor and ice cloud detector component of SPARE-ICE that addresses poor quality input data.

In the following, the quality indicators as provided by the pre-processing toolkit are stated and their structure and inclusion in the pre-processing toolkit is briefly explained. A more detailed description of the way in which the quality indicators are applied to the respective data via the pre-processing toolkit is given in Appendix B.4.

Application of quality indicators

The pre-processing toolkit contains functions tailored for the applications of quality indicators on MHS, AVHRR, and 2C-ICE data. The quality indicators are already included in the respective datasets.

The individual quality indicators that are available for the AVHRR and MHS sensors can be found in *AAPP Documentation - Data Formats* (2021). For 2C-ICE, only one quality indicator exists. This quality indicator differs between the respective 2C-ICE releases. For 2C-ICE P1_R05, a description of the quality indicator can be found in Mace and Deng (2019). For 2C-ICE P1_R04, a description of the quality indicator can be found in Mace and Deng (2015).

For AVHRR, the quality indicators *calqual*, *scnlinqual*, and *qualind* are available. For MHS the quality indicators *qualind*, *chanqual* and *scanqual* are available. In addition, the MHS quality indicator *dataqual* is available in the MetOp-A data. This variable is missing in the NOAA-18 data. For 2C-ICE P1_R05 and 2C-ICE P1_R04, the quality indicator *dataquality* exists.

Each quality indicator represents quality flags, each of which has a different meaning. In the pre-processing toolkit, the user can choose which quality flags they want to select. Default settings are available (Appendix B.4 and Appendix D.1). The quality flags can roughly be divided into three groups: quality flags that indicate a calibration problem, GPS problem, or noise. Before applying the quality flags, every selected flag must first be retrieved from the quality indicator: the quality indicator contains integer numbers for each data point. To retrieve the quality flags, these numbers have to transformed to binary numbers first. After the conversion, the quality flags can each be accessed by their individual bit position assignment. It the bit position assigned to a particular quality flag is zero, the quality flag indicates normal behaviour.

A more detailed explanation of how to select the respective quality indicators and quality flags, as well as a brief justification for the default quality flag choices and and example of how the quality flags are extracted from the quality indicators can be found in Appendix B.4.

The functions from the pre-processing toolkit that deal with the application of quality flags are designed to handle a single dataset each time. In order to be able to apply the functions to several datasets in succession, the functions were embedded into wrapper functions. Through the wrapper functions, the datasets can also be processed in parallel. The available wrapper functions are listed in Appendix A.

For this study, all available quality indicators of MHS, AVHRR, and 2C-ICE with their respective default quality flag choices (see Appendix B.4) were used for pre-processing.

Chapter 5

Re-training of SPARE-ICE

For the generation of the new IWP data records, it is not only important to ensure high quality input data, but also that the same channel inputs for the retrieval are provided over the whole retrieval period and are available from both NOAA-18 and MetOp-A. This condition excludes AVHRR channel 3 from the training, because in some periods, AVHRR channel 3 switches between channel 3A and channel 3B.

As AVHRR channel 3 had to be removed from the original feature set, the number of input features for SPARE-ICE changed. This required a re-training of SPARE-ICE.

For the re-training, MHS, AVHRR and 2C-ICE data from the period January 2007 to December 2010 were used. Compared to Mrziglod (2018), who used data from January 2007 to August 2010 for his baseline experiment, the time period is slightly extended.

This chapter first describes the computational environment and preparation procedure of inputs datasets that I used to re-train SPARE-ICE. Afterwards, the re-training results for the IWP regressor and the ice cloud detector are presented and discussed. For the preparation of the input datasets, I applied the newly developed pre-processing toolkit and introduced a new balancing function. The setup of the individual SPARE-ICE components as used for the re-training is described in Appendix B.

5.1 Computational environment

Calculations for SPARE-ICE took place on the high-performance computing system Levante, which is provided by the Deutsche Klimarechenzentrum (DKRZ) Hamburg. Levante uses SLURM as cluster management and job scheduling system. In SLURM,

Package	Version
scikit-learn	1.1.1
numpy	1.23.4
xarray	2022.3.0
pandas	1.4.2
matplotlib	3.5.2
typhon	0.9.0

Table 5.1: Versions of the python packages used to create the two new SPARE-ICE IWP data records.

a set of nodes is grouped to form a partition. The training and retrieval of SPARE-ICE were carried out on so-called *compute* partitions. On a compute partition, a maximum of 512 nodes per job is available with a range of memory per CPU between 960 MB and 380 MB[1]. The default maximum runtime on *compute* is eight hours. For the analysis of SPARE-ICE, I used an *interactive* partition with 100 GB.

The DKRZ provides precompiled python environments on Levante. In the course of this study, the environment python 3/2022.01-gcc-11.2.0 which provides python version 3.9.9 is used. Jupyter notebooks from the DKRZ JupyterHub multi-server are used as main development environment. For development and plotting the packages as listed in Table 5.1 serve as a basis.

5.2   Preparation of input and target datasets

Figure 5.1 illustrates the preparation of the training data and lists the final input features and targets that I used for the re-training of the IWP classifier and ice cloud detector. As inputs, HDF datasets from MHS, 2C-ICE, and AVHRR were required. These were supplemented with land-sea mask information and surface elevation information. The land-sea mask and surface elevation were auxiliary information that did not require further preparation.

In the following, the individual steps, as shown in Figure 5.1, are elaborated.

Pre-processing

In order to provide SPARE-ICE with reliable data for its training, the MHS, 2C-ICE, and AVHRR datasets used for the re-training are first pre-processed with the help of the newly developed pre-processing toolkit. During this process, quality flags are applied and duplicates are removed.

---

[1]`https://docs.dkrz.de/doc/levante/running-jobs/partitions-and-limits.html`, last visited: 03.07.2023, 11:03

HDF datasets

2C-ICE
(CloudSat & CALIPSO)

MHS
(NOAA-18)

AVHRR
(NOAA-18)

Pre-processing

Application of quality flags

Removal of duplicates

Collocating
1. stage

Collocate & Collapse to MHS
(Δs = 7.5 km, Δt = 10 min)

Collocating
2. stage

Collocate & Collapse to MHS
(Δs = 7.5 km, Δt = 30 s)

Homogenisation

$\#(IWP) > 10$, $\frac{\sigma(IWP)}{\overline{IWP}} < 0.5$

Balancing

Latitudinal balancing
(Δlat = 5°,
$\#clouds_{\Delta lat}$ = $\#clearsky_{\Delta lat}$ = 14761)

Feature and
target selection

Inputs

- land-sea mask
- surface elevation

- channel 2, 4, 5
- σ(channel 2)
- σ(channel 5)
- channel 5 - channel 4

- channel 1, 2, 3, 4, 5
- channel 5 - channel 3
- latitude
- scan position
- solar azimuth angle
- solar zenith angle

Targets

$\log_{10}\overline{IWP}$

$\overline{IWP} > 0\ g/m^2$

Training dataset
splitting

75 % for training & validation   25 % for testing

Figure 5.1: Simplified sketch of the data preparation procedure for the re-training of SPARE-ICE. The features used in this study for re-training SPARE-ICE are listed within the box 'Inputs'. The targets for the IWP regressor and ice cloud detector are listed within the box 'Targets'. Input features and targets listed within the yellow, red, and blue boxes originate from AVHRR, MHS, and 2C-ICE respectively. The grey box lists auxiliary datasets. '#' stands for 'number of'. The sketch is based on Figure 4.2 from Mrziglod (2018), but was considerably modified to align with this study.

For quality control, all available quality indicators for MHS and AVHRR from the pre-processing toolkit are selected. For 2C-ICE, the quality indicator *cloudsatR05_dataquality* is selected. For the quality flags, the default quality settings for each quality indicator as listed in Table D.1 are chosen. After the quality control, all three datasets are checked for multiple data entries respectively.

Collocating

After quality control, the pre-processed files undergo a collocation procedure to bundle data points that roughly represent the same scene. The collocation procedure is subdivided into two stages: during stage one, MHS and 2C-ICE are collocated. During stage two, the MHS and 2C-ICE collocations are further collocated with AVHRR.

All time limits and spatial distance restrictions as chosen in this study for collocating are in accordance with Mrziglod (2018). An overview of the individual collocation inputs and settings used for training by Holl et al. (2014), Mrziglod (2018), and in this study can be found in Appendix D, Table D.2.

For the first collocation stage, MHS and 2C-ICE are collocated with a Δs of 7.5 km and a Δs of 10 min. Mrziglod (2018) stated that the choice of Δs of 7.5 km and Δt of 10 min for creating the MHS_2C-ICE collocations represents a compromise between getting too much noise through a change of scene and too few collocations for the limb angles of the instruments. When fixed at 7.5 km, Δs represents slightly less than half of the MHS footprint diameter at nadir, which is approximately 17 km at 870 km nominal orbital altitude (*NOAA KLM User's guide*, 2014).

The spatial extent of one 2C-ICE pixel covers less than 10 % of one MHS footprint (see Subsections 3.2.1 and Section 3.3). According to Δt, this implies that several 2C-ICE pixels can be assigned to one MHS pixel during the collocation process. With $\Delta t = 10$ min and $\Delta s = 7.5$ km, the maximum number of 2C-ICE pixels that can be assigned to one MHS pixel is 15 (Mrziglod, 2018). If more than one pixel is assigned to another pixel during the collocation process, the measurement values of multiple assigned pixels are averaged (*collapsed*) and their average, standard deviation, and the number of assigned pixel is stored. In visual terms this roughly means that if the first instrument has a larger footprint than the second instrument, the footprints from the second instrument are averaged over the footprint size of the first instrument. The result then contains the coordinates of the larger footprint. Since the footprint of the sensors from which 2C-ICE IWP was retrieved is much smaller than the footprint of MHS, 2C-ICE is collapsed onto MHS during the first stage of the collocation procedure. The collapsed IWP averages from 2C-ICE adopt the coordinates from the MHS pixels.

For the second collocation stage, the collapsed MHS_2C-ICE collocations are collocated with AVHRR with $\Delta t = 30$ s and $\Delta s = 7.5$ km. Δt for the second collocation stage is much smaller than Δt from the first collocation stage to make the collocation search more efficient. As MHS and AVHRR are both on NOAA-18 and their
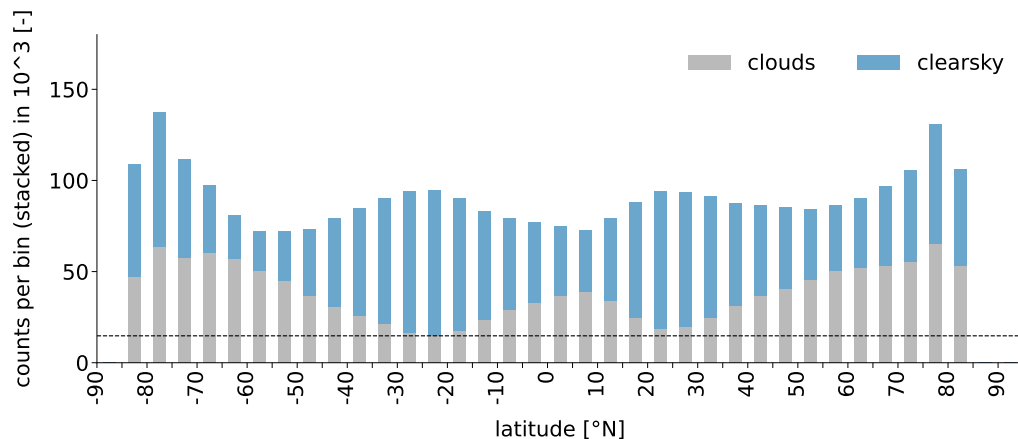
Figure 5.2: Number of cloud and clearsky collocations of the re-training input dataset for SPARE-ICE for each 5° latitudinal band after quality control, removal of MODPs and homogenisation. The black dashed line marks $n\_lat = 14761$, which was used for balancing during re-training.

scans are almost synchronised, not many collocations are lost due to this harsh time difference limitation (Mrziglod, 2018).

Since the footprint of AVHRR is much smaller than the footprint of MHS (see Subsection 3.2.2), the resulting collocations are afterwards collapsed onto MHS.

Homogenisation

As even several 2C-ICE pixels can not fully cover the same scene that one MHS pixel represents, a systematic error is introduced to each collocation (Mrziglod, 2018). To limit this effect, inhomogeneity filters are applied as in Mrziglod (2018) on the MHS_2C-ICE_AHVRR co-collocations for this study: More than ten 2C-ICE pixels must have been collapsed onto one MHS pixel and the standard deviation of the collapsed 2C-ICE pixels must be smaller than half of their average.

Balancing

The focus of the training is to reproduce the IWP distribution across the zonal latitudes globally as consistently well as possible. However, most collocations for the training are found at the poles (cf. Figure 5.2). As a result, the poles would have been over-represented in the training. This would have led to the model mainly trying to match the IWP values of 2C-ICE in the polar regions. Hence, atmospheres from the tropics, subtropics and mid-latitudes would be analysed by a model biased towards polar atmospheres. This can worsen the accuracy of the retrieval in these warmer regions. To avoid this, the data points from the training dataset were balanced across latitudes.

	After pre-processing	**After ho-mogenisation**	**After latitudinal balancing**
Number of co-collocations (collapsed to MHS)	6423574	3078810	1003748

Table 5.2: Number of available co-collocations for the re-training of SPARE-ICE.

For this purpose, a latitudinal balancing function was created. The latitudinal balancing function differentiates between cloud (IWP > 0 g/m^2) and clearsky (IWP $= 0$ g/m^2) pixels and is applied to cloud and clearsky pixels respectively over the whole training period.

For the latitudinal balancing of cloud pixels, the homogenised co-collocations are first subdivided into 36 latitudinal bands between 90°S and 90°N. Each of the latitudinal bands has a width of 5°. Afterwards, all latitudinal bands are compared to find out which latitude band has the lowest number of clouds greater than zero. This number is defined as n_lat. The number n_lat is then selected as the upper boundary of cloud counts for each latitudinal band. If a band contains more than n_lat cloud pixels, n_lat cloud pixels are randomly selected from that band. This ensure that in the end each band that contains a cloud pixel has the same number of cloud pixels as the other bands. For the random selection of points from the latitudinal bands, a seed[2] of 1234 is fixed. If desired, the minimum number of pixels for each latitudinal band can be set manually.

The same balancing procedure with the same seed for random pixel selection is applied to clearsky pixels. After balancing, the clearsky and cloud pixels are combined into one dataset.

For SPARE-ICE, I set n_lat to 14761 for clouds and clearsky pixels respectively. Figure 5.2 illustrates the number of cloud and clearsky collocations that existed before balancing. After balancing, 2 x 14761 = 29522 collocations remain at each latitudinal band between 85°S and 85°N.

It may be noted that, according to the handed over SPARE-ICE code, Mrziglod (2018) also applied a latitudinal balancing over 5° latitudinal bands and set n_lat to 40000 pixels for clouds and clearsky respectively. However, his balancing function was not yet fully elaborated and n_lat was set too high, so that an equal distribution was not guaranteed after balancing.

---

[2]Generally, random functions are pseudorandom. Such functions calculate their random outputs. Hence, the name pseudorandom. If a seed is given, the function is primed with that seed and will generate the same outputs every time it is called with that seed again.

Feature and target selection

After balancing, the preparation of the co-collocations is complet and the input features and targets to be passed to the IWP regressor and ice cloud classifier must be selected.

The choice of input features was already optimised by Mrziglod (2018) in his baseline experiment to such a degree that a very good agreement between SPARE-ICE and 2C-ICE for the year 2008 and the testing data was achieved. Hence, this study takes the feature selection of Mrziglod (2018) from his baseline experiment as a basis and does not further optimise the feature selection based on error metrics.

The data basis for the feature and target selection is comprised of the homogenised and balanced co-collocation datasets and two auxiliary files, which are the land-sea mask dataset and the elevation dataset.

For the training of the IWP regressor and the ice cloud detector, I used the input features as listed in Figure 5.1. The feature selection is identical to the one from the baseline experiment from Mrziglod (2018), except that AVHRR channel 3 is removed. Short justification for the feature selection are listed in Mrziglod (2018). A comparison between the feature and target selections from this study and those from Mrziglod (2018) and Holl et al. (2014) can be found in Appendix D.3.

Training dataset splitting

To train the IWP regressor and the ice cloud detector and to evaluate their training results, the MHS_2C-ICE_AVHRR co-collocations have to be split into a *train_validation_data* dataset and a *test_data* dataset (see Appendix B.2).

For this purpose, the co-collocations are shuffled and then randomly split with a seed of 5 into *train_validation_data* and *test_data* by using a *test_size* of 25 %. cv_folds (see Appendix B.2) was set to 5. If cv_folds is set to 5, 20 % of the *train_validation_data* are used for validation during the training of the MLP regressor. The parameter choices are in agreement with Mrziglod (2018).

The final number of data points that are available for training, validation and testing of the MLP are listed in Table 5.3. However, note that not all data points can be used for training and testing the MLP, as the MLP uses only data points with IWP $> 0 \ g/m^2$ for training. On the other hand, the DTC uses all available co-collocations for its training. In total, there are about 2.3 times as many training points available for this study as for the study from Mrziglod (2018).

	Training	Validation	Testing	Total
Number of co-collocations	602249	150562	250937	1003748

Table 5.3: Number of available MHS, AVHRR, 2C-ICE co-collocations for training, validation, and testing during the re-training phase of SPARE-ICE. The training period covers the time period from January 2007 to December 2010.

After the training datasets were created, SPARE-ICE was re-trained on *train_ validation_data* and evaluated on *test_data*.

5.3   Re-training results

In the following, the re-training results for the IWP regressor and ice cloud detector are presented. The re-training of SPARE-ICE was performed on NOAA-18 data from January 2007 to December 2010 with 2C-ICE as reference.

The lower limit of 2C-ICE IWP is 0 g/m^2. As 2C-ICE is used for the re-training of SPARE-ICE, this introduces an additional bias to SPARE-ICE[3]. According to its setup, SPARE-ICE can also only generate IWP values greater than 0 g/m^2.

5.3.1   IWP regressor

The optimal MLP regressor that resulted from the grid search during training has the attributes as listed in Table 5.4. Brief explanations of the parameter functionalities can be found in Appendix B.2. Detailed descriptions of the parameters can be found in the scikit-learn documentation of the scikit-learn MLP regressor[4].

The trained optimal IWP regressor achieved a final R^2 score (coefficient of determination) of 0.838. The score was calculated on *test_data* during training by using the *r2_score* method from sklearn.metrics. The R^2 score is calculated as follows:

$$R^2 = 1 - \frac{\sum_i (y_i - y_{i,predict})^2}{\sum_i (y_i - \overline{y})^2}$$

---

[3]To date, IWP has been estimated based on measured or modelled variables. It is a dependent variable. Both the input data and the model used to calculate IWP introduce uncertainties into the determination of the IWP values. Consequently, a presented IWP value is always associated to a possible distribution of IWP values. For IWP values close to zero, this may cause the distribution to also contain negative IWP values, even though negative IWP values make no physical sense by definition. If these negative values are set to zero, for example, the dataset will have a slight positive bias. For 2C-ICE, no valid range of IWP is specified in the official 2C-ICE P1_R05 documentation (Mace & Deng, 2019). However, no negative IWP values were found in 2C-ICE during the analysis conducted in this study. Thus, the lower limit of 2C-ICE is assumed to be 0 g/m^2.

[4]`https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html`, last visited: 28.07.2023, 23:01

Number of input features	16
Hidden layer sizes	(15,10,3)
Number of outputs	1
Hidden layer nodes activation function	tanh
Output activation function	identity
Solver	lbfgs
Alpha	0.0001
Shuffle	True
Random state	0
Learning rate	constant
Initial learning rate	1
Max iterations	6000
Tolerance	0.0001

Table 5.4: Configuration of the MLP IWP regressor resulting from the re-training of SPARE-ICE.

with $y_1, \ldots, y_n$ being the IWP reference data from *test_data* and $y_{1,predict}, \ldots, y_{n,predict}$ being the predicted IWP values of the IWP regressor based on the data from *test_data*.

The best possible score returned by the *r2_score* method from sklearn.metrics is 1. If the model always predicts the average of the target data, independent of the inputs, the score is 0. For arbitrary worse performance, the score can also be negative[5]. A score of 0.838 implies that the IWP regressor was able to explain 83.8 % of the variations of the test dataset.

The scatter density plot in Figure 5.3 shows the agreement between collocated predicted IWP values from SPARE-ICE and 2C-ICE IWP from *test_data* in double logarithmic representation. The data does not yet include the ice cloud flags from the ice cloud detector. All density regions that have less than one collocation are represented in white.

The majority of collocation points within the range of 10^1 g/m^2 and 10^4 g/m^2 are placed along the diagonal and indicate a high correlation between 2C-ICE and SPARE-ICE within this IWP range. This good to very good fit between SPARE-ICE retrieval data and 2C-ICE is expected as SPARE-ICE was trained to match 2C-ICE as best as possible under the given training setup restrictions. Even though the 2C-ICE IWP data points were not included in *train_validation_data*, *test_data*

---

[5]https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html, last visited: 26.07.2023, 14:30

Performance on testing dataset
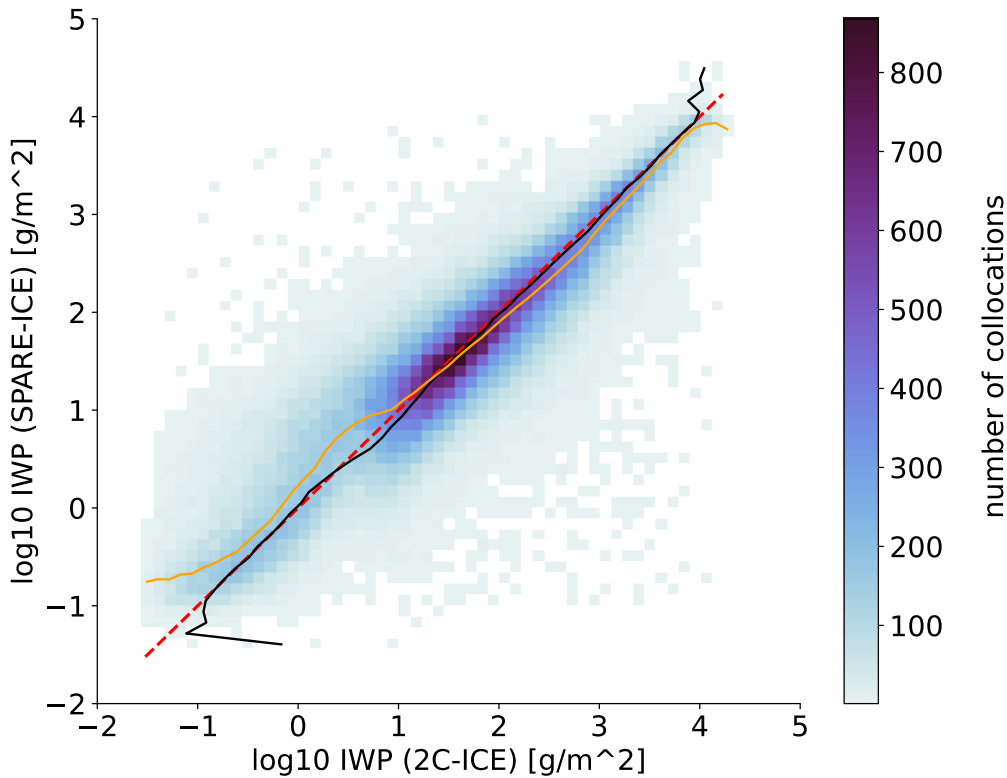
Figure 5.3: Scatter density plot showing the absolute number of collocations between SPARE-ICE IWP and 2C-ICE IWP for *test_data*. The testing data collocations were binned according to the reference 2C-ICE IWP and SPARE-ICE IWP values. The colours indicate the absolute number of collocations with a particular combination of 2C-ICE IWP and SPARE-ICE IWP. The red dashed line is the diagonal and marks collocations that have a very good fit between SPARE-ICE and 2C-ICE IWP. The black line represents the median for 2C-ICE when SPARE-ICE is kept fix. The orange line represents the median for SPARE-ICE when 2C-ICE is kept fix. For example, when 2C-ICE is 1 g/m^2, the median SPARE-ICE is 3 g/m^2. Bins without collocations are represented in white.

exhibits the same statistics as *train_validation_data* (cf. Appendix Figure E.2 and Figure E.3 and Holl et al. (2014)).

For 2C-ICE IWP < 10 g/m^2, the median of SPARE-ICE first moves away from the diagonal, then returns to the the diagonal again between about 0.05 g/m^2 and 10 g/m^2 2C-ICE IWP. Finally, the median of SPARE-ICE levels off at about 0.07 g/m^2 SPARE-ICE IWP for 2C-ICE IWP < 0.05 g/m^2.

For 2C-ICE IWP < 10 g/m^2, the IWP regressor from SPARE-ICE tends to retrieve larger IWP values than 2C-ICE. Hence, as already indicated by Mrziglod (2018), the IWP regressor seems to perform worse low IWP regions. Overall, the IWP regressor

performs better with increasing IWP.

It follows that, assuming that IWP from 2C-ICE represents the truth, the SPARE-ICE IWP regressor is most reliable for IWP $\geq 10 \ g/m^2$. $10 \ g/m^2$ is therefore considered the lower sensitivity limit of SPARE-ICE.

This is in accordance with the findings from Holl et al. (2014). Even though Holl et al. (2014) used a different input feature setup as compared to this study (cf. Table D.3), they concluded, based on the investigation of the median fractional error of SPAR-ICE and scatter plots, that the lower sensitivity limit of their SPARE-ICE retrieval is $10 \ g/m^2$.

Ringel (2023) used an ANN to retrieve IWP based on airborne MW measurements that were taken between Norway and Greenland. A comparison with ICON (ICOsahedral Non-hydrostatic model) (Zängl, Reinert, Rípodas & Baldauf, 2015) showed that his results were most promising for IWP values above $100 \ g/m^2$. In contrast, SPARE-ICE is sensitive down to $10 \ g/m^2$. A larger sensitivity range of SPARE-ICE is expected, however, as in addition to MW, also IR channels are used. This allows SPARE-ICE to better resolve thin ice clouds.

Compared to Mrziglod (2018, Figure 4.7), the scatter density plots for SPARE-ICE and 2C-ICE are almost identical. Mrziglod (2018) represents all density regions that have less than 5 collocations in white. This study set no lower threshold for colouring. For an easier comparison between both scatter density plots, Figure E.1 was re-created with a minimum threshold of 5 collocations for the colouring of regions and finer bin width. Figure E.1 can be found in Appendix E.

The very high degree of correlation between the scatter density plot for the IWP regressor generated in this study and that one of Mrziglod (2018) suggests that removing AVHRR channel 3 from the feature set does not significantly affect the IWP regressor. The statement of Mrziglod (2018) that SPARE-ICE underestimates IWP for higher IWP values (cf. description of Figure 4.7 in Mrziglod (2018)) can only be partly supported by this study based on the scatter density density plot alone. Figure 5.3 only reveals a slight tendency with regard to general underestimation of IWP by the IWP regressor for IWP $\geq 10 \ g/m^2$.

5.3.2   Ice cloud detector

A single DTC was trained as ice cloud detector. As target during the training, labels based on 2C-ICE IWP values were used. A target data point was labeled as cloudy if its average IWP values was greater than $0 \ g/m^2$ (see Appendix D.3). *test_data* contained 49.95 % clearsky and 50.05 % cloud collocations. Hence, the testing dataset was well-balanced.

The resulting DTC from the training has the attributes as listed in Table 5.5. Brief explanations of the functionalities of most of the parameters mentioned in Table 5.5 can be found in Appendix B.3. Detailed descriptions of all the parameters can be found in the scikit-learn documentation[6].

