Thèse présentée pour obtenir le grade de

**DOCTEUR de SORBONNE UNIVERSITÉ**

Specialité

Informatique

École Doctorale

Informatique, Télécommunications et Électronique (Paris)

# Deep learning for sea surface height reconstruction from multi-variate satellite observations

*Apprentissage profond pour reconstruire la hauteur de la surface océanique à partir d'observations satellites multivariées*


Présentée par

**Théo ARCHAMBAULT**

Thèse dirigée par

Dominique BÉRÉZIAT

Co-encadrée par

Anastase Alexandre CHARANTONIS


Soutenue le **4 octobre 2024**, devant un jury composé de :

| | |
|---|---|
| **Alexander BARTH**, Maître de recherches FNRS, Université de Liège | *Rapporteur* |
| **Emmanuel COSME**, Professeur assistant, Université Grenoble Alpes | *Rapporteur* |
| **Cécile MALLET**, Professeure, LATMOS | *Présidente du jury* |
| **Claire MONTELEONI**, Directrice de recherche, INRIA Paris | *Examinatrice* |
| **Alexandre STEGNER**, Professeur, CNRS - Ecole Polytechnique - Amphitrite | *Examinateur* |
| **Maxime BALLAROTTA**, Docteur, CLS-Groupe | *Invité* |
| **Anastase CHARANTONIS**, Professeur associé, ENSIIE, LaMME | *Co-encadrant* |
| **Dominique BÉRÉZIAT**, Maître de conférences, LIP6 | *Directeur de thèse* |

# Résumé

**Titre** : *Apprentissage profond pour reconstruire la hauteur de la surface océanique à partir d'observations satellites multivariées*
**Mots clés** : apprentissage profond, télédétection par satellite, océanographie

**Résumé** : Cette thèse de doctorat porte sur la reconstruction d'images satellites de la surface de l'océan à partir de mesures éparses et bruitées. Son objectif est l'estimation de la hauteur de la mer (SSH), une variable importante pour approximer les courants de surface. Elle est actuellement mesurée par des altimètres pointant au nadir, laissant de nombreuses zones non observées. Les cartes complètes de SSH sont produites en utilisant des interpolations optimales linéaires présentant une faible résolution effective. D'autre part, la température de surface de la mer (SST) est observée sur des zones plus étendues et est physiquement liée aux courants géostrophiques à travers l'advection.
Cette thèse explore les algorithmes d'apprentissage profond pour estimer les champs de SSH. En s'appuyant sur des années de données de simulation et d'observations, les réseaux neuronaux profonds sont capables d'apprendre des relations complexes entre les variables SSH et SST. Nous utilisons ces algorithmes ainsi que les observations de température, pour reconstruire la SSH d'abord dans une perspective de réduction d'échelle sur une simulation physique. Ensuite, nous considèrerons le problème de son interpolation sur des données de simulation et d'observation, en nous concentrant particulièrement sur la manière de transférer l'apprentissage dans des contextes opérationnels. Enfin, nous adaptons notre méthode pour produire des estimations en temps réel et des prévisions.

## Abstract

**Title**: *Deep learning for sea surface height reconstruction from multi-variate satellite observations*

**Keywords**: deep learning, satellite remote sensing, ocean science

**Abstract**: This Ph.D. thesis focuses on reconstructing satellite images of the ocean surface from sparse and noisy measurements. Our objective is the Sea Surface Height (SSH), an important variable to estimate surface currents. It is retrieved through nadir-pointing altimeters, leaving important observation gaps due to their remote sensing technology. Complete SSH maps are produced using linear Optimal Interpolations with low effective resolution. On the other hand, Sea Surface Temperature (SST) products have much higher data coverage, and SST is physically linked to geostrophic currents through advection.

This thesis explores deep learning algorithms to estimate SSH fields. Relying on years of data from simulation and observations, deep neural networks are able to learn complex relationships between SSH and SST variables. Using these algorithms and SST observations, we first enhance SSH mapping from a downscaling perspective on a physical simulation. Then, we tackle the SSH interpolation problem on simulation and observation data, with a particular focus on how to transfer the learning in operational settings. Finally, we adapt our method to produce near real-time and forecast estimations.

# Remerciements

Cette thèse n'aurait jamais été possible sans le soutien d'un grand nombre de personnes que je tiens à remercier.