Maximum depth	12
Number of input features	14
Number of outputs	1
Number of internal nodes + leaf nodes	6303
Random state	5
Splitter strategy	best
Criterion	gini

Table 5.5: Configuration of the DTC ice cloud detector resulting from the re-training of SPARE-ICE.

The DTC achieved a final score of 0.886 on *test_data* during training. The score represents the average number of the successful predictions of the DTC compared to the target. This means that the accuracy the DTC in respect to *test_data* is 0.886.

For a better breakdown of performance, a confusion matrix was created. The confusion matrix in Figure 5.4 shows the classification errors of the re-trained DTC on *test_data* in comparison to 2C-ICE. For 2C-ICE, clearsky and cloud pixels were identified with the same IWP limit as for SPARE-ICE. The confusion matrix is normalised by rows.

According to Figure 5.4, the precision of the classifier is 0.86. This means that 86 % of the predicted clouds were actually clouds. The recall of the classifier is 0.92, meaning that 92 % of the clouds of the dataset were correctly identified as clouds by the classifier. The F1 score combines both metrics by taking the harmonic mean of precision and recall. The F1 score of the model is 0.889. For an F1 score of 1, the classifier would have perfectly classified each data point compared to the target.

There are too few metrics available in Mrziglod (2018) to compare the confusion matrice from this study and from the study of Mrziglod (2018) without further assumptions. In order to be able to make a comparison between both confusion matrices nevertheless, it is assumed that the target values from the testing dataset

---

[6]`https :// scikit -learn .org / stable / modules / generated / sklearn .tree .DecisionTreeClassifier.html`, last visited: 28.07.2023, 23:10

Ice Cloud Detector - Performance
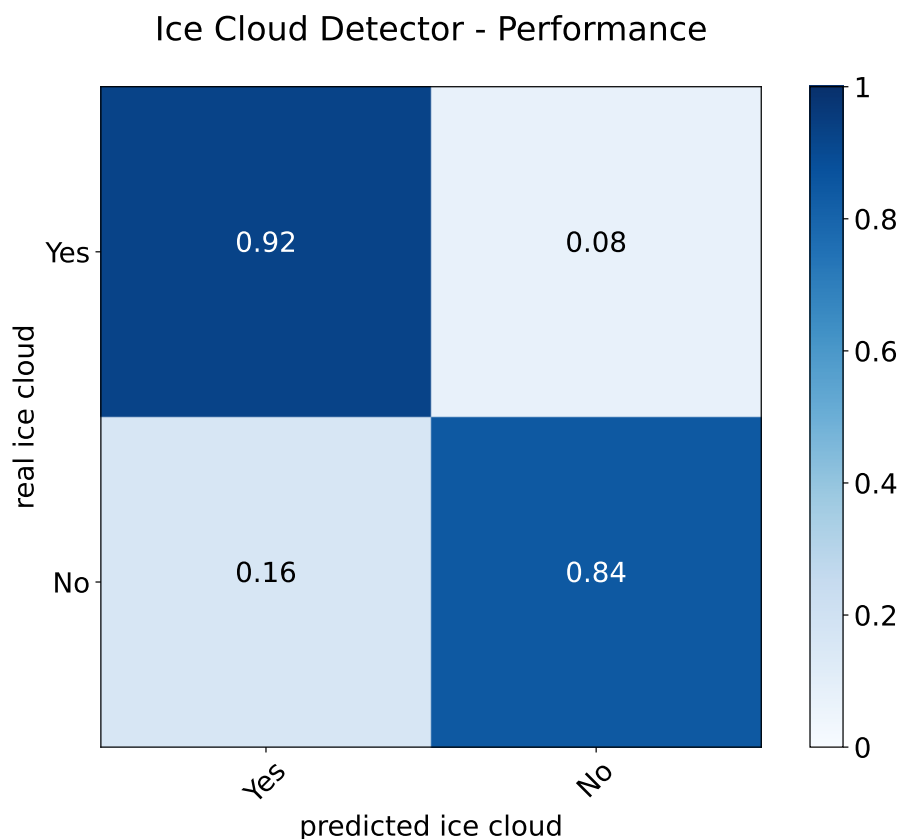
Figure 5.4: Confusion matrix of the ice cloud classifier after re-training of SPARE-ICE. The numbers of correct and incorrect predictions are based on *test_data*. The confusion matrix is normalised by rows.

used by Mrziglod (2018) contained an equal number of cloud and clearsky pixels. This assumption is supported by the description of Table 4.1 in Mrziglod (2018). However, as noted in Chapter 5, the balancing that Mrziglod (2018) applied on his training data was probably incorrect. Since no further information is given in Mrziglod (2018), it cannot be guaranteed that there were actually an equal number of cloud and clearsky pixels in his testing dataset. However, if one assumes that the number of pixels in both classes were equal, the trained DTC from Mrziglod (2018) had a precision of 0.88 and a recall of 0.94, resulting in a F1 score of 0.909. Thus, in relation to the respective testing datasets, the DTC from Mrziglod (2018) performs slightly better.

Mrziglod (2018) possibly trained a RFC and included AVHRR channel 3 in the training of his ice cloud detector. This study only trained a single DTC and excluded AVHRR channel 3 from the training. Hence, assuming that Mrziglod (2018) has trained a RFC, training a single DTC seems to give a comparable good result to training several DTCs during the training of the RFC. The removal of AVHRR channel 3 does not seem to have a strong effect on the performance of the ice cloud detector.

The MLP used by Holl et al. (2014) for ice cloud classification, has a false positive rate of about 21 % and a false negative rate of 9 % (cf. Holl et al. (2014, Figure 9), for a cut off of 0.5). Holl et al. (2014) identify a scene as cloudy during training if IWP $> 10\ g/m^2$. In contrast, the trained DTC of this study has a false positive rate of about 16 % and a false negative rate of about 8 %. The false negative rates thus agree very well between both ice cloud detectors. The MLP from Holl et al. (2014), however, has a false positive rate that is about five times lower. Both false positve rates are still small. Since Holl et al. (2014) uses a stricter IWP criterion for ice cloud identification than this study, it is not surprising that less clearsky pixels were incorrectly identified as clouds. Based on these two metrics and under consideration of the different thresholds used for cloud identification, the single DTC from this study performs very well compared to the MLP used by Holl et al. (2014). Thus, the DTC seems to represent a very good alternative to the MLP for ice cloud detection.

Chapter 6

IWP retrieval records

With the re-trained IWP regressor and ice cloud detector from SPARE-ICE, I generated two new temporally extended IWP data records. The first IWP data record is based on NOAA-18 and covers the period from January 2007 to October 2018. The second one is based on MetOp-A and covers the period from May 2007 to December 2019. Both data records extend the IWP product range of SPARE-ICE by a so-far unprecedented time period and can be used to analyse IWP over more than one decade.

This chapter first describes the processing steps that were carried out to produce both IWP data records. The two new data records are then presented through an application example that explores the derivation of the diurnal IWP cycle for different regions. Furthermore, their global IWP distribution is presented and compared with the 2C-ICE IWP product for the ascending nodes of NOAA-18, MetOp-A, and CloudSat during 2008.

6.1   Retrieval process

The computational environment used for the retrieval is identical to that used during the re-training of SPARE-ICE (Section 5.1).

For the generation of the retrieval products, collocations between MHS and AVHRR data from NOAA-18 and MetOp-A were generated. MHS and AHVRR were each taken from the same satellite for this purpose and pre-processed with the pre-processing toolkit. MHS and AVHRR were pre-processed in the same way and with the same settings as the MHS and AVHRR input data for the re-training (see Section 5.2). However, neither homogenisation nor latitudinal balancing was applied to both datasets. After pre-processing, MHS and AVHRR were collocated with $\Delta s = 30$ s

and $\Delta t = 7.5$ km. The resulting collocations are called *MHS_AVHRR*.

In combination with the elevation dataset and land-sea mask from the training, MHS and AVHRR are the only inputs required by SPARE-ICE for the retrieval. This means that SPARE-ICE relies almost completely on passive sensors for the retrieval of IWP. In contrast to the re-training, no IWP inputs based on active sensors are required.

After creating the collocations, SPARE-ICE is applied on MHS_AVHRR from MetOp-A and NOAA-18 respectively. For the retrieval, SPARE-ICE is configured according the setup results from the re-training (Section 5.3). The resulting IWP retrieval records have a global coverage (except in the immediate vicinity of the poles). One IWP data point represents the area covered by one MHS pixel. Each IWP dataset from the new data records contains IWP in g/m^2 and ice cloud flags. The retrieved IWP values are in the range of 10^{-2} g/m^2 to 10^4 g/m^2.

A more detailed explanation of the retrieval process that focuses on the implementation details of the retraining can be found in the description of the IWP regressor and ice cloud detector in Appendix B.2 and Appendix B.3.

6.2 IWP products

In this section, the two SPARE-ICE IWP data records are first presented by means of an application example. The application example points out how the diurnal cycle of IWP can be estimated based on the new data records. The application example is introduced using annual zonal means of IWP from the MetOp-A and NOAA-18 based data records and 2C-ICE IWP. The variations of the annual zonal means over the years are briefly explored in this context.

Subsequently, an insight into the global IWP distribution of the new data records is given on the basis of data from 2008. Pre-processed IWP data from 2C-ICE P1_R05 is shown alongside the respective SPARE-ICE data records. The pre-processing was performed using the pre-processing toolkit. The default quality flags were selected from the quality indicator. For this study, 2C-ICE was used for accuracy estimation.

All results presented in this chapter refer to allsky IWP values. No ice cloud flags were applied. The data is not collocated. All data used is averaged over 5° x 5° grid cells.

6.2.1   Application example: diurnal cycle of IWP

NOAA-18, MetOp-A, and CloudSat are in sun-synchronous orbits. This means, that the their orbits's geometry with respect to the sun stays nearly fixed (Boain, 2004). As a result, the lighting conditions along the sunlit groudtrack remain approximately unchanged (Boain, 2004), as long as there are no changes in orbits.

A common approach to determine the diurnal cycle of IWP based on sun-synchronous satellites is to split the data into daytime and nighttime values and then to compare the data from both times of day with each other for a specific location. As sun-synchronous satellites only record two times of day for each location for one global data record, intuitively, this approach seems to provide only two IWP values per location for the analysis of the diurnal cycle. Assuming, e.g. that a sinusoidal pattern for the diurnal cycle is expected, this approach is not ideal, as the cycle would be heavily undersampled. (Duncan & Eriksson, 2018).

In fact, however, the approach of splitting the records into daytime and nighttime datasets is much more flexible and can provide IWP at various times of day when applied to different IWP products or to a long data record of IWP. For these cases, the diurnal cycle of IWP can be analysed by evaluating IWP under consideration of the respective ECTs of the orbits to which the datasets can be assigned according to their input sources.

This requires as sources either many satellites with different ECTs, or at least one satellite whose orbit drifts over time. The approach can then take advantage of the fact that, due to the different sun-synchronous satellites or due to the drift of at least one satellite, different local times are always recorded at a given location. Measuring IWP over those different times then gives the diurnal cycle of IWP at this given location.

If one uses the drift of a satellite for the estimation of the diurnal cycle, this approach generally requires a long time series. For a long time series, the IWP values can be averaged over multiple daytime and nighttime values at a specific location. For example, if the ECT of a satellite's orbit has changed from 01:00 pm to 02:00 pm over the course of a year, one can obtain an IWP value for a given location that represents the average IWP at that location between 01:00 pm and 02:00 pm over that year. The more constant the drift in one temporal direction, the more evenly the IWP values of different times of day are represented over the corresponding averaging window.

In order to differentiate between daytime and nighttime values, the IWP data points from the different IWP data records were assigned to ascending or descending nodes.

For this purpose, functions for the SPARE-ICE products as well as for the 2C-ICE product were created within the scope of this study. The functions differentiate between ascending and descending nodes based on the gradient of the latitude coordinates between two successive data points. A short description of those function can be found in Appendix B.5. Due to the setup of the functions, some scan lines are discarded from the SPARE-ICE data records when they are assigned to a specific node. This does not apply to the 2C-ICE data records (see Appendix B.5).

When comparing the newly generated SPARE-ICE data records with 2C-ICE IWP, one has to consider that the CPR from CloudSat faced serious battery anomalies in 2011. This had a negative effect on the 2C-ICE product. For 2011, 2C-ICE should therefore not be used as a reference for IWP accuracy estimation. Due to the battery problem, it was decided to operate CloudSat in DO-OP mode only from 2011 onward. This means that 2C-ICE should only be used as reference during the ascending nodes from 2011 onward. The 2C-ICE data ends in July 2019.

Figure 6.1 and Figure 6.2 show the annual zonal mean values of IWP for the ascending and descending node, respectively. The zonal means include both clearsky and cloudy pixels. As NOAA-18, MetOp-A, and CloudSat are sun-synchronous, each annual zonal mean represents the IWP zonal mean over the local times recorded by each satellite over one year.

Over the whole retrieval period between 2007 and 2018, MetOp-A remains in its mid-morning orbit that crosses the equator at about 09:30 am local time (descending node) (Klaes & Holmlund, 2007). This means that within this period, the annual zonal averages of IWP are based purely on values found around 09:30 pm (ascending node), or 09:30 am (descending node). Thus, the early night is represented in the ascending node (Figure 6.1), and the late morning hours in the descending node (Figure 6.2).

Between 2007 and 2010, NOAA-18 was roughly on the same orbit as CloudSat. Both were placed in an afternoon orbit, which passes the equator at about 01:45 pm local time during its ascending node. This means that the zonal mean values of the NOAA-18 SPARE-ICE product reflect the IWP content of the atmosphere at the respective latitudes at approximately 01:45 pm (ascending node, Figure 6.1) and 01:45 am (descending node Figure 6.2).

The orbital drift of NOAA-18 became clearly evident in 2010. Since about 2010, NOAA-18 drifted away from CloudSat's orbit and approached an orbit in 2019 that crosses the equator at a similar local time (ascending node) as the orbit (ascending node) from MetOp-A (c.f. Figure 3.3). Between 2010 and 2018 the annual zonal
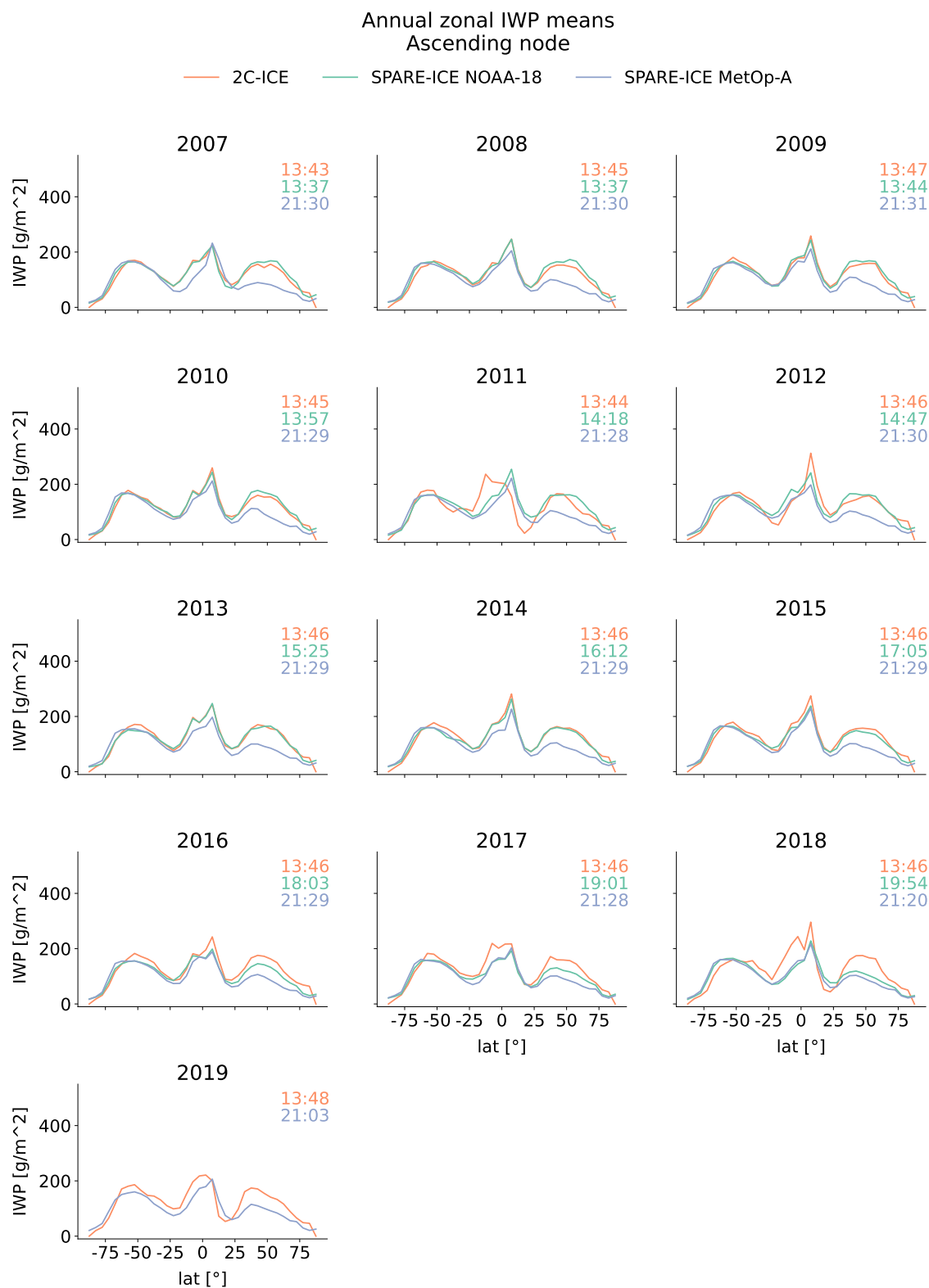
Figure 6.1: Annual zonal IWP averages (5° wide bins) from the NOAA-18 and MetOp-A based SPARE-ICE data records and 2C-ICE during the ascending nodes of NOAA-18, MetOp-A, and CloudSat respectively. 2C-ICE and the input data for the SPARE-ICE retrievals were quality checked and cleared of MODPs via the pre-processing toolkit. The retrieval datasets are not collocated. In the upper right corner of each subplot are the annually averaged ECTs of the respective satellites for the ascending nodes. The ECTs of CloudSat are used for 2C-ICE.

Annual zonal IWP means
Descending node

--- 2C-ICE --- SPARE-ICE NOAA-18 --- SPARE-ICE MetOp-A
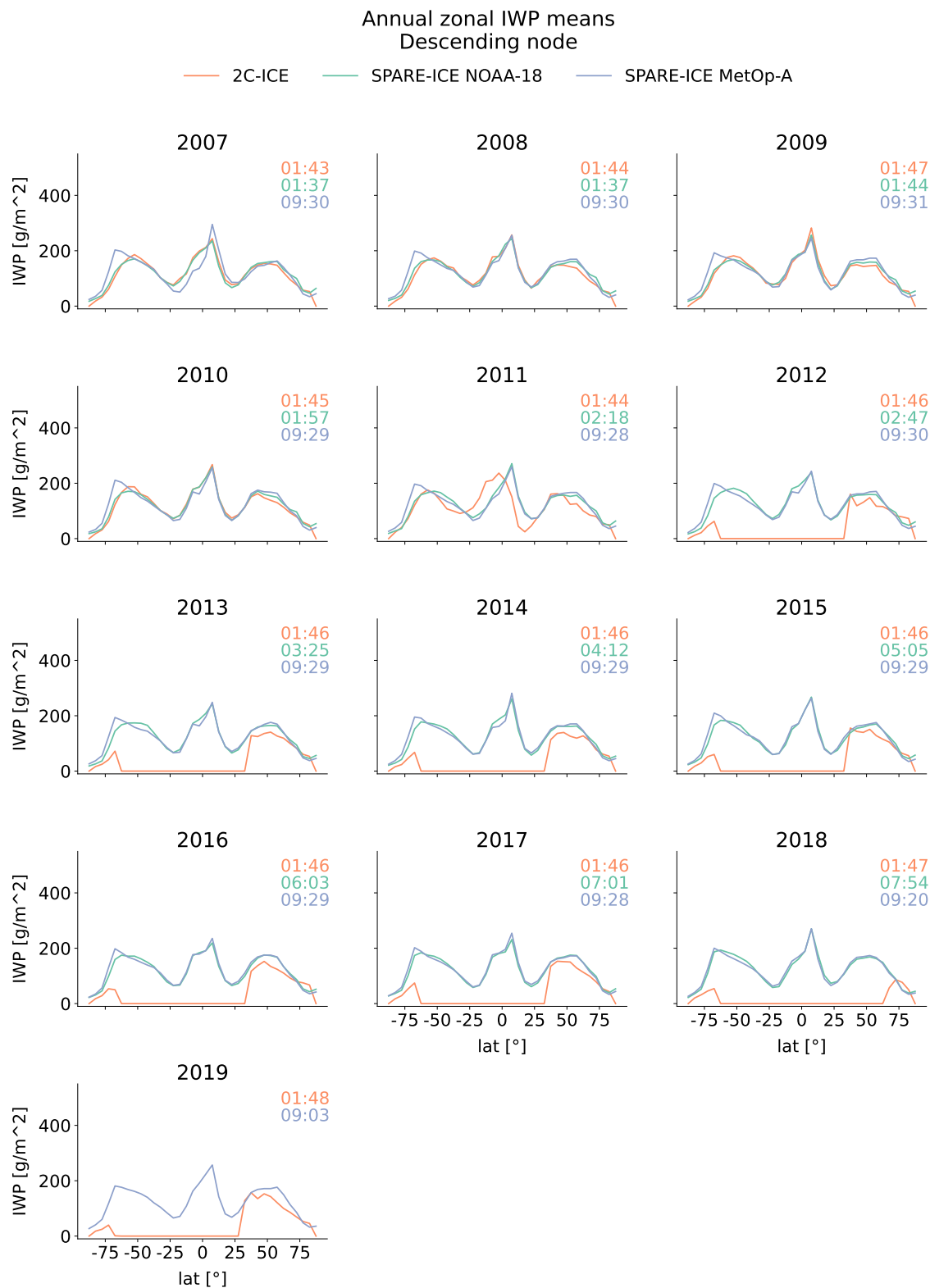
Figure 6.2: As in Figure 6.1, but for the descending nodes of NOAA-18, MetOp-A, and CloudSat respectively.

means of IWP from the NOAA-18 based retrieval for the ascending node refer to local times of day between early afternoon and early evening (see Figure 6.1). Over this time period, the annually averaged local ECTs of the ascending nodes of NOAA-18

move monotonically from the early afternoon hours to the early evening hours. The ECTs for the descending nodes move monotonically from the nighttime hours to the morning hours (see Figure 6.1). In 2019, NOAA-18 crossed the equator at about 08:43 pm local time during the ascending node of its orbit. Hence, in 2019, NOAA-18 and MetOp-A record similar local times globally. Since no more MHS data have been available from NOAA-18 since October 2018 (c.f. Section 3.1), and therefore no IWP retrieval data could be generated from that point on, the comparison between the NOAA-18 and MetOp-A based products is only possible until 2018.

The convergence between the local ECTs from the orbits of NOAA-18 and MetOp-A can also be seen in Figure 6.1. In the northern mid-latitudes the record based on NOAA-18 suggests significantly higher (about 100 g/m^2) IWP values than the MetOp-A record at the beginning of the retrieval period for the ascending nodes. By 2018, this difference between the MetOp-A and NOAA-18 based retrieval records in the northern mid-latitudes is narrowing. In 2018, the annual zonal means of the SPARE-ICE MetOp-A based retrieval and the NOAA-18 based retrieval almost completely overlap. In the Southern Hemisphere, the annual zonal mean of IWP based on NOAA-18 and MetOp-A overlap almost completely during the whole period between 2007 and 2018. The mismatch of the MetOp-A and NOAA-18 based retrieval records in the southern equatorial region in 2007 can possibly be attributed to the circumstance that the operational phase of MetOp-A only started in May 2007, whereas NOAA-18 covers the whole year. As a result, much of the southern displacement of the Intertropical Convergence Zone (ITCZ) is not represented in the MetOp-A dataset. The absence of these strongly convective regions in the southern equatorial region in the MetOp-A based data record could lead to reduced averaged zonal IWP values compared to the NOAA-18 based data record.

One possible factor that may explain the observation that the change in ECTs is much better observable in the Northern Hemisphere than in the Southern Hemisphere is that the Northern Hemisphere has a larger proportion of land area than the Southern Hemisphere. The Southern Hemisphere contains about 19 % land, while the Northern Hemisphere contains about 39 % land (Eldredge, 2002).

During the descending nodes (Figure 6.2), the NOAA-18 based IWP retrieval record slightly in the Southern Hemisphere, but overlaps with the MetOp-A based IWP retrieval record for almost the entire period in the Northern Hemisphere. The observed variations in the NOAA-18 based retrieval record in the Southern Hemisphere during the ascending nodes are much less pronounced than the variations observed in the Northern Hemisphere during the ascending node.

In the following paragraphs, I briefly consider whether the different diurnal cycle of thunderstorms over land and ocean can explain the differences in IWP variation between the Northern and Southern Hemispheres during the ascending and descending nodes.

Correlation between zonal mean variations and thunderstorm frequency

There is a more pronounced cycle of thunderstorms over land than over ocean (Zipser, Cecil, Liu, Nesbitt & Yorty, 2006; Galanaki, Lagouvardos, Kotroni, Flaounas & Argiriou, 2018; Dai, 2001). High IWP values occur especially in thunderstorms (Braga & Vila, 2014). Global annual mean IWP estimates based on CloudSat und CALIPSO are known to be driven by high IWP cases (Duncan & Eriksson, 2018). CloudSat and CALIPSO influence 2C-ICE, and thus impact the SPARE-ICE training results and also the results of the SPARE-ICE retrieval. Therefore, it is expected that global means, and possible also annual zonal means, of SPARE-ICE are driven by high IWP values cases like thunderstorms.

Since thunderstorms have a more pronounced diurnal cycle over land than over ocean, a stronger diurnal cycle of IWP over land than over ocean is expected. Due to the unequal distribution of land masses between the two hemispheres, this would result in a more pronounced cycle of IWP for the Northern Hemisphere. Accordingly, it would be expected that with a change in ECTs, the zonal means of IWP in the Northern Hemisphere vary more than in the Southern Hemisphere. This argumentation would offer one explanation for the observation that the change in ECTs from NOAA-18 's orbit does not have a clear impact on the IWP zonal mean in the Southern Hemisphere.

On the other hand, according to the diurnal cycle of thunderstorms, a maximum of IWP in the late afternoon between 4 pm and 6 pm would be expected over land (Zipser et al., 2006; Dai, 2001). This would correspond to a maximum in IWP (ascending node) in the NOAA-18 based retrieval between 2014 and 2016 in the Northern Hemisphere. This is not observed in the annually averaged zonal means. Instead, the magnitude of zonal mean IWP remains largely the same over these time periods. What would be consistent, however, is that IWP decreases towards the evening hours.

During its descending nodes, NOAA-18 covers night times into the morning hours. The frequency of thunderstorms over the ocean shows a short increase over night, but decreases again towards the evening hours (Zipser et al., 2006; Dai, 2001). Over land, the frequency of thunderstorms decreases over night (Zipser et al., 2006; Dai, 2001). Therefore, one would expect decreasing IWP values over the night over land

in the Northern Hemisphere. This expectation can be verified using Figure 6.2.

Considering the only partial correspondence of the dynamics of the zonal IWP means with the diurnal cycle of thunderstorm frequency between 2007 and 2019, this simple comparison does not reveal a clear correlation between both variables. In order to explain the variations in the IWP annual zonal means, it is recommended to also consider other factors, e.g diurnal cycles of shallow and deep convection.

Diurnal IWP cycle

To obtain the diurnal cycle based on the ascending and descending nodes, different zonal bands of annual mean IWP are plotted against the corresponding annually averaged local ECTs of the respective satellites. The results are shown in Figure 6.3. The data used covers the period 2007 to 2019 and takes into account both clouds and clearsky pixels. The IWP means are based on the SPARE-ICE NOAA-18 and MetOp-A retrieval data records and 2C-ICE respectively. Due to the almost unchanged orbit of MetOp-A and CloudSat over the entire retrieval period, the annually averaged ECTs of both satellites remain almost unchanged.

Two zonal bands were selected to represent the latitudes in the Northern and Southern Hemispheres where distinct inter-annual variations of annual zonal means were observed in Figure 6.1 and in Figure 6.2. The southern zonal band spans from 55°S to 50°S, the northern zonal band spans from 50°N to 55°N. The southern band covers almost entirely only ocean areas. The northern band contains both ocean and larger land regions. The equator is represented by a zonal band that spans from 5°S to 5°N. In addition, a land region and an ocean region were plotted as a reference for the diurnal cycle over land and over ocean. A 'box' between 35°N - 50°N and 130°W - 20°E (*Eurasia*) was chosen to represent the land region. A box between 60°S - 60°N and 30°W - 15°W was chosen to represent the ocean (*Atlantic*).

Since the main purpose of this subsection was to show how the diurnal cycle of IWP can be estimated by using the different IWP retrieval records, the evaluation below is mainly limited to a short description of the results of the diurnal cycle.

According to Figure 6.3, the annual averaged IWP above the ocean (cf. Atlantic) is higher than above land (cf. Eurasia) over the available time periods in general. The southern band shows a similar diurnal cycle as the selected Atlantic region. The magnitude of IWP of the southern band is similar to the magnitude of IWP from the Atlantic region. The northern band shows IWP values that are between those of Atlantic and Eurasia. According to the mixed land-sea fraction this would be in line with expectations, assuming that the land-ocean distribution has a strong influence on the diurnal cycle.

IWP during ascending and descending nodes (annual averages)
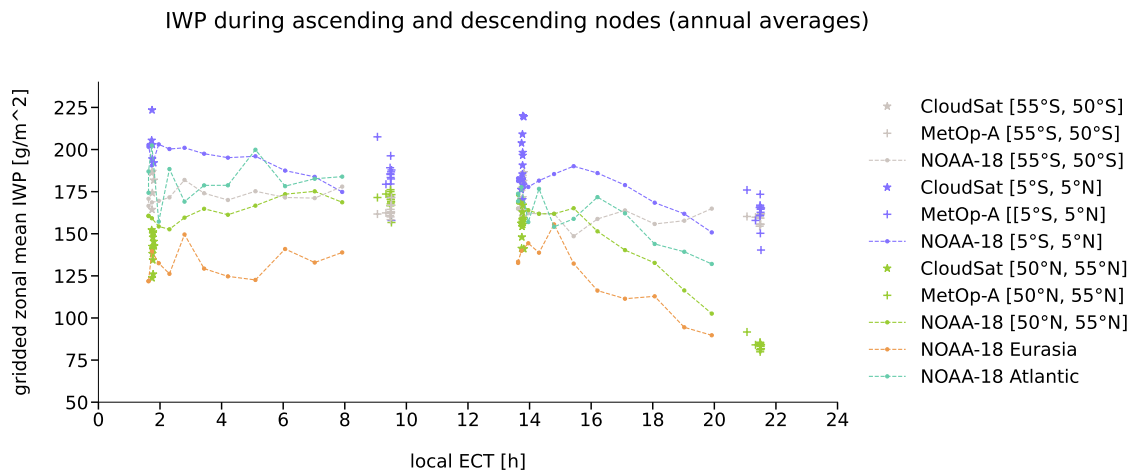
Figure 6.3: Diurnal cycle of IWP of different regions based on the MetOp-A (2007-2019) and NOAA-18 (2007-2018) SPARE-ICE IWP data records and 2C-ICE IWP (2007-2019). The ECTs of CloudSat are used for 2C-ICE. The Atlantic and Eurasia were presented using only NOAA-18 data as a reference.

Due to missing times, the maximum of the diurnal IWP variations can not be determined. There are no strong variations of IWP during night hours in all regions except at the equator. According to Figure 6.3, IWP decreases in the afternoon until the evening. Based on the available times, no pronounced cyclic IWP variation can be identified in any of the regions. However, with the exception of the southern band, there is a tendency towards lower values in the afternoon in the regions considered.

Given the large increase in IWP required between the evening and night hours to form a closed cycle in some of the regions, the results seem questionable. However, the IWP values on the basis of NOAA-18 are in the IWP ranges that are indicated by 2C-ICE (ECTs from CloudSat) and by the MetOp-A based retrieval record. The IWP spread shown by the MetOp-A based retrieval product and 2C-ICE are indicative of the intra-annual variability of IWP.

It should be noted that the results shown in this study are averaged over large regions. As a result, the diurnal cycle may have been obscured by the overlap of several regional phenomena. The fact that SPARE-ICE was trained with afternoon/night values alone could also have an effect.

Furthermore, this analysis was based on allsky values. This means that different diurnal cycles are mixed with each other. For example, the diurnal cylce of precipitating ice plays is also expected to influence the diurnal cycle of IWP. Seasonal effects are included due to the large temporal averaging window.

For potentially more promising results, it is recommended to apply the ice cloud

flags of the retrieval records and test different temporal averaging windows.

When comparing the results from this study with other studies, it should be kept in mind that IWP retrievals in general hardly match when it comes to the diurnal cycle of IWP and also differ strongly from each other in their zonal means (Duncan & Eriksson, 2018). This phenomenon has been attributed mainly to differences in microphysics (Duncan & Eriksson, 2018). Even though microphysical assumptions were not explicitly present in SPARE-ICE, they may have an impact on SPARE-ICE through the use of 2C-ICE as a reference.

6.2.2   Global IWP distribution in 2008: ascending node

To better assess the global distribution of IWP values of the newly created SPARE-ICE IWP data records, the following paragraphs present a global map of averaged IWP for each of the new data records for the ascending nodes during 2008 respectively. To estimate the accuracy of the new retrieval records, I also present the their relative differences with respect to 2C-ICE IWP. For this purpose, 2C-ICE was not collocated with the SPARE-ICE retrieval products. The ascending nodes were calculated based on NOAA-18, MetOp-A, and CloudSat.

The IWP values were annually averaged over 5° x 5° grid cells for each case. As the aim of this subsection is to provide a brief introduction to the new IWP data records, this subsection is limited to a brief description of the records.

For future users of the SPARE-ICE products, additional global maps are available as reference in Appendix E.2. The maps show global annual mean values of the SPARE-ICE products and 2C-ICE IWP averages for the years 2008 and 2017. A distinction is made between ascending and descending nodes. Relative differenes between the SPARE-ICE based IWP data records and 2C-ICE are also shown. Since CloudSat operated in DO-OP mode only in 2017, only the ascending nodes of 2C-ICE and the SPARE-ICE products are presented for 2017 in the Appendix.

Figure 6.4a shows the global distribution of atmospheric ice from the NOAA-18 based SPARE-ICE IWP data record. Spatial structures governed by the atmospheric general circulation are well represented. The highest IWP values can be found in the ITCZ and in the Pacific warm pool region. In addition, high values are particularly common in the storm track regions of both hemispheres and on the western edges of the subtropical gyres. Low IWP values prevail in the subtropical high pressure belts. Over land, especially the Sahara and Arabian deserts show a low amount of atmospheric ice within the subtropical high pressure belt.

SPARE-ICE (NOAA-18) IWP 2008
Ascending node

(a)

SPARE-ICE (NOAA-18) - 2C-ICE IWP 2008
Ascending node

(b)
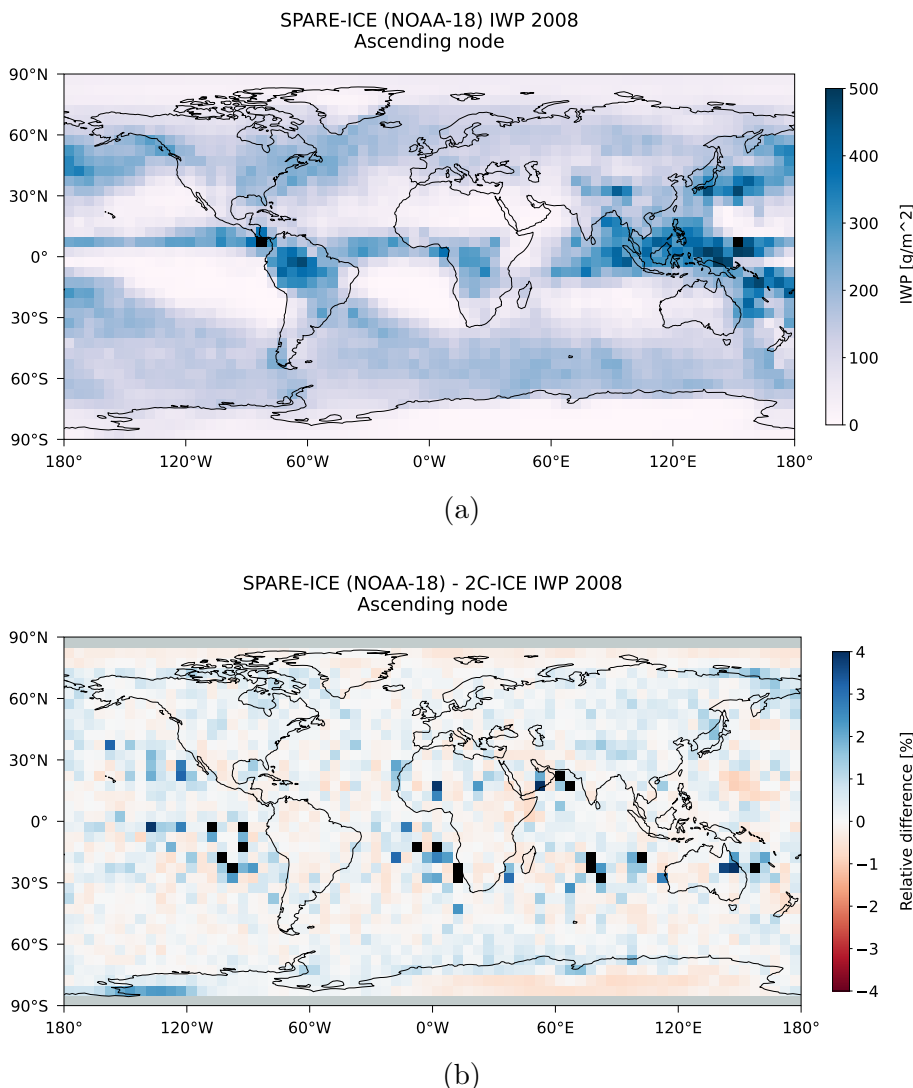
Figure 6.4: Global maps of (a) annually averaged IWP from the NOAA-18 based SPARE-ICE retrieval, and (b) the relative difference between annually mean IWP values of the NOAA-18 based SPARE-ICE IWP retrieval product and 2C-ICE IWP for 2008 during the ascending nodes of NOAA-18 and CloudSat (as representative for 2C-ICE). The data is averaged over 5° x 5° grid cells. 2C-ICE was quality checked and cleared of MODPs via the pre-processing toolkit before calculating the relative difference. Areas where no values from 2C-ICE were available are indicated in grey.

Due to the rough averaging, the representation of local features is difficult to assess, as they are not clearly visible. However, there is a slight tendency towards higher values in regions of orographic uplift, for example on the edge of the Himalayas, at the coast of Alaska and at the Alps. This indicates a rather good representation of local IWP features (c.f. Duncan and Eriksson (2018)). Overall, the NOAA-18 based IWP data record shows reasonable results during the ascending nodes of NOAA-18 for 2008. The global mean of IWP during the ascending nodes in 2008 is 120 g/m^2.

(a)

(b)
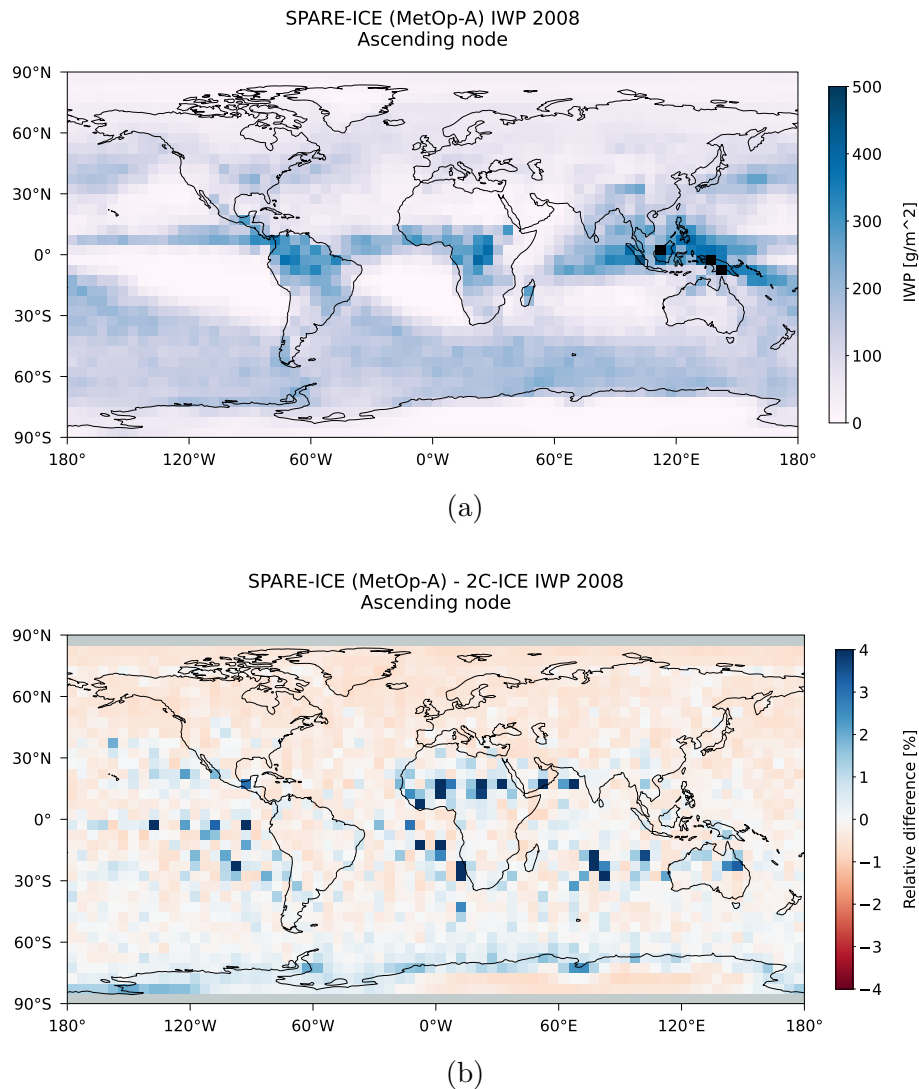
Figure 6.5: As in Figure 6.4, but for the MetOp-A based SPARE-ICE IWP data record.

In Figure 6.4b the relative differences between the NOAA-18 based IWP data record and 2C-ICE IWP are shown. Grey areas indicate where no comparisons were possible. CPR and CALIOP do not observe regions beyond 82° North and South.

In the Northern Hemisphere, the SPARE-ICE record seems to slightly overestimate IWP compared to 2C-ICE. Over the southern continents, the SPARE-ICE record slightly overestimates IWP compared to 2C-ICE.

The largest differences ($> 4 \%$) exist in the subtropical high pressure belts. Very low IWP values prevail in these regions. The differences may be due to different sensitivity limits of 2C-ICE and SPARE-ICE. According to the retraining, the sensitivity limit of SPARE-ICE is about 10 g/m^2. The sensitivity limit of CloudSat-CALIPSO products is estimated to be roughly 1 g/m^2 (Duncan & Eriksson, 2018; Holl et al., 2014).

Figure 6.5a shows the same analysis as Figure 6.4a, but for the MetOp-A based SPARE-ICE IWP data record. The same spatial structures as for the NOAA-18 based IWP data record can be observed. However, these are not as pronounced in the Northern Hemisphere and in the region of the ITCZ. MetOp-A samples the early evening hours during its ascending nodes in 2008, whereas NOAA-18 samples the early afternoon hours. Especially in the tropics, differences between both data records might originate from differences in the strength of convection between early afternoon and early evening. The global mean of IWP during the ascending nodes in 2008 is 95 g/m^2.

There are pronounced relative differences to 2C-ICE (Figure 6.5b), compared to the NOAA-18 based data record (Figure 6.4b). The SPARE-ICE based product shows a negative bias in the Northern Hemisphere, and a positive bias in the lower latitudes of the Southern Hemisphere. The differences can probably also be attributed to different sampling times. CloudSat and CALIPSO sample the early afternoon hours during their ascending nodes. The largest relative differences (> 4 %) between the MetOp-A based IWP data record and 2C-ICE IWP are scattered throughout the subtropical high pressure belts.

Chapter 7

Summary and outlook

During this study, two new IWP data records were generated using SPARE-ICE. The first one is based on MHS and AVHRR/3 data from NOAA-18 and covers the period from January 2007 to October 2018, the second one is based on MHS and AVHRR/3 data from MetOp-A and covers the period from May 2007 and December 2019. A new pre-processing toolkit was created and added to SPARE-ICE. All unresolved issues found in Mrziglod (2018)'s SPARE-ICE implementation were resolved. The code was optimised to be more user-friendly, and implementation descriptions were added to make the code more developer-friendly.

This study builds on the SPARE-ICE algorithm and baseline experiment from Mrziglod (2018). For the generation of the new SPARE-ICE IWP data records, the IWP regressor and ice cloud detector of SPARE-ICE were re-trained without AVHRR channel 3. This was necessary as AVHRR channel 3 had frequency changes within the analysed time periods. For the re-training, MHS and AVHRR data from NOAA-18 from January 2007 to December 2010 were used. The training dataset was balanced with respect to the number of cloud and clearsky data points. As target during the re-training, IWP data from 2C-ICE was used. The re-trained IWP regressor achieved a final R^2 score of 0.839 on the testing dataset and shows a lower sensitivity limit of IWP of 10 g/m^2. The re-trained ice cloud detector achieved a F1 score of 0.889 on the testing dataset. The results of the re-training suggest that the exclusion of AVHRR channel 3 and the choice of this study to train a single decision tree classifier as ice cloud detector did not significantly affect the performance of SPARE-ICE compared to the original baseline experiment by Mrziglod (2018).

The new IWP data records represent SPARE-ICE IWP records of so-far unprecedented length and help to meet the need for more accurate global IWP data records.

The long temporal coverage of the newly created data records makes SPARE-ICE attractive for studies that focus on the diurnal, seasonal, intra-seasonal, and intra-annual temporal and geographical variability of IWP and makes the data records interesting for the evaluation of models. By applying SPARE-ICE to data from MetOp-A, the retrieval is applied for the first time to a satellite beyond NOAA-18.

The two new SPARE-ICE IWP data records were presented by means of an application example that derived the diurnal cylce of IWP based on the new data records and ETC changes of NOAA-18 and MetOp-A. Zonal means of annually averaged IWP were also presented. The spatial IWP distribution was presented using global maps for the ascending nodes of NOAA-18 and MetOp-A in 2008 respectively.

The derived diurnal cycle of IWP seems questionable. In particular, the results do not indicate that there is a smooth transition of the IWP values during the night. However, this could be due to the fact that the displayed diurnal cycle might be distorted, for example by the inclusion of various regional effects.

According to their annual zonal means, the two new SPARE-ICE IWP data records align well witch each other between 2007 and 2018, both during the ascending and descending nodes of the respective satellites on which the retrieval data records are based. Both IWP data records reproduce the zonal distribution of IWP from 2C-ICE quite well from 2007 onward during the ascending nodes. Deviations from 2C-ICE IWP and differences between the newly generated SPARE-ICE data records correspond very well with differences in ECTs of the orbits of the various satellites on which the retrieval data records are based on.

In order to ensure high quality of the sensor-based input data, a new pre-processing toolkit was created and added to SPARE-ICE. The new pre-processing toolkit can be used independently of SPARE-ICE. The toolkit deals with the application of dataset inherent quality indicators from MHS, AVHRR, and 2C-ICE and handles the removal of multiple occurrences of data points. Although the application of the quality indicators works very well, the quality indicators do not flag all apparent abnormalities in the datasets. For instance, even after applying all available AVHRR quality indicators to datasets from AVHRR from NOAA-18, brightness temperatures of about -323 K were still present in the datasets.

To improve the SPARE-ICE algorithm, all unresolved issues found in Mrziglod (2018)'s SPARE-ICE implementation were addressed and corrected. These included among others, inconsistencies between similar methods, difficulties in handling certain file anomalies, the creation of artificial duplicates, missed collocations due to unsorted data, and non-executable scripts.

To make SPARE-ICE more user- and developer-friendly, shell scripts, jupyter note-books, detailed implementation and process descriptions, as well as workflow illustrations were added. The new SPARE-ICE version will be available via the typhon package.

The two new SPARE-ICE IWP data records still need to be evaluated in more detail. Of particular interest would be a comparison with other IWP data records such as those from PATMOS-X, and MODIS over several years, as both products are known to be in closest agreement with CloudSat based products in terms of IWP magnitude and spatial distribution (Eliasson et al., 2011).

A further extension of the SPARE-ICE based IWP product range beyond 2019 based onMetOp-A, MetOp-B, and MetOp-C is recommended. It could also be promising to generate a single IWP time series that is purely based on combined sensors measurements from MetOp-A, MetOp-B, and MetOp-C. In addition to the advantage that all three satellites are equipped with MHS and AVHRR/3, the three satellites were placed on the same orbit. This facilitates the coupling of retrieval data records based on these satellites and opens up the possibility of generating a single SPARE-ICE IWP data record extending from May 2007 until after 2023.

Appendix A

User guide

The SPARE-ICE package is available via typhon on GitHub.

Typhon package:
`https://github.com/atmtools/typhon`

Typhon documentation:
`https://www.radiativetransfer.org/misc/typhon/doc/index.html`

SPARE-ICE

The SPARE-ICE package offers various processing steps:

> (1) Collocating (collocations): generation of pre-processed collocations for the training of SPARE-ICE and for the retrieval of the IWP values with SPARE-ICE
> (2) Training (training): homogenisation and balancing of training collocations and training of SPARE-ICE
> (3) IWP retrieval (retrieval): retrieval of IWP values and cloud flags
> (4) Post-processing (postprocessing): separation of retrieval datasests into ascending and descending nodes and creation of gridded means
> (5) visualisation (visualisation): plot of IWP zonal means, global maps, ECT, etc.

The steps build on each other in the numbered sequence. The folders are indicated in round brackets.

Python scripts (.py), shell scripts (.sh) and jupyter notebooks (.ipynb) are available for processing steps (1)-(4). The jupyter notebooks can be used for testing. Only the python and shell scripts are required for executing the processing steps. For the

execution of the python scripts, the shell scripts have to be used. To execute the python files without the shell scripts, the parameters specified in the python files need to be manually replaced. Please note that the terms 'CloudSat' and '2C-ICE' are used interchangeably in the scripts, as the name 2C-ICE is not always practical due to naming conventions.

Input data and format:

MHS	HDF5
AVHRR	HDF5
2C-ICE (R05)	HDF4

Additional inputs:

Elevation file	ETOPO1_Bed_g_gmt4_1m_rn.grd
Land-sea mask file	land_water_mask_5min.png

> **Example: Training of SPARE-ICE and Generation of new SPARE-ICE IWP datasets**
>
> 1. Download typhon from GitHub (includes SPARE-ICE package)
> 2. Set up computational environment according to Section 5.1
> 3. Set parameters in *setup.py*
> 4. Select quality flags via *qualinds.py* if the default quality settings are not taken
> 5. For training:
>   (a) Generate pre-processed MHS_2C-ICE_AVHRR collocations via *Create_pptraining_collocations.sh* for the desired time frame
>   (b) Create training dataset via *prepare_spareice_training_dataset.sh* for the desired time frame
>   (c) Train SPARE-ICE via *train_spareice.sh*
> 6. For retrieval (generate IWP datasets):
>   (a) Create pre-processes MHS_AVHRR collocations via *create_ppretrieval_collocations.sh* for the desired time frame
>   (b) Generate IWP datasets via *run_spareice.sh* for the desired time frame

All parameters that can be set for the individual processing steps are listed and described in the file *setup.py*.

Processing step	Python functions
collocations	create_pptraining_collocations.py
	create_ppretrieval_collocations.py
training	prepare_spareice_training_dataset.py
	train_spareice.py
retrieval	run_spareice.py
post-processing	create_spareice_grdmeans.py
	create_2cice_grdmeans.py
	create_ascending_descending_nodes_spareice.py
	create_ascending_descending_nodes_2cice.py

Table A.1: Python functions available for the different processing steps of SPARE-ICE.

The available functions and their default output folders are listed in Table A.1 and Table A.2.

If the paths are not explicitly specified, the inputs and results are searched in subfolders starting from the top hierarchy of the SPARE-ICE package folder. The names of the subfolders are set by default in the respective scripts. Table A.2 lists the default output folders.

Selection of default directory path	$< path >= ""$
Subfolders	$< DefaultPath > /results$ $/< subdirectoryName >$

Pre-processing toolkit

The pre-processing toolkit provides pre-processing functions that can also be used independently of SPARE-ICE. The pre-processing functions deal the application of quality indicators and the removal of multiple occurrences of data points (referred to as 'functions for the removal of duplicates' in the scripts).

The pre-processing functions are compiled in the file *preprocessing_rawdata.py*. Among other things, the file contains wrapper functions that combine different pre-processing steps or enable the parallel execution of the functions. These wrapper functions are listed in the following. The call of the wrapper is exemplified in the file *tutorial_preprocessing.ipynb*.

Input: xarray.Datasets

Removal of multiple occurrences of data points

Applicable to data from:

AVHRR/3	Level 1c
MHS	Level 1c
2C-ICE P1_R04	Level 2c
2C-ICE P1_R05	Level 2c

The functions have not yet been applied to datasets from other sensors.

> **Recommended process for removing multiple occurrences of data points from datasets:**
> (1) Load dataset(s) as xarray.Dataset(s) / Define FileSets
> (2) Apply wrapper functions

Information about FileSets can be found in the official typhon documentation.

Available wrapper functions:

- *get_nodupl_data*
  removes multiple occurrences of data points according to latitude, longitude, and time from xarray.Dataset

- *get_nodupl_data_parallelised*
  applies *get_nodupl_data* to a list of xarray.Datasets in parallel

- *get_nodupl_data_mfilesets*
  combines all files on a search path within a certain time slot and removes multiple occurrences of data points according to latitude, longitude and time by calling *get_data_within_timeframe* and *get_nodupl_data*

- *get_nodupl_data_mfilesets_parallelised*
  applies *get_nodupl_data_mfilesets* to all files within a certain time slot in parallel. The time slot is split before being passed into *get_qchecked_data_mfilesets*.

Application of quality indicators

Applicable to data from:

AVHRR/3	Level 1c
MHS	Level 1c
2C-ICE P1_R04	Level 2c
2C-ICE P1_R05	Level 2c

The functions have not yet been applied to datasets from other sensors.

> **Recommended process for applying quality indicators to datasets:**
> (1) Select quality indicators and quality flags in *qualinds.py*
> (2) Load dataset(s) as xarray.Dataset(s) / Define FileSets
> (3) Apply wrapper functions

Information about FileSets can be found in the official typhon documentation.

Available wrapper functions:

- *get_qchecked_data*
  applies quality indicator(s) to xarray.Dataset

- *get_qchecked_data_parallelised*
  applies *get_qchecked_data* to a list of xarray.Datasets in parallel

- *get_qchecked_data_mfilesets*
  combines all files on a search path within a certain time slot and applies quality
  indicators by calling *get_data_within_timeframe* and *get_qchecked_data*

- *get_qchecked_data_mfilesets_parallelised*
  applies *get_qchecked_data_mfilesets* to all files within a certain time slot in par-
  allel. The time slot is split before being passed into *get_qchecked_data_mfilesets*.