Tout d'abord ma famille; maman, papa, petite soeur, je veux vous dire que je vous dois tout et que je ne sais comment vous le rendre. Je suis fier de l'amour et de l'éducation que vous m'avez donnée. Merci à tous les professeurs qui s'ignorent et qui m'ont appris à être curieux. À mes grands parents qui m'ont appris à aimer la science, et à toutes les personnes avec qui j'aime tant en parler, Steve, Pierre, Brigitte, Nathalie...

Merci à mes encadrants, Dominique et Anastase avec qui j'ai tellement aimé travailler, vous avez su me conseiller et me soutenir tout en me laissant la liberté nécessaire pour passer du *supervised* au *self-supervised*. Vous êtes le duo idéal que tout étudiant rêve d'avoir. Merci à Arthur, mon grand frère académique, qui a porté ce titre comme peu de personnes le peuvent. Merci d'avoir pris tout ce temps pour discuter et rigoler, avec toi ces trois années sont passées trop vite. J'ai aussi une pensée pour Pierre, je sais que tu écriras une belle thèse au sein de notre équipe, et j'ai hâte de continuer de travailler avec toi.

Merci à toutes les personnes qui sont devenues mes ami·e·s. À Daniel, Thomas et Philippe, les trois mousquetaires, avec qui je compte vivre bien d'autres aventures de capes et d'épée. À Paul, Maëli, Eva, mes ami·e·s grenoblois·e·s, vous m'avez autant gagné que la montagne et vous me manquez. À Iris, Poly, Tenma, Louchette, Marven, mes ami·e·s de l'ADC avec qui j'ai tant aimé faire les 400 coups. Aux déconfiné·e·s ainsi que PL, Marie, Jean, Guilhem, merci d'être là depuis si longtemps. Merci à Ramen et Karma, et à celles qui me les ont confiés, vos petites boules de poils m'ont bien aidé dans la rédaction. Aux doctorants du troisième 26-00 Quentin, Roméo, Rémi, Jorje, Dimitri, Kevin, c'était un honneur de voguer sur la même galère que vous. Merci à Camille d'être là par tout temps.

Enfin, merci aux membres du jury de cette thèse; Alexander Barth et Emmanuel Cosme d'avoir pris sur leur temps estival pour rédiger un rapport, ainsi que Claire Monteleoni, Cécile Mallet, Alexandre Stegner et Maxime Ballarotta pour leur participation à la soutenance. Je remercie également toutes les personnes qui m'ont conseillées et orientées au cours de ma thèse; Sylvie, Carlos, Michel au LOCEAN, Alexandre, Briac, Evangelos et Hannah à Amphitrite, et aux inconnu·e·s qui ont relu mes articles.

# Contents

# List of Figures

# List of Tables

# Acronyms

**4DVarNet** 4 Dimensional Variational Network. 70, 71, 85, 110–112, 136

**ABED** Attention-Based Encoder Decoder. 89, 95, 105, 108–113, 126, 130

**BFN-QG** Back and Forth Nudging Quasi-Geostrophy. 70, 112, 113

**DIP** Deep Image Prior. xi, xii, 34, 71–84, 86, 110–112, 132

**DOI** Dynamic Optimal Interpolation. 70

**DUACS** Data Unification and Altimeter Combination System. xii, 3, 10, 11, 13, 22, 23, 25–28, 50, 52, 60, 67, 70, 74–78, 98, 112, 113, 126, 127

**MOI** Multiscale and multivariate Optimal Interpolation. 70

**MUSTI** Multi-variate Unsupervised Spatio-Temporal Interpolation. xii, xv, 78–84, 86, 110, 112, 113, 132

**3D-VAR** 3 Dimensional Variational data assimilation. 25, 29, 73

**4D-VAR** 4 Dimensional Variational data assimilation. 29–31, 71, 85

**AAE** Absolute Angular Error. 63

**ADAM** ADAptative Moment estimation optimizer. 57, 74, 81

**ADT** Absolute Dynamical Topography. 9, 10

**AMEDA** Angular Momentum for Eddy Detection and tracking Algorithm. 98–100

**AMSR** Advanced Microwave Scanning Radiometer. 15

**ANN** Artificial Neural Network. 32

**AVHRR** Advanced Very High-Resolution infrared Radiometer. 15, 17

**BFN** Back and Forth Nudging. 31, 32

**BLUE** Best Linear Unbiased Estimator. 25, 28, 70

**BN** Batch Normalization. 46

**BPTT** Back Propagation Through Time. 42

**CBAM** Convolutional Block Attention Module. 44, 45, 89

**CGAN** Conditional Generative Adversarial Network. 37, 40

**CNN** Convolutional Neural Network. 39–41, 71, 131

**ConvLSTM** Convolutional Long-Short Term Mermory. 43, 71, 110, 112, 113, 136

**DA** Data Assimilation. 2, 18, 24, 72

**DDPM** Denoising Diffusion Probabilistic Models. 35–37, 136

**DL** Deep Learning. 2, 32, 33, 67, 71, 72, 79, 119, 131–134

**DT** Delayed Time. 5, 22, 68, 96, 110, 121, 124, 127, 128, 130, 132

**GAN** Generative Adversarial Network. 35, 37, 136

**GCOS** Global Climate Observing System. 7

**GHRSST** Global High-Resolution SST. 15

**GLORYS** Global Physical Reanalysis. 49, 106, 121, 122

**GLORYS12** Global Physical Reanalysis 12°. 86, 87

**GPS** Global Positioning System. 9

**GPU** Graphical Processing Unit. 45

**KaRin** Ka-band Radar interferometer. 10, 11, 14

**KE** Kinetic Energy. 50

**LNAM** Local Normalized Angular Momentum. 98

**LSTM** Long-Short Term Mermory. 42, 43

**MDT** Mean Dynamical Topography. 9, 26

**MHCA** Multi-Head Cross Attention. 46

**MHSA** Multi-Head Self Attention. 45, 46

**ML** Machine Learning. 2, 32, 59

**MLP** Multi-Layer Perceptron. 32, 45

**MSE** Mean Squared Error. 54, 58, 81, 123, 126, 130, 139

**MSS** Mean Sea Surface. 9

**MUR SST** Multiscale Ultrahigh Resolution SST. 69, 107

**NEMO** Nucleus for European Modelling of the Ocean. 27, 49

**NLP** Natural Language Processing. 44, 46

**NOAA** National Oceanic and Atmospheric Administration. 15

**NRT** Near Real-Time. 6, 22, 119, 121, 122, 124, 127, 128, 130

**ODC2020** Ocean Data Challenge 2020. xi, xii, xv, 68–72, 76–78, 81–86, 105, 110, 111, 113, 114, 144, 145

**ODC2021** Ocean Data Challenge 2021. xii, xv, 68–70, 72, 76–78, 81–83, 85, 86, 108–111, 113, 116, 126, 144

**OGCM** Ocean General Circulation Models. 121

**OI** Optimal Interpolation. 10, 11, 16, 17, 22, 25–28, 70, 74, 98, 111, 113, 118, 131, 132, 134

**OSE** Observing System Experiment. 24, 28, 118, 119

**OSSE** Observing System Simulation Experiment. 23, 24, 27, 28, 71, 72, 85, 94, 106, 118, 119, 123, 135

**OSTIA** Operational Sea Surface Temperature and Ice Analysis. 16

**PRF** Pulse Repetition Frequency. 10

**PV** Potential Vorticity. 18

**QG** Quasi-Geostrophic. 18, 70, 113

**ReLU** Rectified Linear Unit. 39

**RESAC** REsolution by Stages of Altimetry and Currents. 5, 53–55

**ResNet** Residual neural Network. 41

**RMS** Root Mean Square. 11, 69, 139

**RMSE** Root Mean Squared Error. 78, 109, 111, 127, 128, 139

**RNN** Recurrent Neural Network. 41, 42

**RV** Relative Vorticity. 64

**SiLU** Sigmoid Linear Unit. 55

**SLA** Sea Level Anomaly. 9, 26

**SR** Super Resolution. 23, 33, 35, 37, 40, 53, 131

**SSH** Sea Surface Height. 2, 3, 9–11, 14, 17, 19, 21, 49, 50, 54, 64, 82, 106, 113, 128, 130–132, 134, 136

**SST** Sea Surface Temperature. xii, 3, 8, 15–17, 19, 49, 58, 64, 67–69, 71, 72, 78–86, 88, 89, 91, 93–95, 99, 100, 103, 106–113, 118, 131, 132

**SWH** Significant Wave Height. 9, 11, 14

**SWOT** Surface Water and Ocean Topography. 3, 10–12, 14, 22, 24, 73, 76, 77, 110, 111, 134

**ViT** Vision Transformer. 46, 108, 121, 136

# 1 Introduction

## 1.1 Context

Oceans represent a source of food, of commercial exchanges, but also natural frontiers and hazards; their strategic importance is undeniable. They also have a crucial role in regulating the Earth's climate. They absorb large amounts of solar radiation and redistribute heat around the globe through currents, which helps moderate temperatures and influence weather patterns [Schuckmann et al., 2020]. Oceans also store the main part of the emitted CO2 (50 times more than the atmosphere itself) [Raven and Falkowski, 1999]. Accurate knowledge of oceanic processes is therefore indispensable for improving climate models and developing effective strategies to mitigate and adapt to climate change.