Combinations: Removal of multiple occurrences of data points and application of quality indicators

The pre-processing toolkit also provides wrapper functions to apply quality indicators and remove multiple occurrences of data points directly after each other.

Available wrapper functions:

- *get_qchecked_nodupl_data_mfilesets*
  combines all files on a search path within a time slot, applies the quality indicators to them, and removes multiple occurrences of data points by calling *get_data_within_timeframe*, *get_qchecked_data*, and *get_nodupl_data* respectively

- *get_qchecked_nodupl_data_mfilesets_times*
  does the same as *get_qchecked_nodupl_data_mfilesets*, but outputs the times required to execute *get_data_within_timeframe*, *get_qchecked_data*, and *get_nodupl_data* respectively

- *get_qchecked_nodupl_data_mfilesets_parallelised*
  applies *get_qchecked_nodupl_data_mfilesets* to all files within a certain time slot in parallel. The time slot is split before being passed into *get_qchecked_nodupl_data_mfilesets*

Additional wrappers

To simplify the handling of the files and to prepare the input for the pre-processing wrapper functions, additional wrapper functions are available.

- *get_data_within_timeframe*
  collects all files that are on the search path and combines them into a single xarray.Dataset

- *get_data_within_timeframe_parallelised*
  calls *get_data_within_timeframe* in parallel for different lists of file paths and time slots

Integration into SPARE-ICE

The functions of the pre-processing toolkit are connected with SPARE-ICE via wrapper functions that can be found in *preprocessing_coll_wrappers.py*. The functions apply *get_qchecked_nodupl_data_mfilesets* on the MHS, AHVRR, and 2C-ICE input files

and then pass the pre-processed input datasets directly into the collocator in order to create pre-processed collocations. If desired, the pre-processed MHS, AVHRR, and 2C-ICE files can be stored by setting the parameter *store_preprocessed_ data* to true. As default, only the pre-processed collocations are stored.

Available wrapper functions:

- *get_trainingcollocations_from_qflagged_nodupl_data*
  creates pre-processed collocations between MHS, AVHRR, and 2C-ICE files

- *get_retrievalcollocations_from_qflagged_nodupl_data*
  creates pre-processed collocations between MHS and AVHRR files

Table A.2: Python functions required for the different processing steps of SPARE-ICE with their respective ouput files and default output folders.

Python functions	Output	File names	Default output subfolderName
create_pptraining_collocations.py	Pre-processed MHS_2C-ICE collocations	pp_ $< start\_hour >$ $< start\_minute >$ $< start\_second >$ $\_ < end\_hour > < end\_minute >$ $< end\_second >$ .nc	training_ppcollocations $/MHS\_2C - ICE/ < satellite >$ $/ < year > / < month >$ $/ < day > /$
	Pre-processed MHS_2C-ICE_AVHRR collocations; default: collapsed onto MHS	pp_ $< start\_hour >$ $< start\_minute >$ $< start\_second >$ $\_ < end\_hour > < end\_minute >$ $< end\_second >$ .nc	training_ppcollocations $/MHS\_2C - ICE\_AVHRR/$ $< satellite > / < year >$ $/ < month > / < day > /$
create_ppretrieval_collocations.py	Pre-processed MHS_AVHRR collocations	pp $\_ < start\_hour >$ $< start\_minute >$ $< start\_second >$ $\_ < end\_hour > < end\_minute >$ $< end\_second >$ .nc	retrieval_ppcollocations $/MHS\_AVHRR/ < satellite >$ $/ < year > / < month >$ $/ < day > /$
			Continued on next page

Table A.2 – continued from previous page

Python functions	Output	File names	Default output subfolderName
prepare_spareice _training_dataset.py	Optional: concatenated pre-processed training dataset	spareice_training_data $\_ < start\_time > \_ < end\_time >$ .nc	training
	Concatenated pre-processed, homogenised, and balanced dataset (expected format: .nc)	$< filename\_balanced$ $\_homogenised\_data >$	training
train_spareice.py	Optimised parameters of the trained IWP regressor and ice cloud detector	$< experiment >$ _spareice.json	training
	Test score of the IWP regressor and ice cloud detector and hyperparameter combinations tested during training for the IWP regressor	$< experiment >$ _BestEstimator.txt	training
Continued on next page			

Table A.2 – continued from previous page

Python functions	Output	File names	Default output subfolderName
run_spareice.py	Retrieved IWP datasets containing IWP values and ice cloud flags	$< year > < month > < day >$ $\_ < start\_hour >$ $< start\_minute >$ $< start\_second >$ $\_ < end\_hour > < end\_minute >$ $< end\_second >$ .nc	retrieval_iwp
create_spareice _grdmeans.py	Gridded SPARE-ICE IWP datasets	spareice_gridded_ $< year >$ $\_ < allsky/clouds >$ .nc	gridded_spareice / $< year >$ /
create_2cice _grdmeans.py	Gridded 2C-ICE IWP datasets (create pre-processed 2C-ICE IWP datasets first if desired)	2cice_gridded $\_ < processing\_level >$ $\_ < 2ciceVersion > \_ < year >$ $\_ < allsky/clouds >$ .nc	gridded_2cice / $< year >$ /
create_ascending _descending_nodes _spareice.py	Gridded SPARE-ICE IWP datasets for the ascending and descending nodes	spareice_gridded_ $< year >$ $\_ < allsky/clouds >$ $\_ < ascending/descending >$ .nc	gridded_spareice / $< year >$ /

Table A.2 – continued from previous page

Python functions	Output	File names	Default output subfolderName
create_ascending _descending_nodes _2cice.py	Gridded 2C-ICE IWP datasets for the ascending and descending nodes (create pre-processed 2C-ICE IWP datasets first if desired)	2cice_gridded $_- < processing\_level >$ $_- < 2ciceVersion > _- < year >$ $/ < allsky/clouds >$ $/ < ascending/descending >$ .nc	gridded_2cice $/ < year > /$

Appendix B

Component descriptions and implementation details

This chapter describes the collocator, IWP regressor, and ice cloud detector components of SPARE-ICE. First, there is a summary of the individual components, followed by a description of the respective implementations. As an example of the general structure of the SPARE-ICE code, the implementation of the collocation component will be described more thoroughly. Thereby this study strives to help further developers and users to gain a more profound understanding of the underlying algorithm of SPARE-ICE and thus to facilitate further changes.

B.1   Collocation method

The collocator component finds collocations between different sensor measurements based on the latitude, longitude and time stamps of the respective measurements.

The following paragraphs describe the implementation of the collocation component in SPARE-ICE more thoroughly. In this context, the method used for finding potential collocations according to the spatial collocation criterion is explained in more detail.

Implementation details

In the current version of SPARE-ICE, which builds on the SPARE-ICE version of Mrziglod (2018), the collocator component is structured according to object-oriented programming principles and relies on the classes `FileHandler`, `FileSet`, and `Collocator`. The `FileHandler` class provides reading and writing routines for different file formats like for AVHRR HDF files or MHS HDF files. The `FileSet`

class is mainly used to create `FileSet` objects that contain the `FileSet` object name, file path, and a handler from `FileHandler`. The paths to the files can be given as regular expressions which allows to find multiple datasets that fulfil a certain search criterion within a directory and between different sub-directories. The `Collocator` class allows to collocate `FileSet` objects and xarray.Datasets objects.

The basic workflow of the collocator component for collocating two `FileSet` objects, called *primary* (e.g. `FileSet` object for MHS data) and *secondary* (e.g. `FileSet` object for AVHRR data), within a time period defined by *start* (start date) and *end* (end date) under consideration of a time threshold *max_interval* (Δt) and a
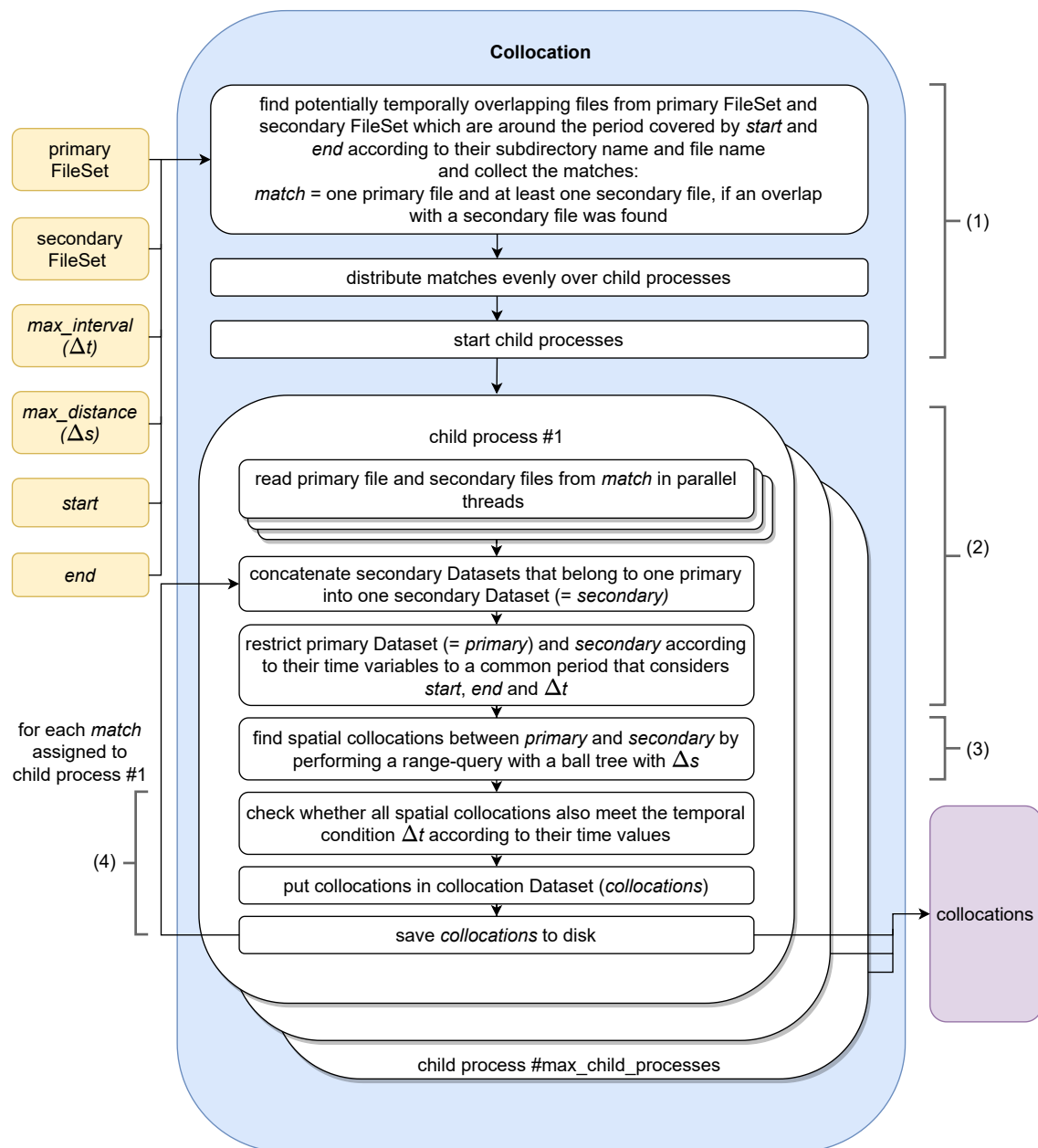
Figure B.1: Simplified depiction of the collocation component workflow of SPARE-ICE with its main inputs and outputs.

distance restriction of *max_distance* (Δs), is shown in Figure B.1.

In the following, the main steps that take place within the collocator are explained by using Figure B.1. The numbers referenced in the text refer to the individual segments of the collocator noted the Figure. A more detailed description of the collocation procedure, including names of the used function, can be found in Appendix C.1.

In segment (1), the collocator component searches, according to the basic path structure given in the path attribute of the `FileSet` object, through all directories that might contain files that could potentially overlap with the time period covered by start and end. At this point, the first temporal restriction is made. The search is only performed on the information of the subdirectory and file names. During its search, the collocator component collects file information (i.a. file paths), called *FileInfos*, for all files from primary and secondary that match its temporal search criterion and create so-called *matches*. matches consists of several tuples where one tuple represents a *match*. A match contains the FileInfos of one file from primary and all FileInfos from all files from secondary which overlap with the primary file according to their FileInfos. This implies that every match contains the FileInfos of a different primary file. The matches are then evenly distributed on different child processes. Each child process gets at least one match for processing. The user can set the maximum number of child processes that will be initiated. If the given maximum number of child processes exceeds the number of matches, only as many child processes as matches are initiated. If the given maximum number of child processes is less than the number of matches to be distributed, several matches are assigned to one child process.

Segment (2) to (4) are executed by each child process respectively. In segment (2), each child process reads the files from its match(es) in parallel threads via the *read* method from the primary and secondary handlers. To make the reading process more efficient, secondary files that have to be read multiple times are cached in the background as long as they are still needed for another match that has been assigned to the same child process. Every time the secondary file is needed again, it is taken from the cache instead of being read again. This enhances performance as the same files do not have to be read in multiple times. If a secondary file is not longer needed for another primary file, it is removed from the cache. Each child process manages its own cache.

When all secondary files that were matched with one primary file have been read in as xarray.Datasets, they are concatenated and form one xarray.Dataset.
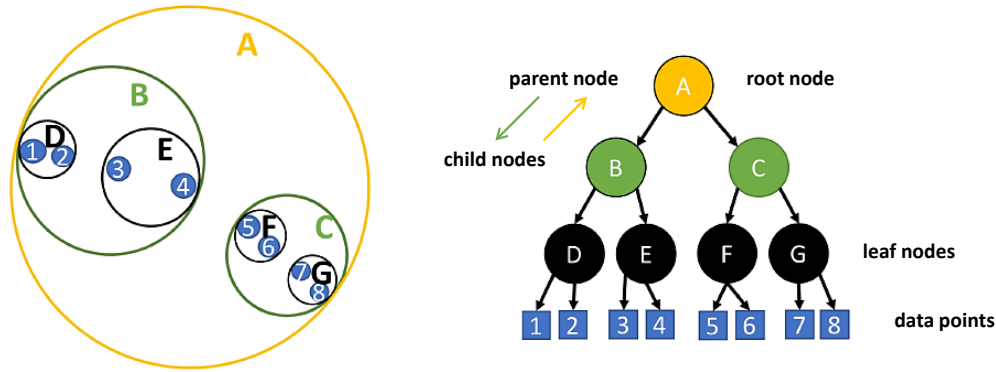
Figure B.2: Schematic sketch of a ball tree with depth 3. The sketch has been adopted from Mrziglod (2018, p. 9) (Figure 2.5) with minor changes.

From now on, if not stated otherwise, the term *secondary* is used to describe one matched xarray.Dataset that has been read in from the secondary `FileSet`. The corresponding xarray.Dataset that belongs to the matched file from primary `FileSet` is from now on called *primary*.

After concatenation of the matched secondaries, a common time period is defined based on the time values from primary, the concatenated secondaries, start, end, and Δt. Within this common time period, collocations are possible. Primary and the concatenated secondaries are then restricted to the common time period. This again imposes a time restriction, but now by taking into account actual times available in the datasets. Afterwards, undefined latitude, longitude, and time values are removed from primary and secondary and spatial collocations are searched.

The search for spatial collocations in segment (3) can be seen as a spatial range-query problem where one wants to get all points b from a set B whose distances to a point a from set A are less than a radius r (Dolatshah, Hadian & Minaei-Bidgoli, 2015; Mrziglod, 2018). To solve this problem, Mrziglod (2018) chose a ball tree algorithm, whose underlying mechanism is described in the following.

Ball trees are efficient space-partitioning data structures that are specific for spatial queries which use euclidean distances (Dolatshah et al., 2015). Their data structure is based on hierarchically related nodes where each node represents a set of points (T. Liu, Moore & Gray, 2006). Each branch of the tree splits its set of points into two sub-partitions, hence into two subsets.

Figure B.2 shows a schematic sketch of a constructed ball tree. The topmost node is called the root node. The nodes below are interior nodes which have a parent node and children nodes. All nodes without children are called leaf nodes.

The root node contains the full set of points in the data set (T. Liu et al., 2006). Each

interior node represents a spherical partition of the space ('ball') where each interior node's ball is the smallest ball which contains the balls of its children (Omohundro, 1989; Dolatshah et al., 2015). The interior nodes are used only to efficiently search through the leaf structures (Omohundro, 1989). Leaf nodes contain a list of the points represented by the node explicitly (T. Liu et al., 2006).

The SPARE-ICE collocation algorithm utilises a ball tree algorithm to significantly reduce the number of distance calculations needed for finding spatial collocations. The ball tree is build with the points from the larger dataset and queried with the points from the smaller dataset. When a leaf node is reached during the query, the distance to all points in the leaf node is calculated to check for collocations under consideration of Δs. As the leaf nodes contain only a very small subset of the overall data set, this approach strongly reduces the number of required distance calculations. If no leaf node is reached, there are no collocations. A more detailed description of the ball tree structure, and an example for a range-query with a ball tree can be found in Mrziglod (2018).

For the implementation, the BallTree algorithm from sklearn.neighbours is used as default with a minkowski metric of order 2, which results in the standard Euclidean distance[1] for distance measurement.

In segment (4), after the spatial collocation search, all found collocation pairs are checked on the basis of their temporal distances from each other to see whether they fulfil the temporal criterion Δt. This ensures that a maximum temporal distance of Δt is kept. The collocations, which now have been checked in terms of time and space, are then combined in a xarray.Dataset and saved to disk.

B.2   IWP regressor

The core of the IWP retrieval component of SPARE-ICE is a Multilayer Perceptron. This section first briefly introduces MLPs by explaining their general structure and training process. It then outlines the main advantages and disadvantages of MLPs and explains what attempts have been made during this study to compensate for their disadvantages. Finally, the integration of the MLP into SPARE-ICE is described in more detail.

---

[1]`https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.minkowski.html#scipy.spatial.distance.minkowski`, last visited: 22.06.2023, 16:35

Introduction to Mulilayer Perceptrons

An MLP represents an example of a fully connected feed-forward ANN (da Silva et al., 2017). It consists of one input layer, one output layer, and at least one hidden layer between the input and the output layer (Figure B.3). Each layer contains nodes, so-called *neurons*. For each input variable, called *feature*, one neuron in the input layer exists. The output layer contains as many neurons as desired output variables. Ideally, the number of hidden layers and number of neurons per hidden layer are chosen so that the network performs in an optimal way. They are architectural choices that depend on the specific application (Ramchoun, Ghanou, Ettaouil & Janati Idrissi, 2016).
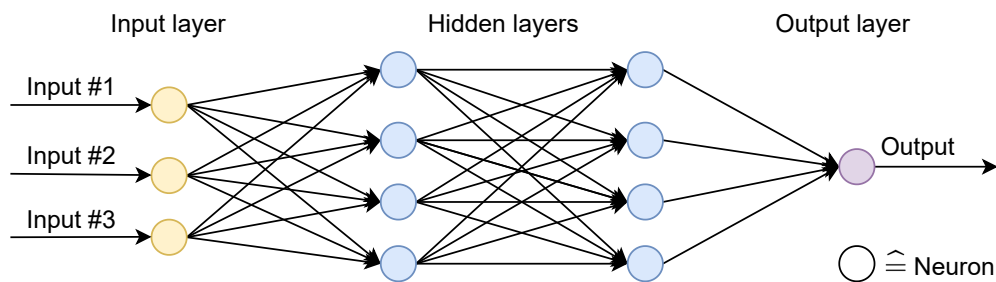
Figure B.3: Sketch of a simple ANN with two hidden layers.

A neuron from within a layer is linked to all neurons from the next layer via weighted connections (da Silva et al., 2017; Ramchoun et al., 2016; Sawaf & Groves, 2013). Every neuron from the hidden layer applies a non-linear activation function to the sum of its summed up weighted inputs from the previous layers and a bias, and then forwards its result to the neurons of the next layer. The node(s) in the output layer also connect the weighted inputs from the previous layer, but the choice of activation function in the output neuron(s) may differ from that used in the hidden neurons.

Before applying the MLP, the MLP has to be trained first. The training is performed with a supervised process were the weights and biases of each neuron are adjusted. For the supervised process, a *target* dataset is needed. The target data represents the desired output for the input training data (da Silva et al., 2017; Ramchoun et al., 2016). During the training, the network tries to adjust its parameters in such a way, that the difference between the network output and the desired output is minimised (Ramchoun et al., 2016). This 'learning' process during the training of is called *backpropagation* (da Silva et al., 2017) and is performed by a backpropagation algorithm. The first stage of the learning is called *forward propagation*. During this stage, the network forwards the information from the input layer through the hidden layers to the output layer. In the second learning stage (*backward stage*), the total

difference between the network output and the target data is calculated and used for adjusting the weights and biases of the nodes (da Silva et al., 2017). Those two stages are performed several times in succession during the learning process. After each training iteration, the error on a validation dataset is checked. If the error increases, the training stops. After training, the total error of the network is calculated on a testing dataset.

To get the training, validation, and testing datasets, the original dataset is split into three parts. For example, Holl et al. (2014) used 40 % of their dataset for training, 20 % for validation, and 40 % for testing. This study uses 60 % of the processed data points used for re-training for training, 15 % for validation, and 25 % for testing.

Advantages and disadvantages

On the one hand, neural networks, like MLPs, are very useful for identifying complex nonlinear relationships between the input and the target data, even when they have to deal with slightly imprecise data or when the exact relations between both variables still have to be discovered by the research society. As such, they can also derive meaning from complicated data. (Dumitru & Maria, 2013; Tu, 1996)

On the other hand, they also have several drawbacks: They are often regarded as a 'black box' since their rules of operation are not transparent (Dumitru & Maria, 2013). This makes their results hard to interpret. They also require large datasets to make accurate predictions, are sensitive to the scale of their input features and bad input data quality, and they are relatively computationally expensive (Dumitru & Maria, 2013; Tu, 1996; Rusiecki, Kordos, Kamiński & Greń, 2014; D. Chen & Jain, 1994). In addition, they are known to be susceptible to overfitting which means they may perform well on their training dataset, but fail on data outside of their training dataset. Hence, they lack the ability to generalise (Dumitru & Maria, 2013). Adding terms that constrain the model weights (*regularisation terms*) can help to prevent overfitting (Santos & Papa, 2022).

MLPs are also not well suited for extrapolation. This implies that that they should be applied only on value ranges they have already seen during the training (Courtois, Morel & Arias, 2023; Haley & Soloway, 1992). However, modifications, for example adding regularisation terms, can improve their extrapolation performances (Davini et al., 2021).

Another disadvantage regarding the training is that training results are usually hard to reproduce. That means different trainings can result in different optimal parameter sets and total errors. Randomness is introduced into the neural network

during the initialisation of its weights and biases, the selection of points for the testing, training, and validation dataset for training and when shuffling the training data during the learning phase. To handle those problems, for example, a seed can be fixed for the random selection of data points and for the random initialisation of the weights and biases, and shuffling can be switched off. However, this may degrade the overall training performance, as better initial weights, for example, can be missed more easily.

Finally, one of the biggest training issues is to choose good *hyperparemters*. Hyperparameters, like the number of hidden neurons, number of layers, the type of activation function, learning rates etc. (Feurer & Hutter, 2019), strongly impact the convergence rate of the backpropagation algorithm and hence strongly impact the computational effort required for training (Yang & Shami, 2020). In addition, a wrong choice of hyperparameters can lead to overfitting. Unfortunately, there is no 'best choice' hyperparameter setup, as the setup is always dataset-dependent, and the best hyperparameter setup has to be found during the training process (Feurer & Hutter, 2019).

In this study, the hyperparemeters of the MLP are tuned by training several MLPs with different hyperparameter combination. Input features are scaled to make sure that they are all within the same range. Randomisation is counteracted by setting a fixed seed for the testing and training split and for the random selection of data points during latitudinal balancing (refer to Section 5.2). The initialisation of weights and biases is also fixed by using a seed. Various seeds for initialisation are tested during training.

The overfitting problem is counteracted by applying regularisation and by using cross-validation during training. Cross-validation is a resampling method that splits up its input dataset into different subsets (*folds*). The number of folds is set by *cv_folds*. The folds are then iteratively assigned a different role as training or validation data set (cv_folds - 1 are assigned for training) and used accordingly for training or validation (refer to Mrziglod (2018) for further information).

Modifications in the regularisation terms to improve the extrapolation performance have not been incorporated, as the training dataset was assumed to be large enough to cover the full range of IWP values.

Implementation details

The implementation of the IWP regressor in SPARE-ICE is directly connected with the implementation of the ice cloud detector in SPARE-ICE. In the following paragraphs, the main steps required for the IWP regression are outlined. The imple-

mentation of the IWP regressor in SPARE-ICE can be split into two parts: the first part deals with the training of the IWP regressor (Figure B.4), the second part deals with the generation of the retrieval IWP dataset (Figure B.5).

The training of the MLP takes place in log space (logarithm to base 10), because of the high dynamic range of 10^{-2} g/m^2 to 10^4 g/m^2 of IWP. This ensures that the reduction of differences between retrieved IWP and high target IWP values is not given preferential consideration during training (Mrziglod, 2018).

Before the training can start, the training, validation, and test datasets first have to be created. When cross-validation is used, the input dataset (in Figure B.4 represented by MHS_2C-ICE_AVHRR collocations) has to be split only into two parts, one for testing (*test_data*) and one for cross-validation (*train_validation_data*). For this purpose, the dataset is shuffled and then split according to a percentage allocation (*test_size*). test_size of the dataset are selected for test_data, the remaining points are assigned to train_validation_data.

Additional inputs for the training process are an elevation and land-sea mask dataset which serve as auxiliary input features, and the names of all features from the input dataset that were selected for the training of SPARE-ICE.

To train the IWP regressor, a `Pipeline` from sklearn.pipeline is set up with `RobustScaler` from sklearn.preprocessing (quantile range between 15 and 85) and with the neural network estimator `MLPRegressor` from sklearn.neural_network (Figure B.4). `RobustScaler` is used for feature scaling. `Pipeline` bundles`RobustScaler` with `MLPRegressor`.

Afterwards, a hyperparameter grid is defined which will be used later on for testing different hyperparameter combinations in order to find the best hyperparameter setup for `MLPRegressor`. The hyperparameter grid uses the hyperparameters listed in Table B.1.

Description of the parameters can be found in the sklearn documentation of the MLPClassifier[2]. For finding the best hyperparameters, only *random_state* and *alpha* are varied. The search for the best hyperparemter set takes place during a grid search via `GridSearchCV` from sklearnmmodel_selection.

During the grid search, the estimator `MLPRegressor` is trained and evaluated with different parameter combinations in parallel via cross-validation on *train_validation _data* to find the best parameter choice. For scoring, the grid search uses the R^2 score from `MLPRegressor`.