Because of their immensity, observing the oceans is challenging, as it requires precise and broad monitoring of multiple variables. Historically, the first ocean observations were limited to coastal tides and marine life, as open-seas travel was perilous. Then, with the increasing mastery of navigation, measurements on board ships began to generalize in the commercial navy and in scientific expeditions, with increasing rigor and methodology [Manzella et al., 2022]. The latest development in ocean observation is the rise of artificial satellite measurements starting in the 1960's. Using various remote sensing technologies, satellites provide an unprecedented amount of data on many physical variables. However, they are mainly limited to surface observations, and depending on the employed sensor, their measurements can be noisy and sparse [Martin, 2014]. In parallel, the physical understanding of oceans drastically improved, leading to global numerical ocean models incorporating the physical knowledge acquired over the centuries. These models, although a powerful tool, are limited by numerical aspects [Chiba et al., 2019] or by their sensitivity to initial conditions [Thoppil et al., 2021]. Observations and physical models leave us with a thorny dilemma: deciding whether to trust what we can see or what we can understand.

Despite the latest advancements in satellite remote sensing, reconstructing the physical state of the ocean from observations alone appears impossible, given the number of unobserved variables. Combining physical models with observations

presents an appealing solution, as it mitigates the limitations of observations and reduces the sensitivity of physical models to initial conditions. This process is called Data Assimilation (DA) and is operationally used in many marine applications. It usually consists of finding the state that complies with physical knowledge, either a dynamical model or prior known distribution, and a set of observations [Asch et al., 2016,Carrassi et al., 2018]. Nevertheless, the increasing volume of available data has made data-driven approaches increasingly appealing. The Machine Learning (ML) methods aim to learn the state estimation on a dataset in which the task of interest should be represented. Among these techniques, Deep Learning (DL) is one of the most promising and has recently attracted renewed attention [Goodfellow et al., 2016]. In particular, DL is well suited to handle data of very high dimensions and to learn complex relationships between multiple variables. These characteristics are particularly appealing in the context of geosciences, where the dimensionality of the state to estimate is very high, and the underlying link between variables is complex. Given the profusion of observation data and physical simulations of the oceans, DL seems to be a viable candidate to improve the monitoring and understanding of the oceans.

## 1.2 Motivations and goals of the thesis

### 1.2.1 Sea Surface Height reconstruction

This thesis focuses on reconstructing the Sea Surface Height (SSH) using deep neural networks. SSH is a determining variable of the ocean, as its spatial gradient can be used to estimate the surface currents through the so-called geostrophic approximation. The currents estimated by this mean are issued from the equilibrium between Coriolis and pressure force in the ocean's surface layer. Far from the Equator, where the surface projection of the Coriolis force becomes null, it is a valid approximation of the circulation. The geostrophic currents are used in many applications such as navigation and weather prediction, hence our interest in retrieving accurate SSH fields.

SSH is currently measured by satellite altimetry, which consists of remote sensing methods that estimate surface height from space. In the past decades, SSH has been measured by nadir-pointing altimeters, meaning sensors able to retrieve SSH data "at the nadir", i.e., directly below the satellite. They calculate the return time of a radar pulse from the satellite to the sea surface and deduce the SSH. Because of this measurement principle, nadir-pointing altimeters leave significant gaps of unobserved

data, even when combining the observations of several satellites. Recently, the Surface Water and Ocean Topography (SWOT) satellite was launched, onboarding a new interferometric altimeter with the exciting property of providing high-resolution data over a wide swath of 120 km [Gaultier et al., 2016]. This acquisition improvement will surely improve our understanding of mesoscale dynamics in the ocean's surface layer. However, even considering these two data sources, reconstructing a complete SSH field is still a challenging spatiotemporal interpolation problem. Due to the long return time of the satellites, large areas can be unobserved for several days, which is a problem for rapidly evolving structures. The oceanography community uses interpolation methods to obtain a complete SSH field from the sparse observations, combining the data from all the satellites. One of the most widely used interpolation products in oceanography applications is provided by the Data Unification and Altimeter Combination System (DUACS) [Taburet et al., 2019]. It is a linear Optimal Interpolation (OI) of the nadir along-track measurements leveraging a covariance matrix tuned on 25 years of data. However, several studies show that DUACS OI misses some of the mesoscales structures and eddies [Amores et al., 2018, Stegner et al., 2021]. Improving the reconstruction of gridded altimetry products remains an open challenge.