---

[2]`https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html#sklearn.neural_network.MLPClassifier`, last visited: 03.07.2023, 13:10

After finding the best hyperperameter set according to the scores, the `MLPRegressor` is re-trained on the whole training set and returned as the best estimator. Afterwards, the score (R^2 score) of the regressor is calculated on *test_data*. Then the final
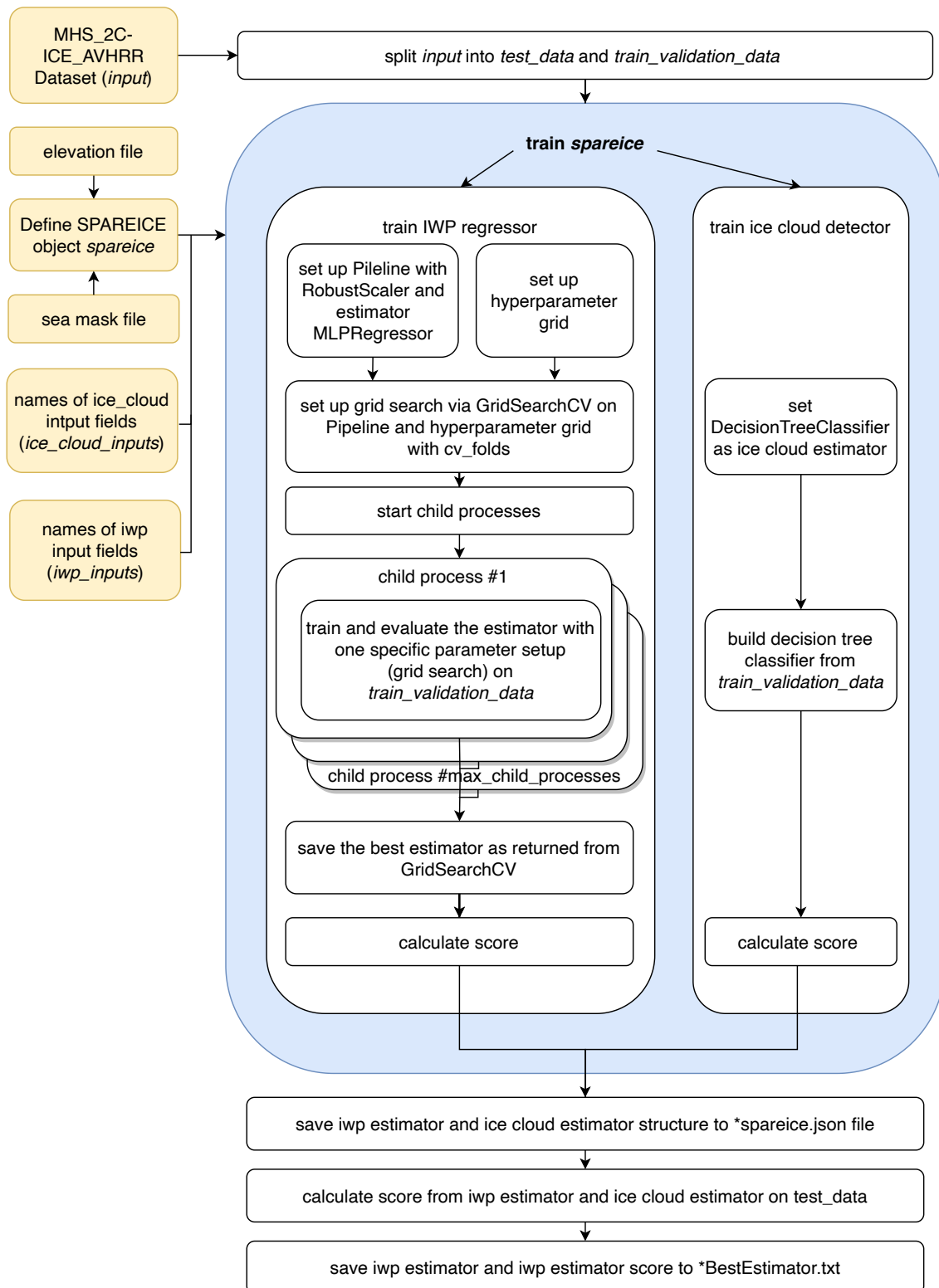
Figure B.4: Simplified depiction of the SPARE-ICE training workflow with its main inputs and outputs.
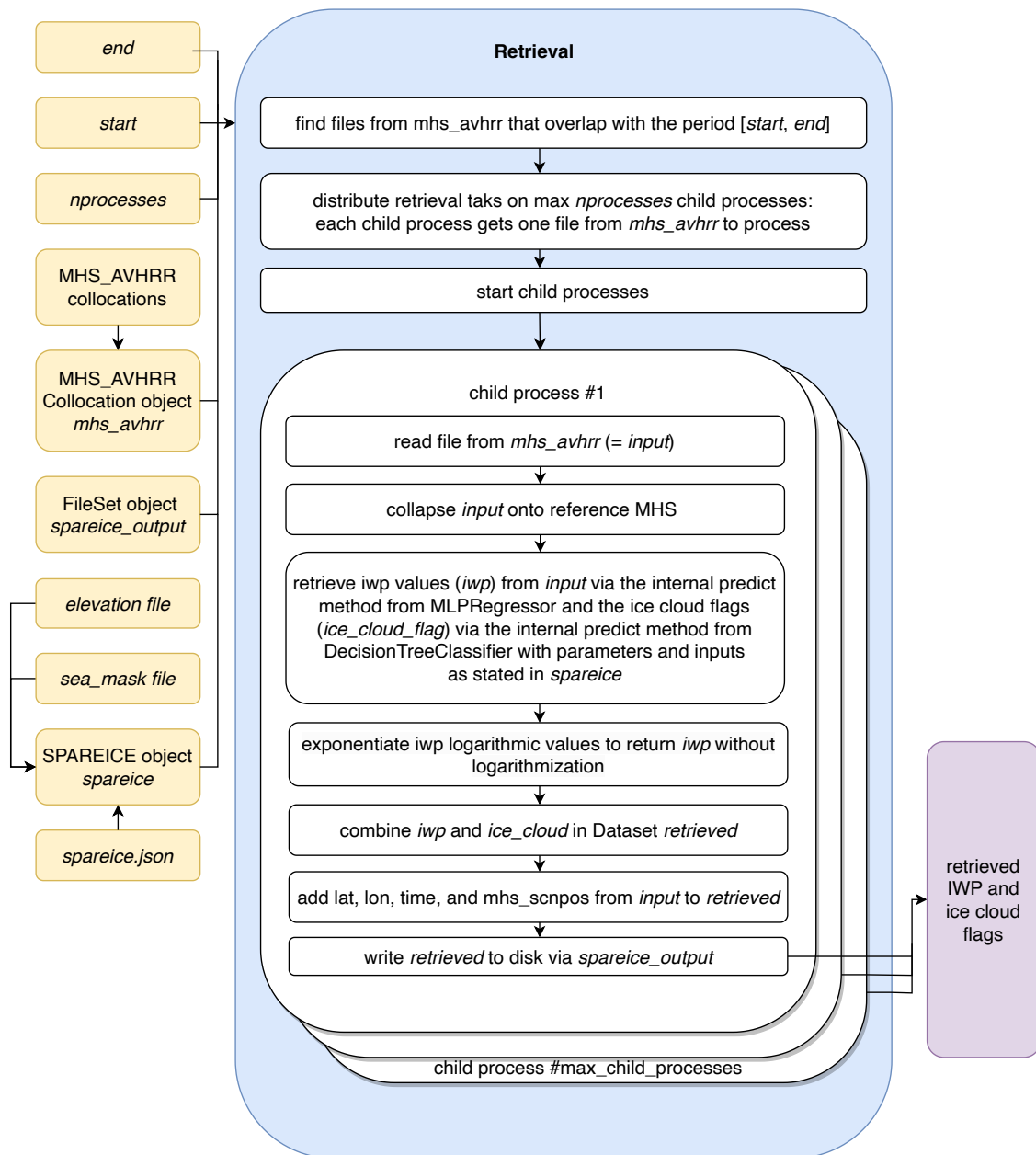
Figure B.5: Simplified depiction of the IWP retrieval workflow of SPARE-ICE with its main inputs and outputs.

structure of the `MLPRegressor`, including the input feature names and the final choice of parameters, is saved to a spareice.json file.

To generate the retrieval IWP values, MHS_AVHRR collocations (*mhs_avhrr*), all auxiliary training features, and the spareice.json file previously created during training have to be provided (Figure B.5). In addition, a time period must be specified by *start* and *end* and an output `FileSet` object (*spareice_output*) must be given.

All collocation files from *mhs_avhrr* that fall within the period of interest are then distributed on different child processes. The number of initiated child processes can

hidden_layer_sizes	hidden layer #1: 15
	hidden layer #2: 10
	hidden layer #3: 3
solver	lbfgs (quasi-Newton-method)
activation	tanh
random_state	[0, 42, 100, 3452]
alpha	[0.1, 0.001, 0.0001]

Table B.1: Hyperparameter grid variables for optimising the hyperparameters of the MLP in SPARE-ICE. *hidden_layer_sizes* contains the number of neurons in the hidden layer. *solver* is the solver used for weight optimisation. *activation* is the activation function used for the hidden layers. *random_state* is the *seed* used for the random number generation that is used for the weights and biases initialisation. alpha is the strength of the L2 regularisation.

be limited by setting the parameter *nprocesses*. Each child process only handles one collocation input file at a time.

Each child process first reads in a collocation file from mhs_avhrr and collapses it onto its MHS reference. Then the IWP values are retrieved from the auxiliary input features and from the selected input features that are taken from the read-in mhs_avhrr dataset with the help of the previously trained `MLPRegressor`. The setup of the `MLPRegressor` is described by the parameters from the .json file. Resulting IWP values are then back-transformed from log space and saved in the variable *iwp* in the xarray.Dataset *retrieved*. retrieved is then stored later as a SPARE-ICE retrieval dataset.

B.3 Ice cloud detector

This section first briefly explains what Decision Tree Classifiers are. Then the advantages and disadvantages of DTCs are discussed and the implementation of the ice cloud detector in SPARE-ICE is outlined.

Introduction to Decision Tree Classifiers

DTCs are maximum likelihood classifiers that use multi-stage decision logic to classify data (Hauska & Swain, 1975). They belong to the tree-based supervised learning techniques where the *target* is categorical (Wilkinson, 2004). A DTC consists of a *root node*, *intermediate nodes* (*decision stages*) and terminal nodes (*leaves*). Each intermediate node describes an attribute in the data and the edges of the tree describe a decision based on the attribute (Hagenlocher, n.d.; Quinlan, 1996). The

leaves contain the class labels and are used for the final classification (Hauska & Swain, 1975).

The *depth* of a DTC is the maximum length between the root node and a leaf node (Azad & Moshkov, 2023). The maximum depth of the tree would be N-1, with N being the number of training samples. Choosing a tree depth close to N-1 as depth is not recommended, as it would lead to drastic overfitting.

In the case of a binary decision tree, each node, apart from the leaves, has two edges emerging from it (Wilkinson, 2004). In this case, one can imagine the tree to be simply based on yes/no questions, where the question is, whether an attribute of the input fulfils the given condition as stated in the node or not. The conditions of the yes/no questions are learned from the training dataset during the training period. The training dataset contains classes as target for the training. After the training, a new input data point is guided through the yes/no questions until it reaches one leave. The input data point is then classified according to the category indicated by the leave. (Hagenlocher, n.d.)
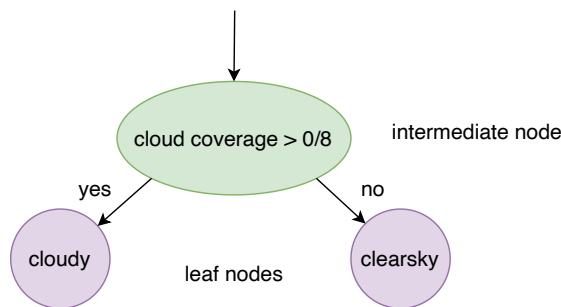
Figure B.6: Extract of a DTC: In the intermediate node, the attribute 'cloud coverage' is compared against the threshold 0/8. According to the decision based on the comparison, the scene is classified as 'cloudy' or 'clearsky'.

Figure B.6 illustrates an example classification of a DTC based on one intermediate node. The decision shown in Figure B.6 is only an example and is not used in the SPARE-ICE classification. A simplified example of a DTC, more similar to the one used in SPARE-ICE, can be found in Mrziglod (2018, p.42).

Advantages and disadvantages

One of the most frequently mentioned advantages of decision trees is that they are relatively easy to interpret as they are easy to visualize and their decisions are traceable (Hagenlocher, n.d.; Charbuty & Abdulazeez, 2021; Safavian & Landgrebe, 1991; Song & Ying, 2015). Another major strength of them is that they are insensitive to different feature scales (Zhuang, Wang, Bendersky & Najork, 2020). Besides, since they focus on salient attributes for their decisions, decision trees have a strong

tolerance towards irrelevant attributes and are less affected by high computational costs that are connected to high number of features (*high dimensional data*) (Hodge & Austin, 2004; Hagenlocher, n.d.). Another advantage is that decision trees are robust, which in this case means that they can withstand outliers in the input data quite well (Hodge & Austin, 2004; Song & Ying, 2015). However, creating the yes/no questions for the nodes might be strongly affected by outliers, depending on the method chosen for constructing the questions. For example, when using a non-robust method like principal component analysis (Serneels & Verdonck, 2008) for constructing the questions (Mrziglod, 2018, no source of origin indicated), outliers have a strong effect. Another drawback is that a wrong construction of the tree (e.g. too many nodes and/or subtrees) can leads to overfitting (Hagenlocher, n.d.). This issue can be addressed for example by *pruning*, which refers to the replacement of subtrees that do not contribute much to the predictive accuracy of the tree by a leaf (Quinlan, 1996).

A single decision tree can also be quite unstable. This means, when using different subsets from the training data, different trees may be constructed. To improve the stability, one can train multiple decision trees on different subsets of the training data and then combine their classification choices into one final classification (Hagenlocher, n.d.; Quinlan, 1996; Breiman, 1996). This approach of ensemble learning is for example used in Random Forest Classifiers (Pal, 2005) and can substantially improve the accuracy of the classification (Breiman, 1996).

Finally it should be pointed out that decision trees can be sensitive to imbalanced datasets. In case there is an unequal amount of information for each class contained in the training data, the classifier will have an imbalance towards the majority class when the information gain or the Gini index is used as a metric for the segmentation of the tree (W. Liu, Chawla, Cieslak & Chawla, 2010). This case can be avoided by ensuring a balanced representation of the classes during training.

Implementation details

In the SPARE-ICE implementation, the ice cloud detector, similar to the IWP regressor, goes through a training and a retrieval step that are implemented separately (Figure B.4 and Figure B.5). The training and retrieval step of the IWP regressor and the ice cloud detector are both simultaneously initiated by calling the same methods. The training and retrieval for the IWP regressor and the ice cloud detector are then executed sequentially.

For the training (Figure B.4), the ice cloud detector sets the class `DecisionTree Classifier` from sklearn.tree with a maximum tree depth of 12 as estimator. To

obtain a deterministic behaviour for the 'random' permutation that the `Decision TreeClassifier` applies at each split[3], a seed is fixed (seed = 5). As a measure, to quantify the quality of each split within the tree, the Gini impurity[4] index is used.

Based on these characteristics, the tree is build during training from train_validation _data. As target, the boolean variable *ice_cloud* from train_validation_data is taken. ice_cloud represents *ice cloud flags* that are either true or false, depending on whether the averaged 2C-ICE IWP values over one MHS pixel are greater than (flag = *True*) or equal to (flag = *False*) 0 g/m^2.

After the tree has been built, its performance on train_validation_data is evaluated by calculating how many times the predicted ice cloud flag from the decision tree matches exactly the ice cloud flags from the target in relation to the overall number of samples in train_validation_data. In contrast to the training of the IWP regressor, only one DTC trained. The structure of the trained decision tree is then saved to spareice.json and the final score of the ice cloud detector is evaluated based on test_data.

For the retrieval of the ice cloud flags on new data during the retrieval step, the procedure is the same as for the MLP until the estimator is applied (Figure B.5). To retrieve the ice cloud flags, the trained `DecisionTreeClassifier` is used. The resulting ice cloud flags from the `DecisionTreeClassifier` are stored in the boolean variable *ice_cloud* in the xarray.Dataset *retrieved* and then saved via the `SPAREICE` object *spareice*. If an element of the retrieved ice cloud flag is true, an ice cloud was detected by the decision tree.

B.4   Application of quality indicators

The following sections explains in more detail how the pre-processing toolkit extracts the quality flags from the quality indicators and how quality indicators and quality flags can be selected in the toolkit. Brief justifications for the default selection of the quality flags are also given.

---

[3]`https://scikit-https://www.overleaf.com/project/64bfbdbcc1842d00b0366639learn .org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html`, last visited: 03.07.2023, 10:09

[4]`https://scikit-learn.org/stable/modules/tree.html#tree`, last visited: 05.07.2023 17:56

Dataset origin	Quality indicators	Quality indicator reference	Remarks
MHS	qualind, chanqual, scanqual, dataqual	mhs_qualind, mhs_chanqual, mhs_scanqual mhs_dataqual	dataqual is not available in the MHS datasets of NOAA-18
AVHRR	qualind, calqual, scnlinqual,	avhrr_qualind, avhrr_calqual, avhrr_scnlinqual	
2C-ICE	dataquality	cloudsatR05_dataquality, cloudsatR04_dataquality	

Table B.2: Quality indicators available in the pre-processing toolkit with their respective quality indicator reference. The references are used to select the respective quality indicators.

Implementation details

The second column of Table B.2 lists the quality indicators that are available in the pre-processing toolkit for MHS, AVHRR, and 2C-ICE respectively.

Each of the quality indicators contains information that can be related to specific quality flags. The quality flags each indicate a specific quality abnormality. This means that before the quality abnormalities can be checked, the quality flags have to be extracted from the quality indicators.

For this purpose, all quality indicators selected for pre-processing must be read in as variables with the respective dataset first. Each of these quality variables contains one integer number for each data point that has to be transformed into bit representation. The bits are then associated with the individual quality flags. If the bit at the bit position assigned to a particular quality flag is zero, no abnormality according to the quality flag could be detected. If the bit is set to one, an abnormality according to the quality flag was detected. Which quality flag is assigned to which bit position is summarised in Table D.1.

Figure B.7 shows the main steps of the process that is undertaken in order to assess whether one data point of an AVHRR dataset is flagged as abnormal or normal. For the example it is assumed that the quality indicator *calqual* from AHVRR was selected via *avhrr_calqual* in *qualinds*. Out of *avhrr_calqual*, the quality flags *some_bad_prt* and *some_bad_bdcounts* were selected. Both quality flags refer to calibration deficiencies. The assumption is made that for the processed data point, the *calqual* variable is set to the integer number 70.
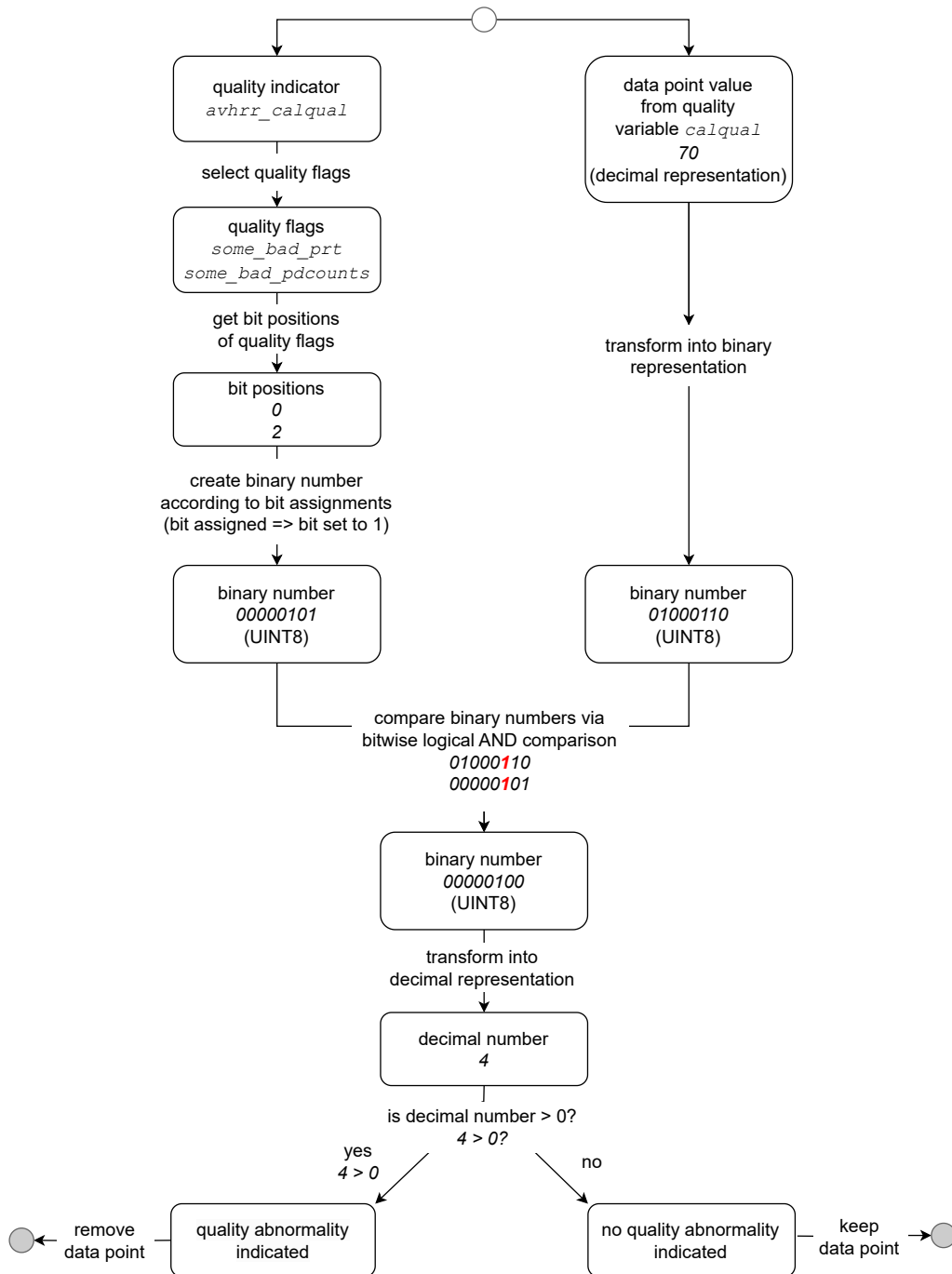
Figure B.7: Illustration of the process performed for the quality check of one data point by the pre-processing toolkit. The white circle indicates the beginning, the grey circles each indicate a possible end of the processing. An example is shown in italics. In the case of the example, a quality abnormality is detected.

For simplicity, all binary numbers in Figure B.7 are represented as unsigned 8-bit integers (UINT8). In the implementation, the *calqual* variable contains signed 32-bit integers. In both cases, the bit numbering starts as zero for the least significant bit.

For applying the quality flags, the algorithm first checks the assigned bit positions of the selected quality flags. According to Table D.1, bit positions zero and two are assigned to the quality flags respectively. If one composes a new binary number from these positions by setting the respective bit positions of the flags in the new binary number to one, the resulting binary number is 00000101.

The *calqual* variable value is also transformed into binary representation. The result is the binary number 01000110.

Afterwards, both binary numbers are compared bitwise using the logical AND operation. The comparison reveals whether the quality indicator reports one of the abnormalities indicated by the quality flags. The result represents a binary number whose bits are set to one only at the position where two bits that were set to one each were compared. If the binary number contains at least one bit that is set to one, at least one of both quality abnormalities is indicated by the quality indicator. The resulting binary number from the comparison is then transformed into decimal representation and compared against the threshold zero. If the number is larger than zero, a quality abnormality is indicated. The boolean result from the comparison against the threshold is referred to as the final flag of the quality indicator.

In the example from Figure B.7, the bitwise comparison resulted in the binary number 00000100. Hence, according to the quality indicator, only the quality abnormality *some_bad_bdcounts* has occurred.

In decimal representation, the binary number 00000100 represents the number four. Thus, a quality abnormality is indicated overall. The corresponding data point is therefore removed from the dataset. For the result it is unimportant that the abnormality is only based on one quality flag.

In the implementation of the pre-processing toolkit, the application of quality indicators is performed as described above, but handled via array operations. Multidimensional quality indicators can also be applied. During the process, flag variables are created that replace the quality indicator variables. Each flag variable contains the final flags of its quality indicator. After the removal of the abnormal data points, the flag variable is deleted from the dataset.

Selection of quality indicators and quality flags

The user can decide for themselves which quality indicators and which quality flags they would like to apply. In order to achieve the highest possible flexibility in the selection of the various quality indicators, the quality indicators can be selected by their name, supplemented with a prefix (see third column from Table B.2), in the

dictionary *qualinds*. An example of *qualinds* is shown in Figure B.8.

```
qualinds = {'MHS'.   : [mhs_dataqual, mhs_qualind, mhs_chanqual],
            'AVHRR'  : [avhrr_calqual, avhrr_scnlinqual, avhrr_qualind],
            '2C-ICE' : [cloudsatR05_dataquality]}
```

Figure B.8: Example setup of *qualinds*. Via *qualinds*, the individual quality indicators of the respective MHS, AVHRR and 2C-ICE datasets can be selected.

qualinds can be defined in a separate file or imported from qualinds.py. In *qualinds*, quality indicators can be added or removed as desired. For 2C-ICE two quality flag indicators are available (Table B.2), although only one quality variable exists, because the bit assignments of the quality flags changed between 2C-ICE release P1_R04 and P1_R05. In Figure B.8, three quality indicators for MHS data, three quality indicators for AVHRR data, and the quality indicator for 2C-ICE P1_R05 were selected.

The quality indicators in *qualinds* are references that refer to dictionaries from *qualinds.py*. In *qualinds.py*, dictionaries for each quality indicator exist. A quality indicator dictionary is generally structured as illustrated in Figure B.9. Additional quality indicators can be added to *qualinds.py* as desired, following the scheme shown in Figure B.9.

For each quality indicator dictionary, default settings for the quality flags are provided in 'default_quality'. If the user wants to select their own quality flags, 'set_default _qual' has to be switched off and the modified quality flag choices have to be inserted into 'sel_quality_attrs'. All available quality flags including their bit position assignment are listed in 'attr_bit_assignment'.

Default quality settings

For the SPARE-ICE IWP data records that resulted from this study, the default quality flags were chosen for the pre-processing. The default quality flags for each quality indicator dictionary were selected in order to cover as many of the indicated error sources as possible. Thus, the choice of default quality flags is conservative. An overview of the default quality flags can be found in Table D.1. The quality flags can roughly be categorised into those dealing with calibration problems, GPS problems, or noise.

All but two quality flags that indicate a calibration problem are selected for the default quality settings, because calibration is necessary to ensure continuously accurate, consistent and traceable measurements and a failure or problem during calibration could strongly impact the quality of data points. Excluded from the

```
<avhrr/mhs/cloudsatR04/cloudsatR05>_<quality indicator name>

   = {'name'                 : <name of the quality variable as specified in
                                 the data set>,
      'dimension'            : [<name of dimension(s) of data variable
                                 specified by name>],
      'dtype'                : <numpy data type of data variable specified by
                                 name>,
      'set_default_qual'.    : <if True: default quality set is chosen;
                                  if False: own quality flags have to be chosen
                                  via sel_quality_attrs>,
      'default_quality'      : [<quality flag name keys that define the
                                 default quality>],
      'channneldim_name'     : <name of channel dimension>,
      'sel_channels'         : [<tuples of the form: ('channelNumber',
                                 True/False, channelIndex). Set True or False
                                 depending on demand>],
      'sel_quality_attrs'    : [<if set_default_qual is set to False, the
                                 user has to insert their own preferred
                                 quality flag name keys here>],
      'attr_bit_assignment': {<all quality flags available for the quality
                                 indicator. Each entry is structured in the
                                 following way:
                                 'qualityFlagNameKey':[bitAssignment,
                                 descriptionOfQualityFlag]>}
      'comment'              : <any additional comments>

   }
```

Figure B.9: Common structure of a quality indicator.

default settings are, with reference to their key in their corresponding quality indicator dictionary, *some_no_chcalib* from *mhs_scanqual*, and *some_no_chcalib* from *avhrr_scnlinqual*. This was done to prevent points from being flagged as abnormal in a generalised way across all channels. Otherwise there would have been a risk that the flag refers to a channel which is not relevant for the current channel selection. Instead of including those quality flags, I recommend using quality indicators that address calibration problems specific to individual channels. For example *chanqual* from MHS can be used to address calibration problems in channel 1, 2, and 3. For AVHRR, *calqual* can be used to address calibration problems in channel 3B, 4, and 5. For AVHRR channel 1, 2, and 3A, no onboard calibration is available (Xiong, Butler, Cao & Wu, 2018).