### 1.2.2 Sea Surface Temperature contextual information

To enhance the quality of the SSH reconstruction and sea surface current estimation, using additional physical information such as the Sea Surface Temperature (SST) has been demonstrated to be beneficial [Ciani et al., 2020, Thiria et al., 2023, Martin et al., 2023, Archambault et al., 2023, Fablet et al., 2023, Archambault et al., 2024b]. SST motion is linked to ocean circulation [Isern-Fontanet et al., 2006], and therefore, to SSH, as currents transport heat through advection dynamics. SST measurements retrieved through passive infrared technology offer a remarkably high spatial resolution, ranging from 1.1 to 4.4 km [Emery et al., 1989], but clouds introduce data gaps. On the other hand, microwave sensors provide lower-resolution SST data (25 km), which can be obtained through non-raining clouds [Martin, 2014]. Infrared and microwave data are then combined with *in situ* measurements to produce fully gridded SST maps [Donlon et al., 2012, Chin et al., 2017]. Thus, a crucial challenge lies in developing efficient reconstruction methods capable of fusing data derived from different remote sensing techniques, each presenting distinct interpolation challenges. It is essential to unlock the full potential of satellite oceanography products. Other physical measurements might also improve the reconstruction, such as chlorophyll fields that track plankton advected by currents [Kahru et al., 2012].

However, in this thesis, we will only use SST data while keeping in mind that our methodologies could apply to other contextual data.

### 1.2.3 Objectives

Our main objective is to develop deep learning models able to improve the quality of SSH fields by leveraging SST information. We will study two settings: SSH downscaling and SSH interpolation, which are both image inverse problems that can be solved by neural networks. The downscaling setting corresponds to a situation where we try to enhance the quality of an existing field with low effective resolution and possibly errors. The neural network produces a high-resolution image from low-resolution input and possibly SST high-resolution contextual information. In the interpolation setting, on the other hand, the inputs are incomplete SSH measurements from which we aim to produce a complete SSH image.

Throughout this thesis, we began by exploring proofs of concept to demonstrate the value of this idea using simulated data. After these first experiments, we aimed to develop methodologies to train neural networks using only observations and to apply them to real data. With this focus, we prioritized developing training techniques over neural architecture design and hyper-parameter tuning. Finally, we adapted our methods to leverage both simulated and observational data, resulting in improved performance.

## 1.3 Outline and contributions

This thesis is organized as follows.

Chapter **2** focuses on the satellite observations of SSH and SST and describes the different measurement principles and sources of errors. We also explain the physical relationship between the two variables.

In Chapter **3**, we present SSH downscaling and interpolation as image inverse problems. Then, we present the standard methods to solve it: data assimilation and deep learning.

In Chapter **4**, we present our contributions on SSH downscaling using deep learning methods. This chapter is based on two publications:

- Thiria, S., Sorror, C., Archambault, T., Charantonis, A., Béréziat, D., Mejia, C., Molines, J.-M., and Crepon, M. (2023). *Downscaling of ocean fields by fusion of heterogeneous observations using deep learning algorithms*, published in Ocean Modeling.

- Archambault, T., Charantonis, A., Béréziat, D.,Thiria, S. (2022). *Sea surface height super-resolution using high-resolution sea surface temperature with a subpixel convolutional residual network*, published in Environmental Data Science.

The first one is a proof of concept showing the potential of simple convolutional neural networks fusing SSH and SST information to enhance the resolution of an SSH image. We test this idea on a physical simulation by introducing REsolution by Stages of Altimetry and Currents (RESAC), a simple convolutional neural network controlled at multiple resolutions, which obtains promising results. In the second publication, we push further the downscaling resolution and introduce a new version of RESAC, with a new architecture showing increased performances. In this thesis, we aggregate the experiments of these two publications into a complete study, and provide ablation studies missing from the two publications.

In Chapter **5**, we tackle the Delayed Time (DT) SSH interpolation problem. In DT interpolation, a sequence of observations is used to estimate the field at the central timestep. In this part, our main objective is to produce a method that can be trained using satellite observations to get toward an operational SSH product. The chapter is based on the following publications:

- Filoche, A., Archambault, T., Charantonis, A., and Béréziat, D. (2022). *Statistics-free interpolation of ocean observations with deep spatio-temporal prior*, published in ECML/PKDD Workshop on Machine Learning for Earth Observation and Prediction (MACLEAN).