When a GPS problem occurs, an allocation of a data point to a specific location on Earth at a specific time is impeded. This also impacts the assignment to a specific atmospheric state. While for example inter- or extrapolation may be used to handle the problem of missing GPS values, those methods introduce an additional error to the GPS values. However, accurate latitude, longitude and time values are crucial for the collocation methods in SPARE-ICE (see Appendix B.1). Therefore, all GPS indicators are also included in the default quality settings.

Noisy data can lead, for example, to poor prediction results and decreased classification accuracy (Gupta & Gupta, 2019; Hasan & Chu, 2022). When noise is introduced to a machine learning model, it can also increase the model complexity and time required to train the models (Gupta & Gupta, 2019; Hasan & Chu, 2022). Hence, except from one quality flag, all quality flags that deal with noisy data are included in the default quality settings. Excluded from the default settings is *refsun3b* from *avhrr_qualind*. This quality flag is excluded because for the SPARE-ICE IWP data records that resulted from this study, AVHRR channel 3 is excluded (see Chapter 5).

B.5   Split into ascending and descending nodes

To differentiate between ascending nodes and descending nodes, functions for SPARE-ICE IWP retrieval datasets as well as for 2C-ICE datasets were created in this study. The differentiation between the nodes is based the gradient of the latitude coordinates between two successive measurements along track direction.

The next paragraphs explain in more detail the underlying mechanisms of the functions used to assign the data points to ascending or descending nodes.

Implementation details

Within the functions used for the splitting into descending and ascending nodes, one new variable is created for each dataset that indicates the respective node by a number. The number 0 indicates a descending node, the number 1 indicates an ascending node, the number 2 indicates that no change between latitudes between two consecutive points was detected and the number 10 indicates that the node can neither be flagged as ascending nor as descending. Under normal circumstance, the case corresponding to number 2 should never occur. The flag corresponding to the last data point of each dataset is always set to 10. Otherwise the consecutive data set would have to be read in for the classification of the last point. The classification of one additional data point would not justify the computational capacities involved for reading one additional file if only one dataset is of interest. Therefore, the data point is classified as non-classifiable. Based on the node flag variable, the data points that could be assigned either to the ascending (flag number 1) or to the descending (flag number 0) node are selected and returned.

In the case of SPARE-ICE, it must be taken into account that the retrieval data records contain scan lines as well as scan position. To take this aspect onto account, an additional function for SPARE-ICE retrieval data records was created. Within this function, a specific scan position from each scan line is selected. The latitudinal

coordinates assigned to the data points that belong to this specific scan positions are then used to calculate the latitudinal gradients. For this study, scan position 45 was chosen as a representative. This means, for calculating the latitudinal gradient, only scan lines that contain scan position 45 are considered. If one scan line does not contain scan position 45 due to e.g. an abnormality, the whole scan line is removed. In order not to distort the gradient, no other scan position is selected as a substitute for scan position 45 for this case. Thus, entire scan lines are sometimes not taken into account for the ascending or descending node calculations of the SPARE-ICE datasets. For 2C-ICE, on the other hand, all available latitudes are taken into account. This is because 2C-ICE does not contain different scan positions.

Appendix C

Workflows

This chapter contains detailed workflows that describe the implementation of the collocator component and of the training and retrieval process of the IWP regressor and ice cloud detector in SPARE-ICE. The workflows can be particularly useful for future developers of SPARE-ICE.

C.1 Collocation workflow

Primary FileSet (primary)

Secondary FileSet (secondary)

start

end

max_interval

max_distance

processes (= max number of processes)

- - - - Generator
- Input
- from files/fileset.py
- from collocations/common.py
- from collocations/collocator.py

call CollocationObject.search(...)

search

create generator object *collocated_files* and iterate over it

collocated_files
= collocator.collocate_filesets(
[primary, seondary],...)

match

1. collect FileInfos of all primary and secondary Files that fall within the time period of interest via **find**
files1 = list of FileInfos of all primary files in [start, end]
files2 = list of FileInfos of all secondary files in [start-max_interval, end+max_interval]

2. select times from files1 & files2 and extend the times for file2 further by +-max_interval

3. search for all overlapping secondary time intervals for each primary:
tree = IntervalTree(times2) with IntervalTree from typhon.trees
results = tree.query(times1)

collocate_filesets

1. find temporally overlapping FileSets via *.match* and collect them in *matches: matches* is a list containing all primary (p) matches as tuples:
$[(p_1,[s_{p1,1},...,s_{p1,np1}]),..., (p_i,[s_{pi,1},...,s_{pi,npi}]),...,(p_m,[s_{pm,1},...,s_{pm,npm}])]$
with p and s being the FileInfos of the primary i=1...m, and secondaries respectively, and npi being the n-th secondary belonging to primary i where n is the maximum number of secondaries assigned to primary i. A tuple $(p_i,[s_{pi,1},...,s_{pi,npi}])$ represent a *match*.

2. distribute all matches from *matches* evenly over child processes and start child processes

3. create *errors* queue and *results* queue

matches
= list(primary.match(secondary, start, end, max_interval,...))

yield matches

yield files1 = list(self.find(start, end, ...)
files2 = list (other.find(start-max_interval, end+max_interval,...)

find

1. include previous sub directory in search range

2. find files that match the search interval according to their sub-directory name and filename

3. maximal search time resolution: seconds

Main process

- check which child processes are still running and which ones are crashed
- display progress of processing
- yield results from *results* queue

start child processes

target for each child-process:
_process_caller

child process #1

child process #min(processes, #matches)

push results to *results* queue & errors to *errors* queue

_save_and_return

1. set filename according to collocations.attrs['start_time'] and attrs['end_time']

2. apply a post_processing function if given

3. save *collocations* to disk via *write* method from FileHandler from CollocationObject

_configure_pool_and_worker_args

- if worker_type = 'processes': configure processes from class concurrent.futures.ProcessPoolExecutor
- if worker_type = 'thread', configure threads from classconcurrent.futures.ThreadPoolExecutor

results = self._save_and_return

return filename

return pool class, pool arguments, and worker arguments

self._configure_pool_and_worker_args(*args,**kwargs)
with worker_type = 'thread'

_process_caller

- is wrapper around **_collocate_matches**
- bundling of collocations
  - if bundle is None: initiate saving of collocations to disk via **_save_and_return** and push results to *results* queue
- if child process crashed: push error messages to errors and results queue and then terminate child process

imap

1. create deque *worker_queue* for collecting thread results
2. configure (-> **_configure_pool_and_worker_args**) and initiate threads. Each thread gets the task to execute self.**_call_map_function**
- a finished thread waits until a result has been yielded from *worker_queue* before taking over the next task if there are more tasks than threads

yield *collocations* Dataset and attributes

yield from self.imap(**map_args, worker_type='thread')

submit task to thread

return tuple (fileInfo, fileContent)

yield results from *worker_queue*

icollect

- is wrapper around **imap** that sets up args for imap

thread#1,..., thread#max_number_of_threads

_call_map_function

1. read file via *read* method from FileHandler from FileSet object

primary_loader
= self.icollect(primaries, ...)
secondary_loader
= other.icollect(unique_secondaries)

yield results from *worker_queue* from imap

align

1. extract FileInfos from matches: primary = FileInfos from primary, secondary = FileInfos from secondaries
2. remove duplicates from secondary files that are assigned to one primary file according to their FileInfos: results in *unique_secondaries*
3. find matches between primary and secondary FileSets, read their content in parallel threads and yield their collocations
   - create generator objects *primary_loader* and *secondary_loader* for reading files via **icollect** and iterate over them
   - create dictionary *cache* that caches each secondary file that was read as long as it is still needed for another primary that was assigned to child process #1.
     key: FileInfo of secondary, value: xarray of secondary
   - set up counter *secondary_usage* that reflects how often each secondary is needed in total (for all primaries that were assigned to child process #1). Every time a secondary was used, its count is decreased by one. If its count is zero, the secondary is deleted from *cache*.

collocated_matches =
self._collocate_matches(**kwargs)

loaded_matches =
primary.align(secondary, match,...)

yield list of tuples
$([p_q,\text{FileInfo}, p_{q,\text{Dataset}}], [s_{pq,r},\text{FileInfo}, s_{pq,r,\text{Dataset}}])_r$ for
$r = 1,...,n_{pq}$, $q \in \{1,...,m\}$

_collocate_matches

1. create generator object *loaded_matches* via **align**
2. for each *match* from *matches* with primary p_q with $q \in \{1,...,m\}$ that has been assigned to child process #1:
   1. iterate over generator object *loaded_matches* (-> **align**) and collect results in *got_matches*
      got_matches = $[([p_q,\text{FileInfo}, p_{q,\text{Dataset}}], [s_{pq,1,\text{FileInfo}}, s_{pq,1,\text{Dataset}}]), ...,$
      $([p_q,\text{FileInfo}, p_{q,\text{Dataset}}], [s_{pq,npq,\text{FileInfo}}, s_{pq,npq,\text{Dataset}}])]$
   2. separate FileInfos and Datasets from primary and secondaries from *got_matches*
   3. concatenate all secondary Datasets from *got_matches* along their first time dimension
   4. collect collocations between primary Dataset and the concatenated secondary Datasets via **collocate**. Result: *collocations* Dataset
   5. add attributes of primary and one secondary to *collocations*. As a representative for the concatenated secondary files, the attributes from the first secondary are taken.

_get_common_time_period

1. choose a common time period in which collocations are possible according to max_interval and [start, end]

2. extract all primary and secondary time stamps that lie within the common time period and remove all nan values from the first dimension of the remaining time values. Results: *primary_period*, *secondary_period*

primary_period, secondary_period = self._get_common_time_period (primary, secondary, max_interval, start, end)

return *primary_period*, *secondary_period*

_prepare_data

1. if max_interval is not None, get time period, where primary and secondary points can overlap according to max_interval via **_get_common_time_period.**
Results: *primary_period*, *secondary_period*

2. sort *primary_period* and *secondary_period* by time

3. select the data points from primary and secondary that fall within *primary_period* and *secondary_period* respectively.
Results: primary = time restricted primary, secondary = time restricted secondary

4. flatten selected primary and secondary by stacking the dimensions from lat, lon & time to dimension 'collocation' (dimension 'collocation' replaces the previous dimensions e.g. ('scnline', 'scnpos'))

return *collocations*
Dataset in compact form with dimension 'collocation'

collocations = self.collocate((primary.name, primary), secondary.name, secondary), **kwargs)

return time restricted and flattened primary and secondary Datasets

primary, secondary = self._prepare_data(primary, secondary, max_interval, start, end)

_temporal_check

1. get time difference between each primary and secondary pair and compare it against max_interval

return boolean indicator for each pair whether the time temporal condition is met

self._temporal_check(...times and pairs)

collocate

1. <u>sort</u> primary Dataset and secondary Dataset by time

2. select common time period between primary and secondary and flatten resulting Datasets via **_prepare_data**

3. extract lat, lon & time from primary and secondary and remove all indices, where lat and lon are nan

4. find collocations between primary and secondary xr.Datasets

   ○ if max_interval is None, search for spatial collocations only via **spatial_search**

   ○ if max_distance is None, search for temporal collocations only (not yet implemented)

   ○ if max_interval and max_distance are set:

      ▪ if primarytime.size * secondarytime.size > 100_000: perform temporal pre-binning via **spatial_search_with_ temporal_prebinning** before performing the spatial search

      ▪ else: perform spatial search directly via **spatial_search**

5. check whether all spatial collocations meet the temporal condition via **_temporal_check** and only select the points from *pairs* and *distances* that fulfil the temporal criterion

6. get *collocations* Dataset with collocations in compact form via **_create_return**

_create_return

1. select only the points from primary and secondary that are needed for pairs

2. create final collocation xr.Dataset

3. set output attributes 'start_time' to min time value from primary and 'end_time' to max time value from primary

primary, secondary, pairs, collocation info

return *collocations* Dataset

return *pairs*, *distances*

return *pairs*, *distances*

pairs, distances = self.spatial_search_with_temporal_prebinning(..., max_distance, max_interval)

pairs, distances = self.spatial_search (..., max_distance)

spatial_search_with_temporal_prebinning

1. put lat, lon, and time information from primary and secondary into pd.DataFrames with index 'time'

2. swapping: set the larger DataFrame as primary

3. create generator object *bin_pairs* via **_bin_pairs** (bins primary and secondary along their time axis) for searching for time related primary and secondary bins

4. perform a ball tree search on each primary bin pair by iteratively applying **_spatial_search_bin** on *bin_pairs* and collect results

5. stack primary bin pairs from the spatial search together

6. if primary and secondary were swapped, swap them back again and adjust *pairs* accordingly

map(
Collocator.
_spatial
_search
_bin, args)

bin_pairs = (self._bin_pairs(...))

return primary bin,
secondary bin

_bin_pairs

1. create secondary bins around primary bins. Each secondary bin k covers the time period:[primary_bin$_{k,start}$-max_interval, primary_bin$_{k}$.maxtime+max_interval]

spatial_search

- select and set optimal index choice for tree building via **_choose_points_to_build_index**
- set *build_points* and *query_points* with lat and lon from primary and secondary
- set spatial index for search query (set built up tree) via **_build_spatial_index**
- query the tree via typhon/geographical.py -> query on query_points under consideration of max_distance. Results: *pairs* (contains indices of found spatial matches) and *distances* (distances between primary and secondary pair)
- reorder *pairs* so that the primary indices are in the first row

return
pairs,
distances

self.spatial_search(lat & lon from primary and secondary, max_distance)

return
pairs,
distances

return built
up tree

self.index =
self._build_spatial_index(
**build_points)

boolean
index
choice

_spatial_search_bin

- is wrapper around **spatial_search**

self._choose_points_to
_build_index(lat & lon from
primary and secondary)

_build_spatial_index

1. check whether there is a cached balltree that is still applicable to the current data.
    - if yes: reuse it for the query
    - if no: build query index via typhon/geographical.py GeoIndex: set tree class to BallTree from sklearn.neighbours (default), use minkowski metric for distance measurement, convert lat and lon according to metric and shuffle lat & lon

_choose_points_to_build_index

1. choose via **_build_spatial_index** whether the primary or the secondary should be used for tree building depending on the size relation of primary and secondary. Basic concept: build tree with the larger Dataset & cache and reuse indices if possible

C.2 Training workflow

train

trains the IWP regressor and ice-cloud classifier

1. train ice cloud detector:

    1. set up ice cloud estimator (ice_cloud_model) via self.**_ice_cloud_model()**

    2. train ice cloud estimator via **train** method.
    Columns from train_validation_data according to ice_cloud_inputs and the target variable are selected as inputs_ice_clouds and target_ice_clouds

    3. save trained ice cloud estimator to spareice as self.ice_cloud

2. train iwp regressor:

    1. drop nan from dataset (resulted from transformation to log space)

    2. if cv_folds is None, set cv_folds to 5

    3. set up iwp regressor estimator (iwp_model) via self.**_iwp_model**

    4. train iwp regressor estimator via **train** method.
    Columns from train_validation_data according to iwp_inputs and the target variable are selected as inputs_ice_clouds and target_ice_clouds

    5. save trained iwp regressor estimator to spareice as self.iwp

self.ice_cloud
self.iwp

self.ice_cloud.train(ice_cloud_model, inputs_ice_clouds, target_ice_clouds)

self.iwp.train(iwp_model, inputs_iwp, targets_iwp)

return training score

train

- for ice_cloud_model:

1. build a decision tree classifier from the training inputs targets_ice_clouds and inputs_ice_cloud via estimator.fit(inputs_ice_clouds, targets_ice_clouds) with the fit method from sklearn.tree.DecisionTreeClassifier. Return fitted estimator

2. calculate score of the decision tree classifier via **score** method from DecisionTreeClassifier

- for iwp_model:

1. start training: estimator.fit(inputs_iwp, targets_iwp) -> start child processes. Each child process trains and evaluates a model with one specific parameter setup

2. save only the best estimator as returned from GridSearchCV as self.estimator

3. calculate score of the best estimator via **score** method from MLPRegressor

self.score(inputs_ice_clouds, targets_ice_clouds)

self.score((inputs_iwp, targets_iwp)

return score

score

refers to self.estimator.score(inputs, targets)

- for DecisionTreeClassifier:
.score refers to score method from sklearn.tree -> count how many times the predicted label (self.predict(input_ice_clouds)) match exactly the tree labels or not and return the average of matches

- for MLPRegressor:
.score refers to score method from MLPRegressor which has the R^2 score implemented. Return R^2 score of self.predict(input_iwp) with respect to targets_iwp

1. save self.iwp and self.ice_cloud via spareice.save to spareice.json file

2. calculate testing score of trained ice cloud detector and IWP regressor on test_data via SPAREICE **score** method

3. save training results to .json file via **save**

4. save scores and spareice.iwp.estimator to *BestEstimator.txt

return
iwp_score,
ice_cloud_score

spareice
.score(test_data)

score

calculates the score of the estimators using the score methods of the respective estimators

- for ice clouds:
  1. ice_cloud_score = self.ice_cloud.score(inputs_test_data_ice_clouds, target_test_data_ice_clouds)

- for iwp:
  1. drop nans from test_data
  2. iwp_score = self.iwp.score(inputs_test_data_iwp, target_test_data_iwp)

spareice.save(pathname+experimentname+'_spareice.json')

save

1. write self.iwp and self.ice_cloud to file via write method from spareice

C.3   Retrieval workflow

MHS_AVHRR collocations	

FileSet object for output files (spareice_output)

land-sea mask file

elevation file

Input	
retrieval/spareice/common.py	
files/fileset.py	
retrieval/common.py	

MHS_AVHRR Collocations object

processes (= max number of processes)

start

end

SPAREICE object (spareice)

- - - - - - Generator

call spareice.retrieve_from_collocations(MHS_AVHRR, spareice_output, start, end, processes,...)

1. retrieve IWP from MHS_AVHRR collocations in parallel via map(SPAREICE._retrieve_from_collocations,...)

Remark: If a list [MHS FileSet, AVHRR FileSet] is provided instead of MHS_AVHRR collocations, the MHS_AVHRR collocations will be automatically generated via self.collocator.collocate_filesets(...) with post_processor = SPAREICE._retrieve_from_collocations

return
list of FileInfos and retrieved Datasets

MHS_AVHRR.map(SPAREICE._retrieve_from_collocations, start, end, processes, worker_type='process',...)

map

applies a function on multiple files with parallel workers

1. via _configure_pool_and_woker_args from files/fileset.py: configure pool and worker args and find files via find method from files/fileset.py between [start, end].

2. evenly distribute the tasks (= apply SPAREICE._retrieve_from_collocations on each MHS_AVHRR collocation file) on different child processes via map method from the python multiprocessing package and initiate child processes. Each child process processes one MHS_AVHRR file at a time.

3. collect FileInfos_from_retrieved and retrieved in list

return
pool class, pool arguments, and worker arguments (generator object)

MHS_AVHRR._configure_pool_and_worker_args

return,
FileInfos_from_retrieved, retrieved

_configure_pool_and_worker_args

- if worker_type = 'processes': configure processes from class concurrent.futures.process.ProcessPoolExecutor

- if worker_type = 'thread': configure threads from class concurrent.futures.process.ThreadPoolExecutor

start child processes; each child process has the task to execute MHS_AVHRR._call-map_function

yield list of files (FileInfos) between [start, end]

find(start, end)

find

- find files that match the search interval

_call_map_function

- wrapper around the function that was called via .map (in this case: wrapper around **_retrieve_from_collocations**)

1. get file content of one file from FileInfo via read method from file handler of MHS_AVHRR collocations. Result: *collocations* xarray.Dataset

2. apply the function SPAREICE._retrieve_from_collocations on *collocations*

3. *write retrieved to spareice_output* via write method from file handler of spareice_output

_retrieve_from_
collocations(collocations,...)

_retrieve_from_collocations

1. collapse *collocations* on reference 'MHS'

2. rename variables from *collocations* and convert *collocations* to pandas.DataFrame

3. remove nans from *collocations*

4. retrieve IWP and the ice cloud flag via **retrieve**(*collocations*).

5. convert *retrieved* to xarray.Dataset

6. add latitude, longitude, time, and mhs_scnpos information from collocations to *retrieved*

return
retrieved

return
retrieved

retrieved
= spareice.retrieve(collocations)

retrieve

1. retrieve IWP via spareice.iwp.**retrieve**(*collocations*) Result: *retrieved* pandas.DataFrame

2. exponentiate logarithmic IWP values from *retrieve* to return IWP without logarithmization

3. retrieve ice cloud flag via spareice.ice_cloud.**retrieve**(*collocations*)

4. add retrieved ice cloud flag to *retrieved* as column

return
retrieved IWP and
retrieved ice cloud flag

spareice.iwp.retrieve(collocations)
spareice.ice_cloud.retrieve(collocations)

retrieve

1. use the estimator to get the retrieval values via self.estimator.predict(*collocations*); Estimators: e.g. DecisionTreeClassifier, MLPRegressor

2. convert output to pandas.DataFrame
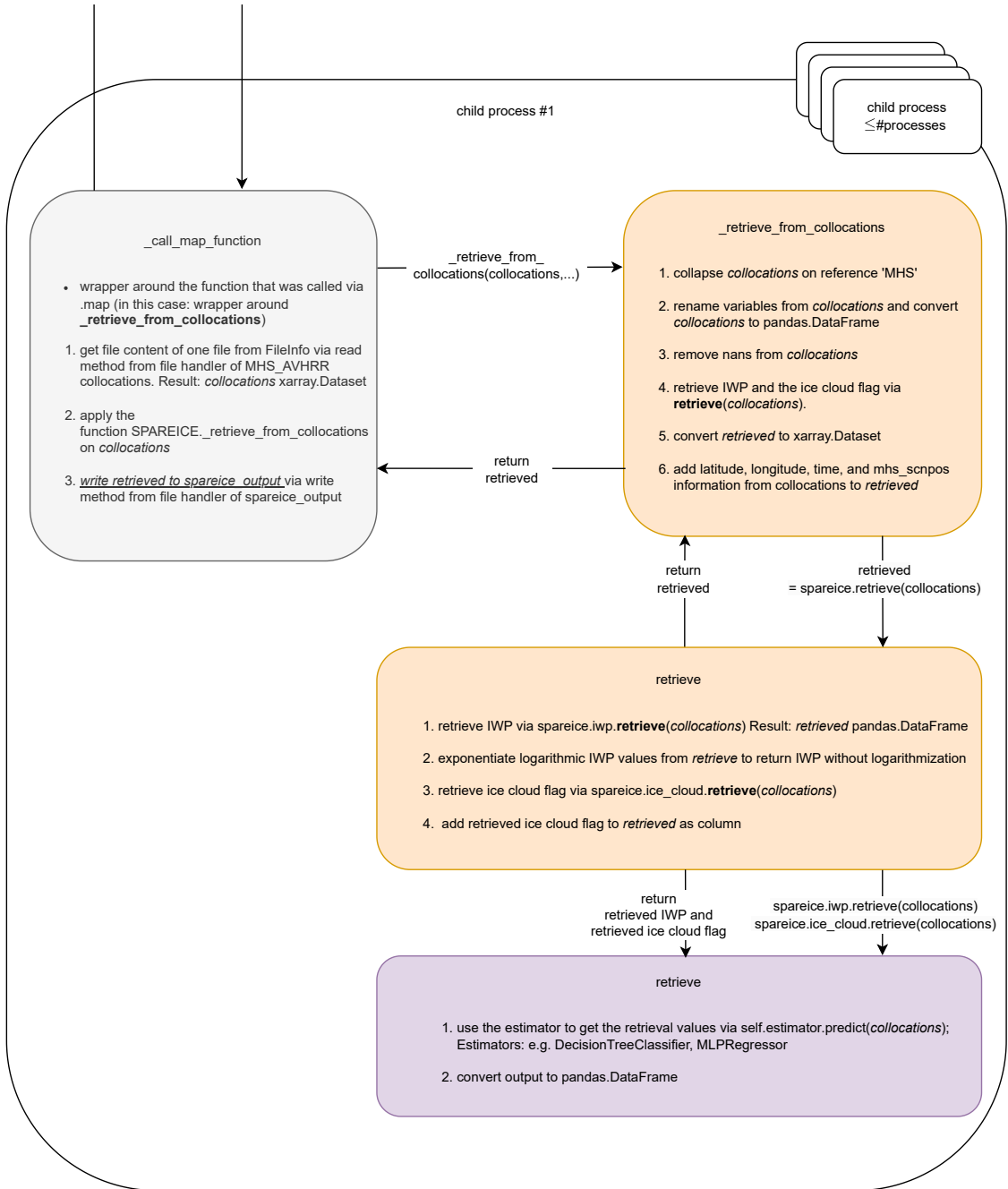
Appendix D

Overviews

This chapter contains tables referred to in this thesis. It lists the default quality flags (Appendix D.1), gives an overview of collocation inputs and thresholds from the literature and this study (Appendix D.2), and compares SPARE-ICE feature inputs (Appendix D.3) as well as SPARE-ICE retrieval periods (Appendix D.4). The duration phases of the satellites used in Section 3.1 are listed (Appendix D.5).

D.1 Quality flags

Table D.1: Overview of the quality indicators and quality flags that are available in the pre-processing package via *qualinds.py*. Each quality flag has a bit position from the respective quality indicator variable assigned to it. All quality flags selected for the default quality are marked with an 'x'. The bit positions were taken from *AAPP Documentation - Data Formats* (2021).

Quality variable reference	Quality flag	Bit position	Default
mhs_dataqual	calib2ry	30	x
	badtb_ch5	5	x
	badtb_ch4	4	x
	badtb_ch3	3	x
	badtb_ch2	2	x
	badtb_ch1	1	x
	chsmissing	0	x
mhs_qualind	dontuse	31	x
	tseqerror	30	x
			Continued on next page

Table D.1 – continued from previous page

Quality variable reference	Quality flag	Bit position	Default
	scan_preceding_datagap	29	x
	no_calib	28	x
	no_earthloc_badt	27	x
	good_1t_after_clockupdate	26	x
	instrstatus_changed_within_scan	25	x
mhs_chanqual	bad_bdcounts	5	x
	bad_svcounts	4	x
	bad_prt	3	x
	some_bad_bdcounts	2	x
	some_bad_svcounts	1	x
	some_bad_prt	0	x
mhs_scanqual	badt_tinferable	23	x
	badt_not_tinferable	22	x
	tdiscont	21	x
	trepeats	20	x
	no_calib_badt	15	x
	calib_poor_nscnlin	14	x
	no_calib_bad_prt	13	x
	calib_marg_prt	12	x
	some_no_chcalib	11	-
	no_calib_instrmd	10	x
	no_earthloc_badt	7	x
	earthloc_quest_t	6	x
	earthloc_quest_low_plaus	5	x
	earthloc_quest_not_plaus	4	x
	earthloc_quest_no_antennacheck	3	x
avhrr_calqual	no_calib	7	x
	calib_quest	6	x
	bad_bdcounts	5	x

Continued on next page

Table D.1 – continued from previous page

Quality variable reference	Quality flag	Bit position	Default
	bad_svcounts	4	x
	poor_prt	3	x
	some_bad_bdcounts	2	x
	some_bad_svcounts	1	x
	some_bad_prt	0	x
avhrr_scnlinqual	badt_tinferable	23	x
	badt_not_tinferable	22	x
	tdiscont	21	x
	trepeats	20	x
	no_calib_badt	15	x
	calib_poor_nscnlin	14	x
	no_calib_bad_prt	13	x
	calib_marg_prt	12	x
	some_no_chcalib	11	-
	no_earthloc_badt	7	x
	earthloc_quest_t	6	x
	earthloc_quest_low_plaus	5	x
	earthloc_quest_not_plaus	4	x
avhrr_qualind	dontuse	31	x
	tsecerror	30	x
	frame_preceding_datagap	29	x
	calib_poor_data	28	x
	no_earthloc_badt	27	x
	sync_lockdrop	26	x
	sync_worderrorg0	25	x
	sync_lockdrop_prev	24	x
	flywheeling	23	x
	bitslippage	22	x
	tipparity1	9	x

Table D.1 – continued from previous page

Quality variable reference	Quality flag	Bit position	Default
	tipparity2	8	x
	tipparity3	7	x
	tipparity4	6	x
	tipparity5	5	x
	reflsun3b	4	-
	reflsun4	3	x
	reflsun5	2	x
	resync	1	x
	pseudonoise	0	x
cloudsatR04_dataquality	notused	7	x
	frame_missing	6	x
	calib_cpr	5	x
	ppower_abnormal	4	x
	radartelem_abnormal	3	x
	badtb	2	x
	badgps	1	x
	raystatus	0	x
cloudsatR05_dataqual	notused	13	x
	frame_missing	12	x
	calib_cpr	11	x
	ppower_abnormal	10	x
	radartelem_abnormal	9	x
	badtb	8	x
	badgps	3	x
	raystatus	2	x

D.2   Collocation settings

Training collocations			
	Holl et al. (2014)	**Mrziglod (2018)**	**This study**
Time period	01/2007-12/2007	01/2007-08/2010	01/2007-12/2010
Passive instruments	MHS, AVHRR from NOAA-18	MHS, AVHRR from NOAA-18	MHS, AVHRR from NOAA-18
IWP reference based on active instruments	2C-ICE	2C-ICE	2C-ICE
MHS_2C-ICE collocation thresholds (Δs, Δt)	7.5 km, 10 min	7.5 km, 10 min	7.5 km, 10 min
MHS_AVHRR collocation thresholds (Δs, Δt)	7.5 km, 10 min	7.5 km, 30 s	7.5 km, 30 s

Table D.2: Collocation inputs and collocation thresholds (maximum distance Δs and maximum time limit Δt) used in Holl et al. (2014), Mrziglod (2018), and in this study

D.3   Feature choices

Table D.3: Overview of the input features used to train the IWP regressor and ice cloud identifier respectively. The values were compiled based on the Table 1 and Table 2 from Holl et al. (2014) and the .json provided for the baseline experiment from Mrziglod (2018). Whether and how the surface temperature was used in Holl et al. (2014) is not entirely clear. This work assumes the use of surface temperature according to Section 5 and the description of Figure 7 in Holl et al. (2014), taking into consideration Table 2.3 from Mrziglod (2018). The overview for Holl et al. (2014) was compiled from the overview by Mrziglod (2018) and from Holl et al. (2014). Conflicting or unclear statements are marked with a question mark. Used features are marked with an 'x'. The only difference between the input features used in this study and the input features used for the baseline experiment in Mrziglod (2018) is that AVHRR channel 3 was removed as input.

		IWP regressor			Ice cloud detector		
		Holl et al. (2014)	Mrziglod (2018)	This study	Holl et al. (2014)	Mrziglod (2018)	This study
Tool		MLP	MLP	MLP	MLP	DTC	DTC
	Input fields						
MHS	channel 1	-	x	x	-	-	-
	channel 2	-	x	x	-	x	x
	channel 3	x	x	x	-	x	x
	channel 4	x	x	x	-	x	x
	channel 5	x	x	x	-	x	x
	channel 5 – channel 3	-	x	x	-	x	x
	local zenith angle	x	-	-	-?	-	-
	local azimuth angle	x	-	-	-?	x	-
						Continued on next page	

Table D.3 – continued from previous page

		IWP regressor			Ice cloud detector		
		Holl et al. (2014)	Mrziglod (2018)	This study	Holl et al. (2014)	Mrziglod (2018)	This study
	solar zenith angle	-	x	x	-	x	x
	solar azimuth angle	-	x	x	-	x	x
	lat	-	x	x	-	x	x
	scan position	-	x	x	-	-	-
AVHRR	channel 1	-	-	-	-	-	-
	channel 2	-	-	-	-	x	x
	channel 3	x	x	-	x	x	-
	channel 4	x	x	x	x	-	-
	channel 5	x	x	x	x	x	x
	std(channel 2)	-	-	-	-	x	x
	std(channel 5)	-	x	x	-	x	x
	channel 5 – channel 4	-	x	x	-	x	x
Surface elevation		x	x	x	x	x	x
Land-sea mask		-	x	x	-	-	-

Table D.3 – continued from previous page

		IWP regressor			Ice cloud detector		
		Holl et al. (2014)	Mrziglod (2018)	This study	Holl et al. (2014)	Mrziglod (2018)	This study
Surface temperature		x?	-	-	x	-	-
IWP target for cloud detection					IWP > 10 g/m^2	IWP = 0 g/m^2	IWP = 0 g/m^2

D.4   Previous SPARE-ICE IWP data records

	Holl et al. (2014)	S. Li (2015)	Mrziglod (2018)
Retrieval period	01/2007-12/2009	06/2006-07/2010 (except 2007)	01/2007-12/2008

Table D.4: SPARE-ICE IWP retrieval products created as part of studies that focused primarily on SPARE-ICE. The time periods were compiled based on the respective papers. The months of the periods given for Mrziglod (2018) and Holl et al. (2014) may be erroneous.

D.5   Satellite mission durations

Satellite	Launch	EOL
NOAA-15	13/05/1998	≥ 2024
NOAA-16	21/09/2000	09/06/2014
NOAA-17	24/06/2002	10/04/2013
NOAA-18	20/05/2005	≥ 2024
NOAA-19	04/02/2009	≥ 2024
MetOp-A	19/10/2006	15/11/2021
MetOp-B	02/04/2012	≥ 2024
MetOp-C	02/04/2016	≥ 2027
CALIPSO	28/04/2006	≥ 2023
CloudSat	28/04/2006	≥ 2025

Table D.5: Launch and end of life (EOL) dates for NOAA-15 to NOAA-19, MetOp-A to MetOp-B and CALIPSO and CloudSat as per July 23, 2023. The dates were compiled from the official World Meteorological Organization OSCAR website (https://space.oscar.wmo.int/).

Appendix E

Additional figures

This chapter contains supplementary figures for the re-training of SPARE-ICE and shows additional global maps of annually averaged IWP from the two new SPARE-ICE IWP data records and 2C-ICE IWP for 2008 and 2017.

E.1   Re-training

Performance on testing dataset
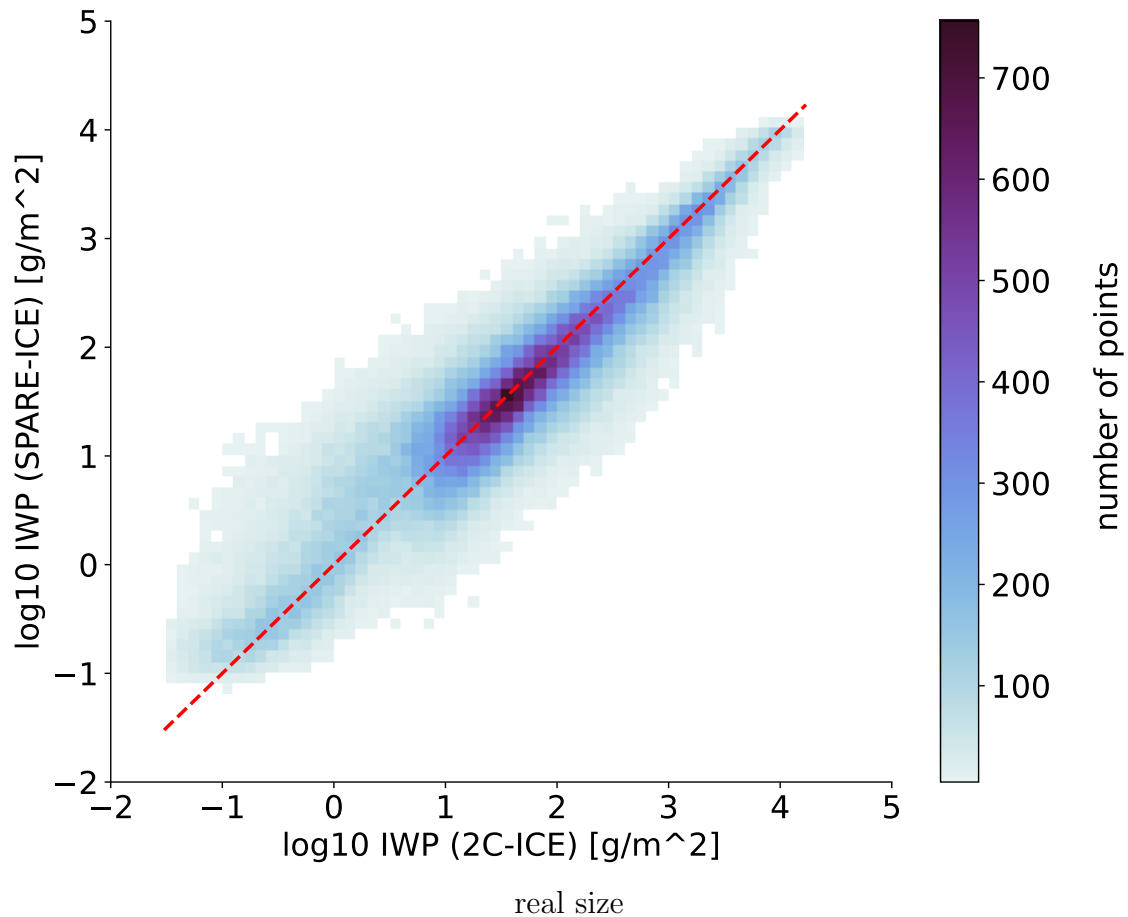
Figure E.1: Scatter density plot showing the absolute number of collocations between SPARE-ICE IWP and 2C-ICE IWP for the testing dataset *test_data*. All testing data collocations were binned according to the reference 2C-ICE IWP and SPARE-ICE IWP. The colours indicate the absolute number of collocations with a particular combination of 2C-ICE IWP and SPARE-ICE IWP. The red dashed line is the diagonal. Bins with less than five collocations are represented in white. This is in accordance with Mrziglod (2018).
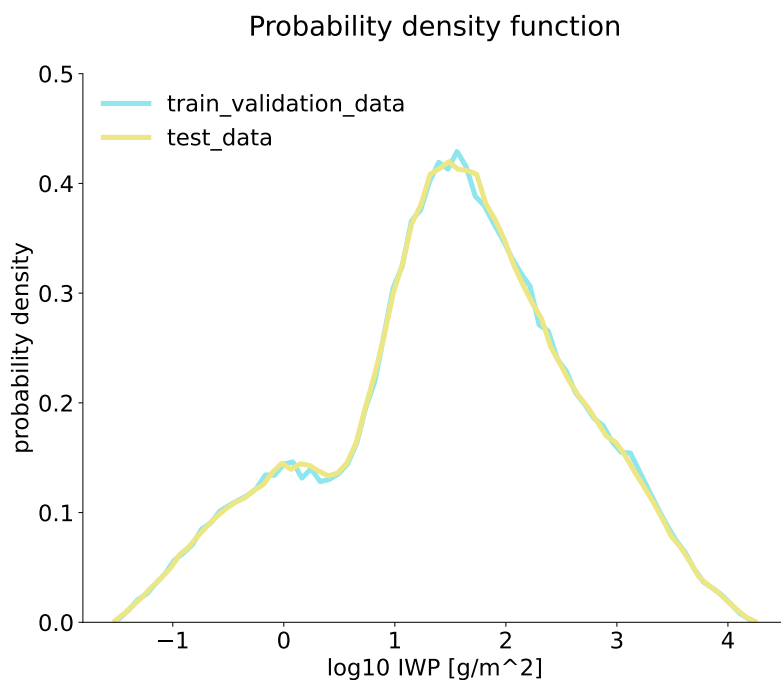
Figure E.2: Probability density functions for *test_data* and *train_validation_data* from the re-training of SPARE-ICE. Both probability density functions match very well.
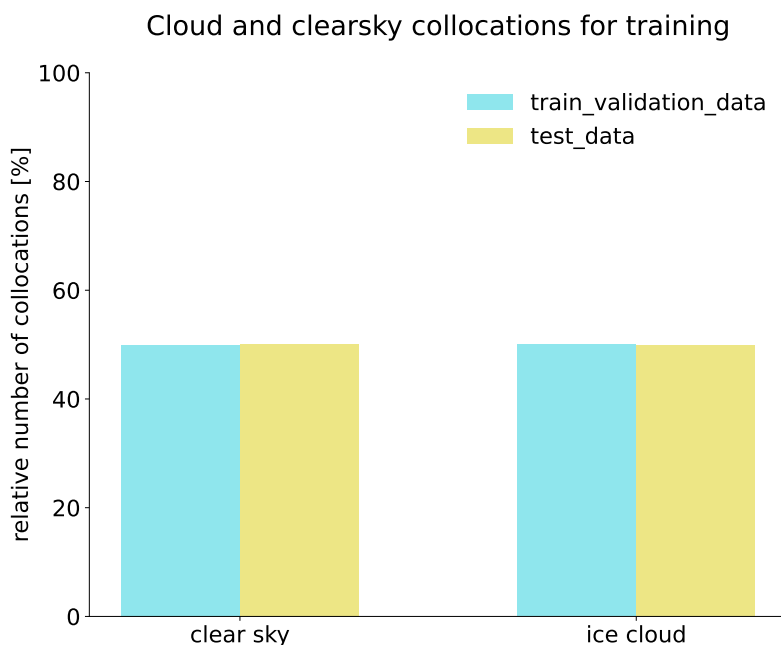
Figure E.3: Relative amount of cloud and clearsky collocation data points for *test_data* and *train_validation_data*. Both datasets contain about the same amount of cloud and clearsky data points.

E.2   Global IWP maps

2C-ICE IWP 2008
Ascending node

(a)

2C-ICE IWP 2008
Descending node

(b)
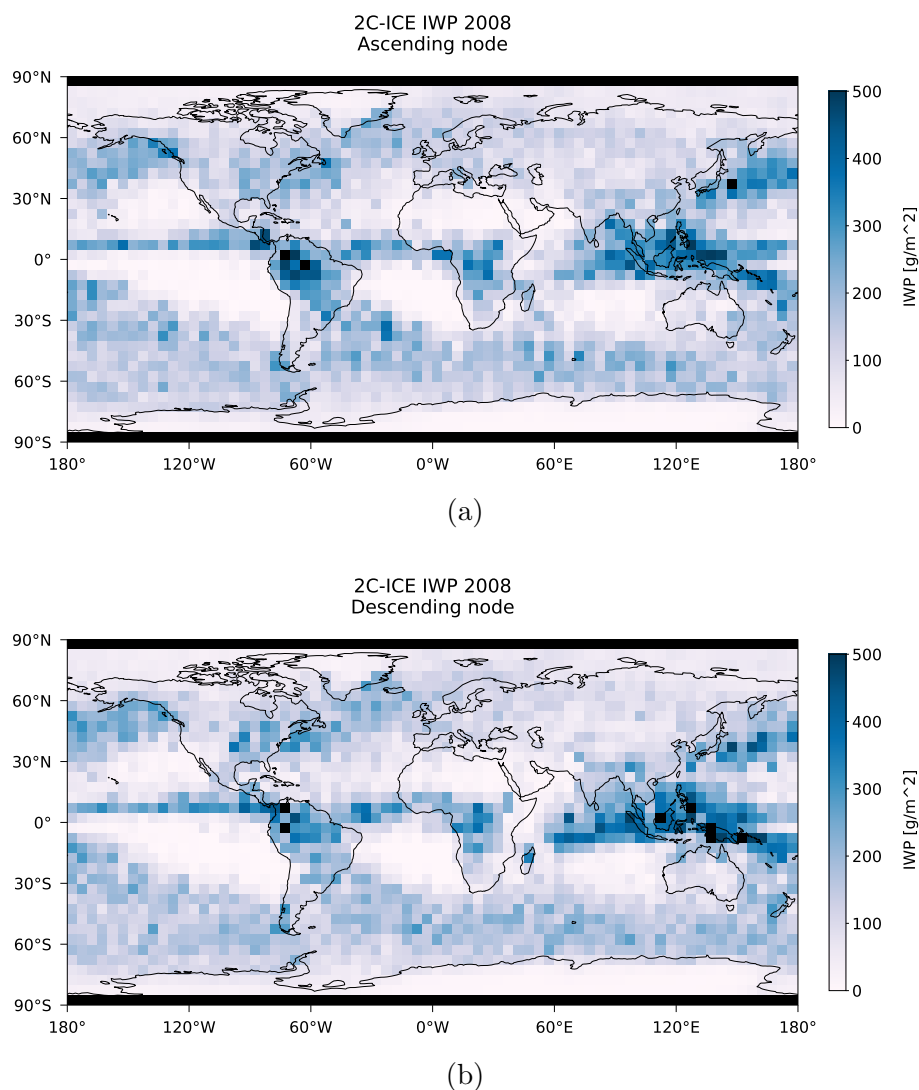
Figure E.4: Global maps of annually averaged 2C-ICE IWP for 2008 during the (a) ascending nodes and (b) descending nodes of CloudSat. The data is averaged over 5° x 5° grid cells. 2C-ICE was quality checked and cleared of MODPs via the pre-processing toolkit. Grid cells for which no data was available are shown in black.

(a)

(b)
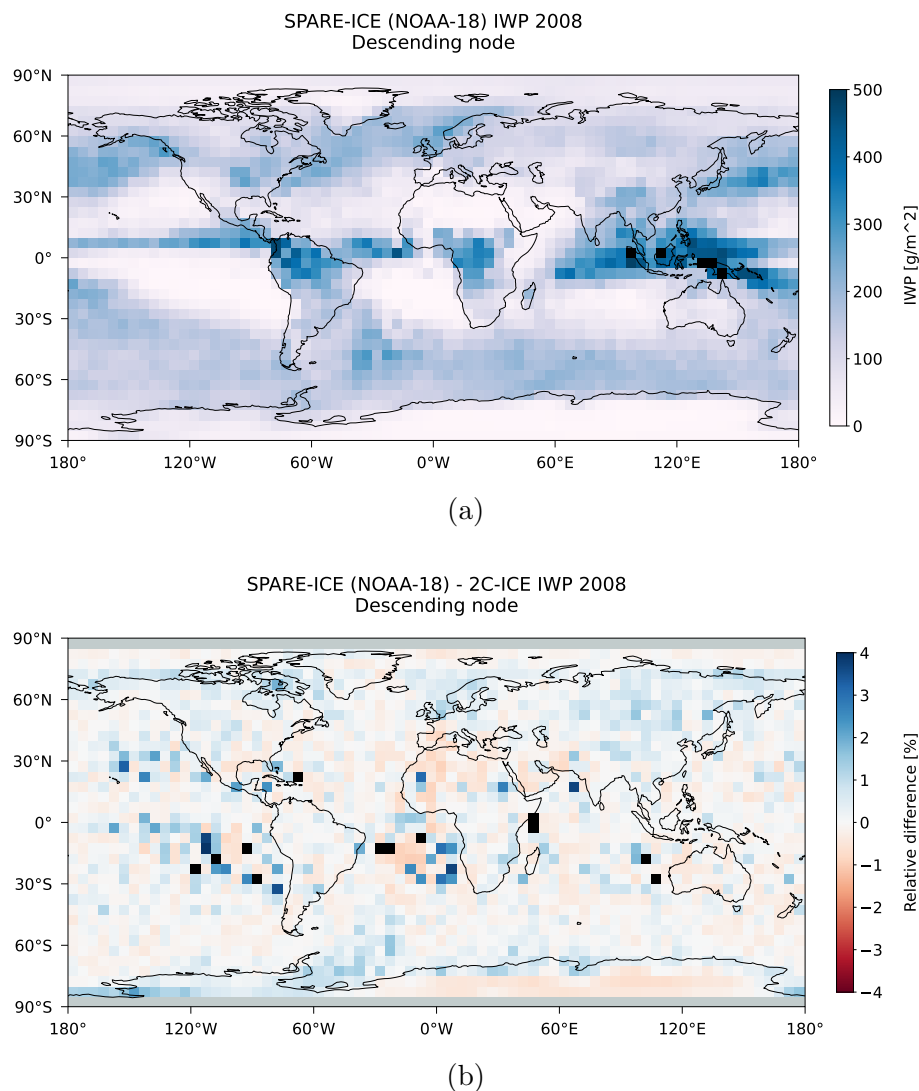
Figure E.5: Global maps of (a) annually averaged IWP from the NOAA-18 based SPARE-ICE retrieval, and (b) the relative difference between annually mean IWP values of the NOAA-18 based SPARE-ICE IWP retrieval product and 2C-ICE IWP for 2008 during the descending nodes of NOAA-18 and CloudSat (as representative for 2C-ICE). The data is averaged over 5° x 5° grid cells. 2C-ICE was quality checked and cleared of MODPs via the pre-processing toolkit before calculating the relative difference. Areas where no values from 2C-ICE were available are indicated in grey.

(a)

(b)
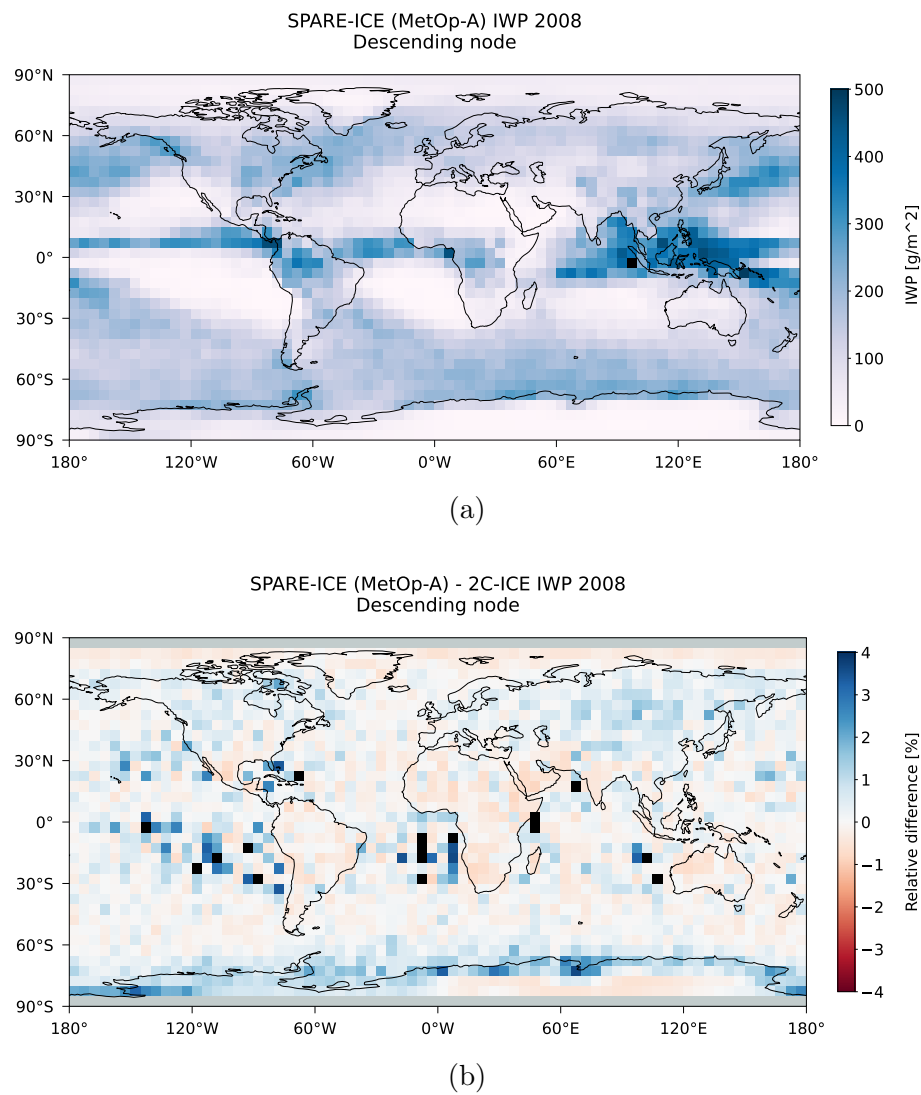
Figure E.6: As in Figure E.9, but for the MetOp-A based SPARE-ICE IWP data record.

2C-ICE IWP 2017
Ascending node
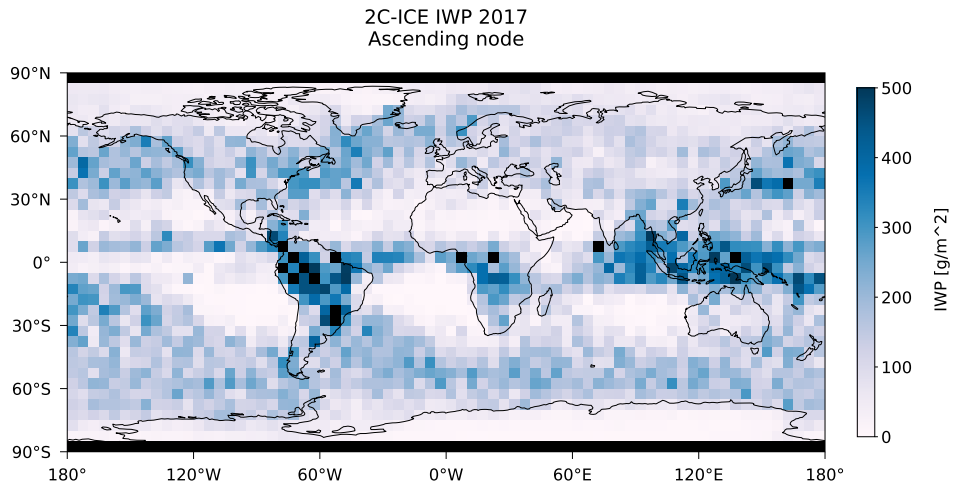
Figure E.7: As in Figure E.4, but for the ascending nodes of CloudSat during 2017.

SPARE-ICE (NOAA-18) IWP 2017
Ascending node

(a)

SPARE-ICE (NOAA-18) - 2C-ICE IWP 2017
Ascending node
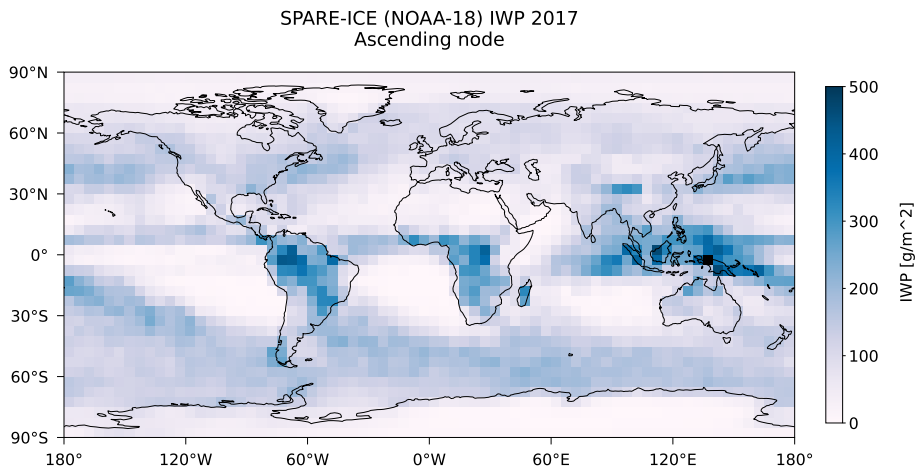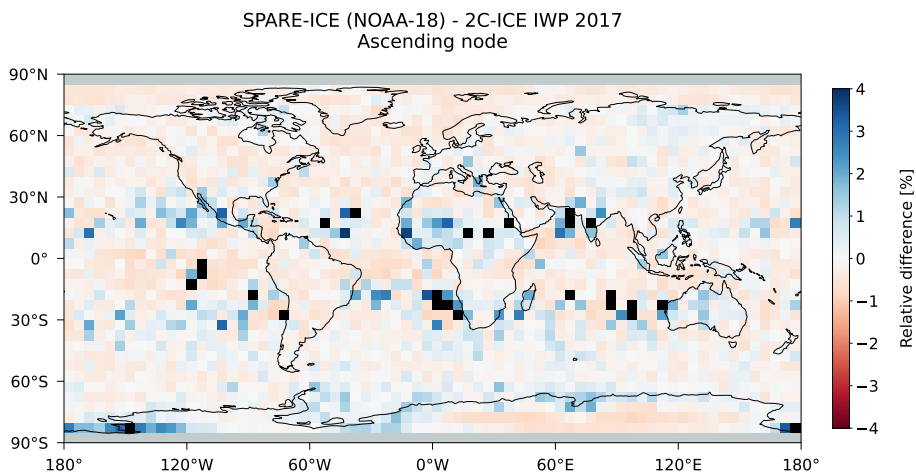
(b)

Figure E.8: As in Figure E.9, but for the ascending nodes of NOAA-18 and CloudSat during 2017.