- Archambault, T., Filoche, A., Charantonnis, A., and Béréziat, D. (2023). *Multimodal Unsupervised Spatio-Temporal Interpolation of satellite ocean altimetry maps*, published in the International Conference on Computer Vision Theory and Applications (VISAPP).

- Archambault, T., Filoche, A., Charantonis, A., Béréziat, D., and Thiria, S. (2024). *Learning sea surface height interpolation from multi-variate simulated satellite observations*, published in Journal of Advances of Modeling Earth Systems (JAMES).

- Archambault, T., Filoche, A., Charantonnis, A., and Béréziat, D. (2024). *Pre-training and fine-tuning attention based encoder decoder improves sea surface height multi-variate inpainting*, published in the International Conference on Computer Vision Theory and Applications (VISAPP).

The first article presents a method for fitting a neural network on a single example using only observations. In the second article, we extend this study by including SST information, showing that it improves SSH reconstruction. However, these two methods are fitted on a small number of examples and must be refitted to estimate unseen data, which can lead to a computational burden if applied operationally. To tackle this issue, we introduce in the third article a new training dataset, including simulated and real observations. The two datasets (simulation and real data) are available online[1]. They include 20 years of simulated and satellite observations of SSH and SST on the Gulf Stream area and the weights of our neural networks. We believe that this dataset is a significant contribution to the state-of-the-art, as it can be used to compare interpolation methods on the same training and testing data, which was lacking in the literature (see Section **5.4.1**). On simulated data, we compare training methodologies able to learn from observations only and evaluate their reconstruction performance. In the last publication, we also derive a way to use simulated data and real observations to improve the SSH interpolation, and we show that it outperforms learning from simulation or observations separately. In this chapter, we provide additional conclusions to those drawn in the four publications by providing ablation studies, new experiments, and a complete comparison of the state-of-the-art interpolation methods.

In Chapter **6**, we delve deeper into operational oceanography by focusing on Near Real-Time (NRT) and forecast SSH field estimation. Many marine applications require real-time or forecast data to plan ship routing and anticipate physical phenomena. We adapt the training methodologies to estimate such fields. This is a preliminary work that has not been published.

Finally, in Chapter **7**, we summarize the conclusions drawn in the thesis and give some perspectives for future work and developments.

[1] `https://zenodo.org/records/10551897` [Archambault, 2024]

# 2 Satellite observation of the ocean

## 2.1 Satellite remote sensing

### 2.1.1 Earth observations from space

Artificial satellites are indispensable tools in many applications, such as telecommunication, cartography, defense, and geoscience remote sensing. In 1959, only two years after the launch of the first artificial satellite, Explorer VI took the first Earth image from orbit: a picture of the Pacific Ocean (Figure **2.1**). Unfortunately, the cloud cover and the sunglint have polluted the photo to the point where the shape of the Earth is barely recognizable. This first experiment, although prescient of many problems in remote sensing, launched an uninterrupted series of observations of the Earth from space.

**Fig. 2.1.:** First satellite image of Earth taken by Explorer VI on August 14, 1959. [Source: NASA].

The first weather satellite, TIROS-1, was launched in 1960, making spaceborne remote sensing of physical variables possible. In addition to weather monitoring, satellites were also used to measure the evolution of the Earth's climate. Ice-melt, increasing Earth's atmosphere and ocean temperature, sea-level increase; out of the 50 essential climate variables defined by the Global Climate Observing System (GCOS), 26 are estimated through satellite [Yang et al., 2013]. Due to their massive heat capacity, oceans play a determining role in climate regulation. Between 1971 and 2018, they captured 89% of the heat gain (6% for the land mass, 4% for the ice melt, and 1% by the atmosphere) [Schuckmann et al., 2020]. Understanding their dynamics and evolution is a key in many scientific and technical fields, ranging from climatology to

navigation. Presently, satellites are used to measure various ocean surface variables, such as height, temperature, ice fraction, or Chlorophyll-A concentration.

### 2.1.2 Satellite products and processing levels

The ocean variables observed through satellites are combined, processed, and analyzed in what we denominate satellite products. From the instrument data to the fully gridded maps that can be used by a broad audience, several processing levels exist [Martin, 2014]:

- **Level 0 (L0):** The data sent by the satellite without any preprocessing except for communication artifacts corrections and duplicate removal.
- **Level 1 (L1):** The data processed to full resolution, geolocalized, calibrated, and converted to the sensor unit.
- **Level 2 (L2):** The L1 data is transformed into the geophysical data of interest. Taking SST as an example, the data of the L1 product measures infrared intensity, whereas the L2 product is converted to °C. Some corrections, such as a cloud mask or atmospheric corrections, are also applied.
- **Level 3 (L3):** The data mapped to a uniform spatio-temporal grid, but without completing the data gaps. The L3 products can be made from the data from a single satellite or collated from satellites at similar resolutions (L3C for collated).
- **Level 4 (L4):** The highest level of Data processing. Usually, the L4 products are obtained by combining several satellite sensors and *in situ* measurements and interpolated where data is missing.

## 2.2 Ocean altimetry

### 2.2.1 Principle and notations

Radar altimeters embedded in satellites measure the return time of a radar pulse from the instrument to any ground surface. Using this data, as well as an estimation of the water quantity in the atmosphere - that impacts the propagation speed of the radar pulse - it is possible to calculate the distance $D$ between the satellite and the overflight reflective surface. Some instrumental specific corrections (internal

delays, the physical structure of the sensor, ...) and sea state corrections must also be applied to give an accurate estimation. The shape of the returned pulse provides other information such as Significant Wave Height (SWH), or surface wind speed. The satellite's altitude $H$ can be precisely determined with the Global Positioning System (GPS) and is defined from an Earth reference. Two references can be used: the Earth's ellipsoid or the geoid (equipotential surface of the Earth's gravity). If the first reference is used, the main variations in the captured signal are due to the Earth's interior heterogeneity, whereas in most of the dynamic ocean applications, the parameter of interest is an altimetry reference to the static geoid [Stammer and Cazenave, 2017, Pujol et al., 2018].

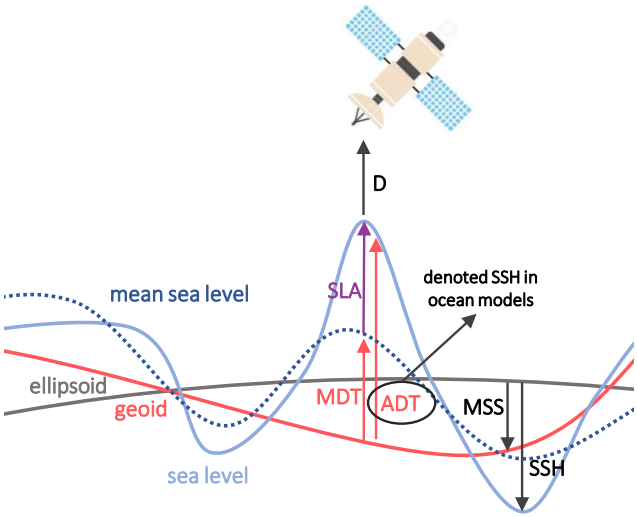


**Fig. 2.2.:** Altimetry variable notations.

We call mean sea level the temporal mean of the ocean computed on an arbitrarily chosen time period. When this mean level is computed from the ellipsoid it is called Mean Sea Surface (MSS) and Mean Dynamical Topography (MDT) when the reference is the geoid. We establish hereafter some notations used by the altimetry community of sea surface altitudes represented in Figure **2.2**:

- In the altimetry community, the Sea Surface Height (SSH) is the distance between the ellipsoid and the sea level.
- The Absolute Dynamical Topography (ADT) is the distance between the geoid and the sea level.
- ADT can be decomposed as the sum of its temporal mean (MDT) and the instantaneous anomaly called Sea Level Anomaly (SLA).

These notations correspond to the altimetry denomination. But in an oceanographic context, the SSH often refers to the sea surface height above the geoid, i.e. the ADT. As ADT is the useful variable to derive geostrophic currents, we adopt this notation and call hereafter SSH the difference between the geoid and the sea level.

### 2.2.2 Nadir and interferometer altimetry

SSH is currently measured by various nadir-pointing altimeters, meaning that the sensor can only take measurements vertically, along the ground tracks of the satellite. The nadir radar system measures the return time of a radar pulse emitted periodically. The Pulse Repetition Frequency (PRF) is chosen so that each backscattered wave returns before the emission of the next pulse. This way, the ambiguity about which pulse caused which echo is removed. Combining data from different missions, the L4 gridded SSH images are reconstructed through linear Optimal Interpolation (OI) such as Data Unification and Altimeter Combination System (DUACS) [Taburet et al., 2019].