(a)

(b)
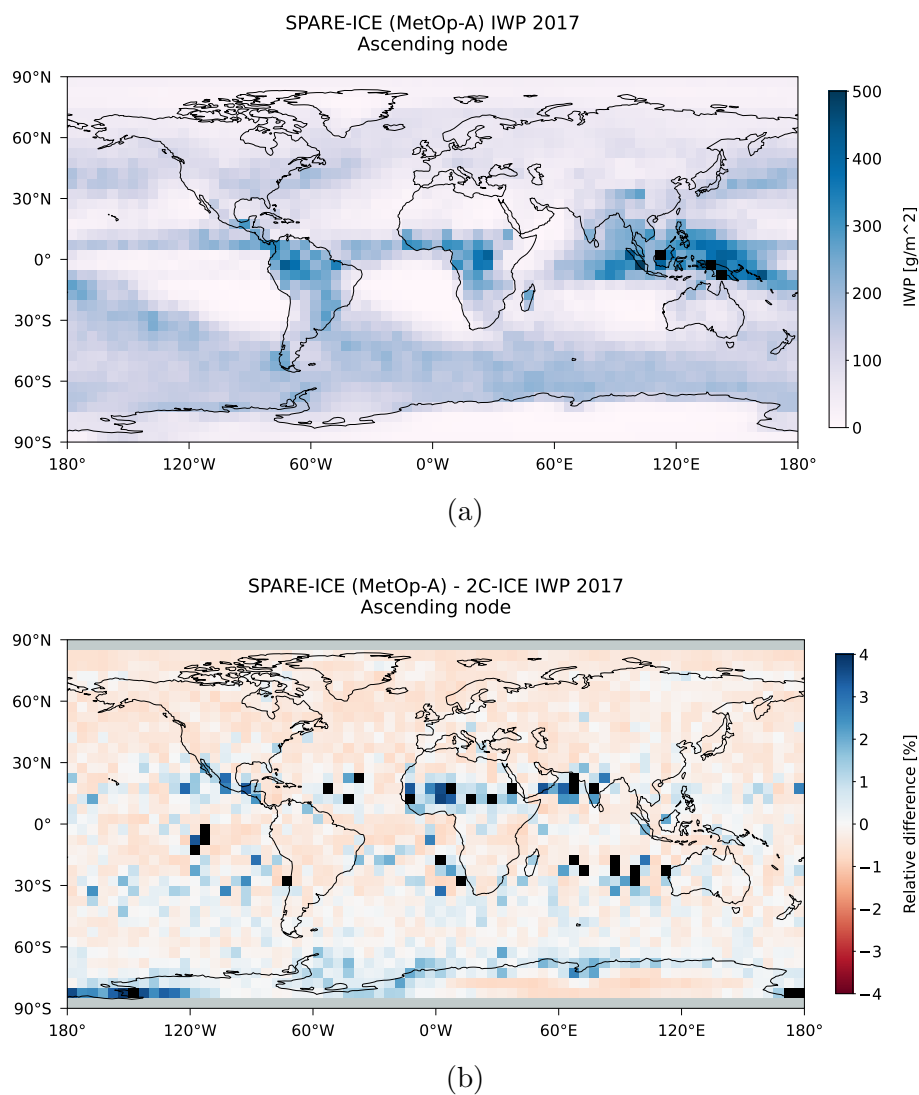
Figure E.9: As in Figure E.9, but for the ascending nodes of MetOp-A and CloudSat during 2017.

References

Aapp documentation - data formats (Tech. Rep.). (2021, August). Retrieved from `https://nwp-saf.eumetsat.int/site/download/documentation/aapp/NWPSAF-MF-UD-003_Formats_v8.1.pdf`

Amante, C. and Eakins, B. W. (2009). Etopo1 arc-minute global relief model: procedures, data sources and analysis. , *NOAA Technical Memorandum NESDIS NGDC-24. National Geophysical Data Center, NOAA*. doi: 10.7289/V5C8276M

AVHRR Level 1b Product Guide [Computer software manual]. (2011, January).

Azad, M. and Moshkov, M. (2023). Applications of depth minimization of decision trees containing hypotheses for multiple-value decision tables. *Entropy*, *25*(4), 547.

Boain, R. J. (2004). A-b-cs of sun-synchronous orbit mission design.

Bonsignori, R. (2007). The Microwave Humidity Sounder (MHS): in-orbit performance assessment. In S. Habib, R. Meynart, S. P. Neeck and H. Shimoda (Eds.), *Sensors, systems, and next-generation satellites xi* (Vol. 6744, p. 67440A). SPIE. Retrieved from `https://doi.org/10.1117/12.737986` doi: 10.1117/12.737986

Braga, R. C. and Vila, D. A. (2014). Investigating the ice water path in convective cloud life cycles to improve passive microwave rainfall retrievals. *Journal of Hydrometeorology*, *15*(4), 1486–1497.

Brath, M., Fox, S., Eriksson, P., Harlow, R. C., Burgdorf, M. and Buehler, S. A. (2018). Retrieval of an ice water path over the ocean from ismar and marss millimeter and submillimeter brightness temperatures. *Atmospheric Measurement Techniques*, *11*(1), 611–632.

Braun, B. M., Sweetser, T. H., Graham, C. and Bartsch, J. (2019). Cloudsat's a-train exit and the formation of the c-train: An orbital dynamics perspective. In *2019 ieee aerospace conference* (pp. 1–10).

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*, 123–140.

Buehler, S. A., Jiménez, C., Evans, K. F., Eriksson, P., Rydberg, B., Heymsfield, A., ... Davis, P. (2007). A concept for a satellite mission to measure cloud

ice water path, ice particle size, and cloud altitude. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, *133*(S2), 109–128.

Carter, J., Schmid, K., Waters, K., Betzhold, L., Hadley, B., Mataosky, R. and Halleran, J. (2012). An introduction to lidar technology, data, and applications. *NOAA Coastal Services Center*, *2*, 17.

Charbuty, B. and Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, *2*(01), 20–28.

Chen, D. and Jain, R. (1994). A robust backpropagation learning algorithm for function approximation. *IEEE Transactions on Neural Networks*, *5*(3), 467-479. doi: 10.1109/72.286917

Chen, T., Rossow, W. B. and Zhang, Y. (2000). Radiative effects of cloud-type variations. *Journal of climate*, *13*(1), 264–286.

Cimini, D., Serio, C., Masiello, G., Mastro, P., Ricciardelli, E., Di Paola, F., ... Romano, F. (2023). Spectrum synergy for investigating cloud microphysics. *Bulletin of the American Meteorological Society*, *104*(3), E606–E622.

Courtois, A., Morel, J.-M. and Arias, P. (2023). Can neural networks extrapolate? discussion of a theorem by pedro domingos. *Revista de la Real Academia de Ciencias Exactas, Físicas y Naturales. Serie A. Matemáticas*, *117*(2), 79.

Cracknell, A. P. (1997). *Advanced very high resolution radiometer avhrr*. Crc Press.

Dai, A. (2001). Global precipitation and thunderstorm frequencies. part ii: Diurnal variations. *Journal of Climate*, *14*(6), 1112–1128.

da Silva, I. N., Hernane Spatti, D., Andrade Flauzino, R., Liboni, L. H. B. and dos Reis Alves, S. F. (2017). Multilayer perceptron networks. In *Artificial neural networks : A practical course* (pp. 55–115). Cham: Springer International Publishing. Retrieved from `https://doi.org/10.1007/978-3-319-43162-8_5` doi: 10.1007/978-3-319-43162-8_5

Davini, D., Samineni, B., Thomas, B., Tran, A. H., Zhu, C., Ha, K., ... White, L. (2021). Using physics-informed regularization to improve extrapolation capabilities of neural networks. In *Fourth workshop on machine learning and the physical sciences (neurips 2021)*.

Delanoë, J. and Hogan, R. J. (2008). A variational scheme for retrieving ice cloud properties from combined radar, lidar, and infrared radiometer. *Journal of Geophysical Research: Atmospheres*, *113*(D7).

Deng, M., Mace, G. G., Wang, Z. and Lawson, R. P. (2013). Evaluation of several a-train ice cloud retrieval products with in situ measurements collected during the sparticus campaign. *Journal of applied meteorology and climatology*, *52*(4), 1014–1030.

Deng, M., Mace, G. G., Wang, Z. and Okamoto, H. (2010). Tropical composition, cloud and climate coupling experiment validation for cirrus cloud profiling retrieval using cloudsat radar and calipso lidar. *Journal of Geophysical Research: Atmospheres*, *115*(D10).

Dolatshah, M., Hadian, A. and Minaei-Bidgoli, B. (2015). *Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces* (Tech. Rep.).

Dou, T., Xiao, C., Huang, Y., Yue, H. and Han, W. (2020). Estimation of the atmospheric ice content mass, spatial distribution, and long-term changes based on the era5 reanalysis. *Geophysical Research Letters*, *47*(15), e2020GL088186. Retrieved from `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020GL088186` (e2020GL088186 2020GL088186) doi: https://doi.org/10.1029/2020GL088186

Dumitru, C. and Maria, V. (2013). Advantages and disadvantages of using neural networks for predictions. *Ovidius University Annals, Series Economic Sciences*, *13*(1).

Duncan, D. I. and Eriksson, P. (2018). An update on global atmospheric ice estimates from satellite observations and reanalyses. *Atmospheric Chemistry and Physics*, *18*(15), 11205–11219.

Eldredge, N. (2002). *Life on earth : An encyclopedia of biodiversity, ecology, and evolution.* ABC-CLIO.

Eliasson, S., Buehler, S., Milz, M., Eriksson, P. and John, V. (2011). Assessing observed and modelled spatial distributions of ice water path using satellite data. *Atmospheric Chemistry and Physics*, *11*(1), 375–391.

Elizondo, D., Hoogenboom, G. and McClendon, R. (1994). Development of a neural network model to predict daily solar radiation. *Agricultural and forest meteorology*, *71*(1-2), 115–132.

Ellis, T. D., L'Ecuyer, T., Haynes, J. M. and Stephens, G. L. (2009). How often does it rain over the global oceans? the perspective from cloudsat. *Geophysical Research Letters*, *36*(3).

Ewald, F., Groß, S., Wirth, M., Delanoë, J., Fox, S. and Mayer, B. (2021). Why we need radar, lidar, and solar radiance observations to constrain ice cloud microphysics. *Atmospheric Measurement Techniques*, *14*(7), 5029–5047.

Faure, T., Isaka, H. and Guillemet, B. (2002). Neural network retrieval of cloud parameters from high-resolution multispectral radiometric data: A feasibility study. *Remote sensing of environment*, *80*(2), 285–296.

Feurer, M. and Hutter, F. (2019). Hyperparameter optimization. In F. Hutter, L. Kotthoff and J. Vanschoren (Eds.), *Automated machine learning: Methods, systems, challenges* (pp. 3–33). Cham: Springer International Publishing.

Retrieved from `https://doi.org/10.1007/978-3-030-05318-5_1`   doi: 10.1007/978-3-030-05318-5_1

Galanaki, E., Lagouvardos, K., Kotroni, V., Flaounas, E. and Argiriou, A. (2018). Thunderstorm climatology in the mediterranean using cloud-to-ground lightning observations. *Atmospheric Research*, *207*, 136–144.

Gardner, M. W. and Dorling, S. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, *32*(14-15), 2627–2636.

Gayet, J.-F., Ovarlez, J., Shcherbakov, V., Ström, J., Schumann, U., Minikin, A., . . . Monier, M. (2004). Cirrus cloud microphysical and optical properties at southern and northern midlatitudes during the inca experiment. *Journal of Geophysical Research: Atmospheres*, *109*(D20).

Gupta, S. and Gupta, A. (2019). Dealing with noise problem in machine learning data-sets: A systematic review. *Procedia Computer Science*, *161*, 466–474.

Hagenlocher, P. (n.d.). Decision tree learning..

Håkansson, N., Adok, C., Thoss, A., Scheirer, R. and Hörnquist, S. (2018). Neural network cloud top pressure and height for modis. *Atmospheric Measurement Techniques*, *11*(5), 3177–3196.

Haley, P. J. and Soloway, D. (1992). Extrapolation limitations of multilayer feedforward neural networks. In *[proceedings 1992] ijcnn international joint conference on neural networks* (Vol. 4, pp. 25–30).

Hang, Y., L'Ecuyer, T. S., Henderson, D. S., Matus, A. V. and Wang, Z. (2019). Reassessing the effect of cloud type on earth's energy balance in the age of active spaceborne observations. part ii: Atmospheric heating. *Journal of Climate*, *32*(19), 6219–6236.

Hartmann, D. L., Ockert-Bell, M. E. and Michelsen, M. L. (1992). The effect of cloud type on earth's energy balance: Global analysis. *Journal of Climate*, *5*(11), 1281–1304.

Hasan, R. and Chu, C. (2022). Noise in datasets: What are the impacts on classification performance? *Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods*. Retrieved from `https://par.nsf.gov/biblio/10319530` doi: 10.5220/0010782200003122

Hauska, H. and Swain, P. H. (1975). The decision tree classifier: design and potential. In *Lars symposia* (p. 45).

Heidinger, A. K., Foster, M. J., Walther, A. and Zhao, X. T. (2014). The pathfinder atmospheres–extended avhrr climate dataset. *Bulletin of the American Meteorological Society*, *95*(6), 909 - 922. Retrieved from `https://journals.ametsoc.org/view/journals/bams/95/6/bams-d-12-00246.1.xml` doi: https://doi.org/10.1175/BAMS-D-12-00246.1

Heymsfield, A. J., Protat, A., Bouniol, D., Austin, R. T., Hogan, R. J., Delanoë, J., ... Wang, Z. (2008). Testing iwc retrieval methods using radar and ancillary measurements with in situ data. *Journal of Applied Meteorology and Climatology*, *47*(1), 135–163.

Hirabayashi, Y., Kanae, S., Motoya, K., Masuda, K. and Döll, P. (2008). A 59-year (1948-2006) global meteorological forcing data set for land surface models. part ii: Global snowfall estimation. *Hydrological Research Letters*, *2*, 65–69.

Hodge, V. and Austin, J. (2004). A survey of outlier detection methodologies. *Artificial intelligence review*, *22*, 85–126.

Holl, G., Eliasson, S., Mendrok, J. and Buehler, S. A. (2014). Spare-ice: Synergistic ice water path from passive operational sensors. *Journal of Geophysical Research: Atmospheres*, *119*(3), 1504-1523. Retrieved from `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2013JD020759` doi: https://doi.org/10.1002/2013JD020759

Hong, G., Heygster, G., Miao, J. and Kunzi, K. (2005). Detection of tropical deep convective clouds from amsu-b water vapor channels measurements. *Journal of Geophysical Research: Atmospheres*, *110*(D5).

Hong, Y. and Liu, G. (2015). The characteristics of ice cloud properties derived from cloudsat and calipso measurements. *Journal of Climate*, *28*(9), 3880–3901.

Islam, T. and Srivastava, P. K. (2015). Synergistic multi-sensor and multi-frequency retrieval of cloud ice water path constrained by cloudsat collocations. *Journal of Quantitative Spectroscopy and Radiative Transfer*, *161*, 21–34.

King, M. D., Tsay, S.-C., Platnick, S. E., Wang, M. and Liou, K.-N. (1997). Cloud retrieval algorithms for modis: Optical thickness, effective particle radius, and thermodynamic phase. *MODIS Algorithm Theoretical Basis Document*, *1997*.

Klaes, D. and Holmlund, K. (2007). The eps/metop system: Overview and first results. In *Proceedings of the joint 2007 eumetsat meteorological satellite conference and the 15th satellite meteorology & oceanography conference of the american meteorological society, amsterdam, the netherlands* (pp. 24–28).

Kox, S., Bugliaro, L. and Ostler, A. (2014). Retrieval of cirrus cloud optical thickness and top altitude from geostationary remote sensing. *Atmospheric Measurement Techniques*, *7*(10), 3233–3246.

Laviola, S., Levizzani, V., Ferraro, R. R. and Beauchamp, J. (2020). Hailstorm detection by satellite microwave radiometers. *Remote Sensing*, *12*(4), 621.

Li, L., Durden, S. and Tanelli, S. (2017, September). *Level 1 b cpr process description and interface control document* (Tech. Rep.).

Li, S. (2015). *Spare-ice satellite cloud ice retrievals* (Unpublished master's thesis). Universität Hamburg, School of Integrated Climate System Science (SICSS) Hamburg.

Liu, T., Moore, A. and Gray, A. (2006). New algorithms for efficient high dimensional non-parametric classification. *Journal of Machine Learning Research*, *7*.

Liu, W., Chawla, S., Cieslak, D. A. and Chawla, N. V. (2010). A robust decision tree algorithm for imbalanced data sets. In *Proceedings of the 2010 siam international conference on data mining* (pp. 766–777).

L'Ecuyer, T. S., Hang, Y., Matus, A. V. and Wang, Z. (2019). Reassessing the effect of cloud type on earth's energy balance in the age of active spaceborne observations. part i: Top of atmosphere and surface. *Journal of Climate*, *32*(19), 6197–6217.

Mace, G. and Deng, M. (2015). *Level 2c cloudsat-calipso combined ice cloud property retrieval product process description document* (Tech. Rep.).

Mace, G. and Deng, M. (2019). *Level 2 cloudsat-calipso combined ice cloud property retrieval product process description and interface control document* (Tech. Rep.).

Mas, J. F. and Flores, J. J. (2008). The application of artificial neural networks to the analysis of remotely sensed data. *International Journal of Remote Sensing*, *29*(3), 617–663.

Mastro, P., Masiello, G., Serio, C., Cimini, D., Ricciardelli, E., Di Paola, F., … Romano, F. (2022). Combined iasi-ng and mws observations for the retrieval of cloud liquid and ice water path: a deep learning artificial intelligence approach. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *15*, 3313–3322.

Minnis, P., Hong, G., Sun-Mack, S., Smith Jr, W. L., Chen, Y. and Miller, S. D. (2016). Estimating nocturnal opaque ice cloud optical depth from modis multispectral infrared radiances using a neural network method. *Journal of Geophysical Research: Atmospheres*, *121*(9), 4907–4932.

Minnis, P., Sun-Mack, S., Smith Jr, W. L., Hong, G. and Chen, Y. (2019). Advances in neural network detection and retrieval of multilayer clouds for ceres using multispectral satellite data. In *Remote sensing of clouds and the atmosphere xxiv* (Vol. 11152, p. 1115202).

Monique F., K. and Kerin, M. R. (2018). Data quality: "garbage in – garbage out". *Health Information Management Journal*, *47*(3), 103-105. Retrieved from https://doi.org/10.1177/1833358318774357 (PMID: 29719995) doi: 10.1177/1833358318774357

Montmessin, F., Forget, F., Rannou, P., Cabane, M. and Haberle, R. M. (2004). Origin and role of water ice clouds in the martian water cycle as inferred from a general circulation model. *Journal of Geophysical Research: Planets*, *109*(E10).

Mrziglod, J. (2018). *Using collocated satellite data for ice water path retrieval—a reimplementation of spare-ice* (Master's thesis). Retrieved from `https://arts .mi .uni -hamburg .de /publications /grouppapers2 /mrziglod18 _using _msc.pdf`

Ninomiya, Y. (1995). Quantitative estimation of sio/sub 2/content in igneous rocks using thermal infrared spectra with a neural network approach. *IEEE Transactions on Geoscience and Remote Sensing*, *33*(3), 684–691.

NOAA KLM User's guide (Tech. Rep.). (2014, August). National Oceanic and Atmospheric Administration, National Environmental Satellite, Data, and Information Service. Retrieved from `https://www.star .nesdis .noaa.gov/ mirs/documents/0.0_NOAA_KLM_Users_Guide.pdf` (Revision)

Omohundro, S. M. (1989). *Five balltree construction algorithms.* International Computer Science Institute Berkeley.

Pakarinen, O., Pulido Lamas, C., Roudsari, G., Reischl, B. and Vehkamäki, H. (2021). The role of latent heat in heterogeneous ice nucleation. In *Egu general assembly conference abstracts* (pp. EGU21–16164).

Pal, M. (2005). Random forest classifier for remote sensing classification. *International journal of remote sensing*, *26*(1), 217–222.

Quinlan, J. R. (1996). Learning decision tree classifiers. *ACM Computing Surveys (CSUR)*, *28*(1), 71–72.

Ramanathan, V., Cess, R., Harrison, E., Minnis, P., Barkstrom, B., Ahmad, E. and Hartmann, D. (1989). Cloud-radiative forcing and climate: Results from the earth radiation budget experiment. *Science*, *243*(4887), 57–63.

Ramchoun, H., Ghanou, Y., Ettaouil, M. and Janati Idrissi, M. A. (2016). Multilayer perceptron: Architecture optimization and training.

Ringel, M. (2023). *A neural-network icewater path retrievalfor airborne microwave observations* (Unpublished master's thesis). Universität Hamburg.

Robel, J. and Graumann, A. (2009). *NOAA KLM User's guide* (Tech. Rep.).

Romano, F., Cimini, D., Rizzi, R. and Cuomo, V. (2007). Multilayered cloud parameters retrievals from combined infrared and microwave satellite observations. *Journal of Geophysical Research: Atmospheres*, *112*(D8).

Rossow, W., Walker, A., Golea, V., Knapp, K. R., Young, A., A., I., ... NOAA's Climate Data Record Program (2016). International satellite cloud climatology project climate data record, h-series [Computer software manual].

Rusiecki, A., Kordos, M., Kamiński, T. and Greń, K. (2014). Training neural networks on noisy data. In *Artificial intelligence and soft computing: 13th international conference, icaisc 2014, zakopane, poland, june 1-5, 2014, proceedings, part i 13* (pp. 131–142).

Rustamov, R. B., Hasanova, S. and Zeynalova, M. H. (2018). *Multi-purposeful*

application of geospatial data. BoD–Books on Demand.

Safavian, S. R. and Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, *21*(3), 660–674.

Santos, C. F. G. D. and Papa, J. P. (2022). Avoiding overfitting: A survey on regularization methods for convolutional neural networks. *ACM Computing Surveys (CSUR)*, *54*(10s), 1–25.

Sawaf, F. and Groves, R. M. (2013). Statistically guided improvements in speckle phase discontinuity predictions by machine learning systems. *Optical Engineering*, *52*(10), 101907–101907.

Serneels, S. and Verdonck, T. (2008). Principal component analysis for data containing outliers and missing elements. *Computational Statistics & Data Analysis*, *52*(3), 1712–1727.

Song, Y.-Y. and Ying, L. (2015). Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, *27*(2), 130.

Stephens, G., Winker, D., Pelon, J., Trepte, C., Vane, D., Yuhas, C., ... Lebsock, M. (2018). Cloudsat and calipso within the a-train: Ten years of actively observing the earth system. *Bulletin of the American Meteorological Society*, *99*(3), 569–581.

Stephens, G. L., Vane, D. G., Boain, R. J., Mace, G. G., Sassen, K., Wang, Z., ... the CloudSat Science Team (2002). The cloudsat mission and the a-train: A new dimension of space-based observations of clouds and precipitation. *Bulletin of the American Meteorological Society*, *83*(12), 1771–1790.

Strandgren, J., Bugliaro, L., Sehnke, F. and Schröder, L. (2017). Cirrus cloud retrieval with msg/seviri using artificial neural networks. *Atmospheric Measurement Techniques*, *10*(9), 3547–3573.

Tu, J. V. (1996). Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology*, *49*(11), 1225–1231.

Vasudevan, B. G., Gohil, B. S. and Agarwal, V. K. (2004). Backpropagation neural-network-based retrieval of atmospheric water vapor and cloud liquid water from irs-p4 msmr. *IEEE transactions on geoscience and remote sensing*, *42*(5), 985–990.

Waliser, D. E., Li, J.-L. F., Woods, C. P., Austin, R. T., Bacmeister, J., Chern, J., ... Wang, Z. W. (2009). Cloud ice: A climate model challenge with signs and expectations of progress. *Journal of Geophysical Research: Atmospheres*, *114*(D8).

Wang, W., Wang, Z., He, Q. and Zhang, L. (2022). Retrieval of ice water path from the microwave humidity sounder (mwhs) aboard fengyun-3b (fy-3b) satellite

polarimetric measurements based on a deep neural network. *Atmospheric Measurement Techniques*, *15*(21), 6489–6506.

Wilkinson, L. (2004). Classification and regression trees. *Systat*, *11*, 35–56.

Winker, D. M., Hunt, W. H. and McGill, M. J. (2007). Initial performance assessment of caliop. *Geophysical Research Letters*, *34*(19). Retrieved from `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2007GL030135` doi: https://doi.org/10.1029/2007GL030135

Winker, D. M., Pelon, J. R. and McCormick, M. P. (2003). CALIPSO mission: spaceborne lidar for observation of aerosols and clouds. In *Lidar remote sensing for industry and environment monitoring iii* (Vol. 4893, pp. 1–11).

Wunderle, S. and Neuhaus, C. (2020). AVHRR Master Data Set Handbook (DEL-16) [Computer software manual]. Hallerstrasse 12, CH-2012 Bern.

Xiong, X., Butler, J., Cao, C. and Wu, X. (2018). 1.13 - optical sensors—vis/nir/swir. In S. Liang (Ed.), *Comprehensive remote sensing* (p. 353-375). Oxford: Elsevier. Retrieved from `https://www.sciencedirect.com/science/article/pii/B9780124095489103252` doi: https://doi.org/10.1016/B978-0-12-409548-9.10325-2

Yang, L. and Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, *415*, 295-316. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0925231220311693` doi: https://doi.org/10.1016/j.neucom.2020.07.061

Zängl, G., Reinert, D., Rípodas, P. and Baldauf, M. (2015). The icon (icosahedral non-hydrostatic) modelling framework of dwd and mpi-m: Description of the non-hydrostatic dynamical core. *Quarterly Journal of the Royal Meteorological Society*, *141*(687), 563–579.

Zhao, F., Tang, C., Tian, X., Wu, X., Dai, C. and Wei, H. (2023). The global spatial and temporal distribution of ice cloud optical thickness based on modis satellite data during 2000–2021. *Atmosphere*, *14*(6), 977.

Zhuang, H., Wang, X., Bendersky, M. and Najork, M. (2020). Feature transformation for neural ranking models. In *Proceedings of the 43rd international acm sigir conference on research and development in information retrieval* (pp. 1649–1652).

Zipser, E. J., Cecil, D. J., Liu, C., Nesbitt, S. W. and Yorty, D. P. (2006). Where are the most intense thunderstorms on earth? *Bulletin of the American Meteorological Society*, *87*(8), 1057–1072.

Acknowledgements

Affirmation in lieu of oath/Versicherung an Eides statt

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit im Studiengang M. Sc. Meteorologie selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht. Einer Veröffentlichung der vorliegenden Arbeit in der zuständigen Fachbibliothek des Fachbereichs stimme ich nicht zu.

21.08.2023, Lorena Kowalczyk

Datum, Unterschrift