To enhance SSH recovery, a new altimeter called Surface Water and Ocean Topography (SWOT) was launched on the 16 *th* of December 2022. It provides two 60-km-wide swaths separated 20-km gap instead of nadir observations [Gaultier et al., 2016]. Thanks to its high resolution (250 m per pixel for the best products) and data coverage, SWOT will push further our understanding of ocean topography and hydrology. Its sensor technology is different from the nadir altimeter's as it is based on a Ka-band Radar interferometer (KaRin) measure. Figure **2.3** shows a simplified overview of the KaRin measurement principle. The satellite is composed of 2 antennas $A_1$ and $A_2$ separated by a distance $d$ emitting in a bistatic mode; only one antenna can emit, but both can receive. We aim to measure SSH at an altitude $H$ from any reference. The $A_1$ antenna emits a radar pulse that is reflected on the ocean. First, the distance $r_1$ is estimated from the return time in a classical altimetry way. The two antennas receive the backscattered signal with a phase

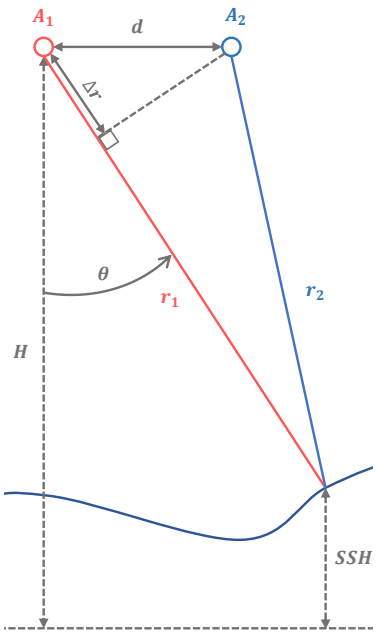


**Fig. 2.3.:** SWOT interferometer principle. For more simplicity, only one look-up angle ($\theta$) is represented, and without roll correction.

shift of $\Delta\phi = 2\pi\nu(r_1 - r_2) = 2\pi\nu\Delta r$, where $\nu$ is the pulse frequency. This phase shift is used to estimate the angle $\theta$ as $\Delta\phi = 2\pi d\nu\sin(\theta)$. However, the same phase shift can lead to different measurements of $\Delta r$ as $\Delta\phi = 2n\pi + \Delta\phi'$ (where only $\Delta\phi'$ is retrieved). To unwrap the phase shift, we must use auxiliary data, giving a proxy of $\Delta r$. For the KaRin sensor, the ambiguity amplitude is between $10$ to $60$ meters, which, given the known amplitudes of SSH, leads to only one value for $\Delta r$. Finally, the SSH is computed as $= H - r_1\cos(\theta)$. This explanation is simplified as, in reality, the horizontal localization of the pixel should also be determined, and corrections are applied (satellite roll, tropospheric delay) [Fjørtoft et al., 2010, Fjørtoft et al., 2014, Rosen et al., 2000].

In Figure **2.4**, we present an example of the L3 altimetric data over the North Pacific. The large swath corresponds to SWOT measurements, with a product resolution of 2 km [NASA/JPL and CNES, 2024], and the thin along-track data come from the nadir-pointing altimeters [CMEMS, 2021]. In Figure **2.5** we plot the associated L4 product: the DUACS [CMEMS, 2023a, Taburet et al., 2019] OI (see Section **3.2.1**).

### 2.2.3 Sources of errors

#### Nadir altimeter errors

Nadir pointing altimeters have various sources of error summarized in Table **2.1**, which can be classified into the following categories: the altimeter noise, the atmospheric correction errors, the sea state bias, and the orbital position error. All the values are given for a SWH of 2m.

| Error sources (cm) | TOPEX | JASON-1 | JASON-2 |
|---|---|---|---|
| Altimeter noise | 1.7 | 1.6 | 1.8 |
| Atmospheric corrections | | | |
| - *dry troposphere* | 0.7 | 0.7 | 0.7 |
| - *wet troposphere* | 1.1 | 1.2 | 0.8 |
| - *ionosphere* | 0.5 | 0.5 | 0.3 |
| Sea state bias | 2.3 | 2 | 2 |
| Orbital position error | 2.5 | 1.5 | 1 |
| Total SSH error | 4.1 | 3.3 | 3.1 |

**Tab. 2.1.:** Error sources of the nadir altimetry measurements [Martin, 2014].

For JASON-2, the instrumental noise of the altimeter has a Root Mean Square (RMS) of 1.8 cm for SWH of 2 m. This noise increases with the height of the waves because the backscattered amplitude is smaller and has a wider temporal support.

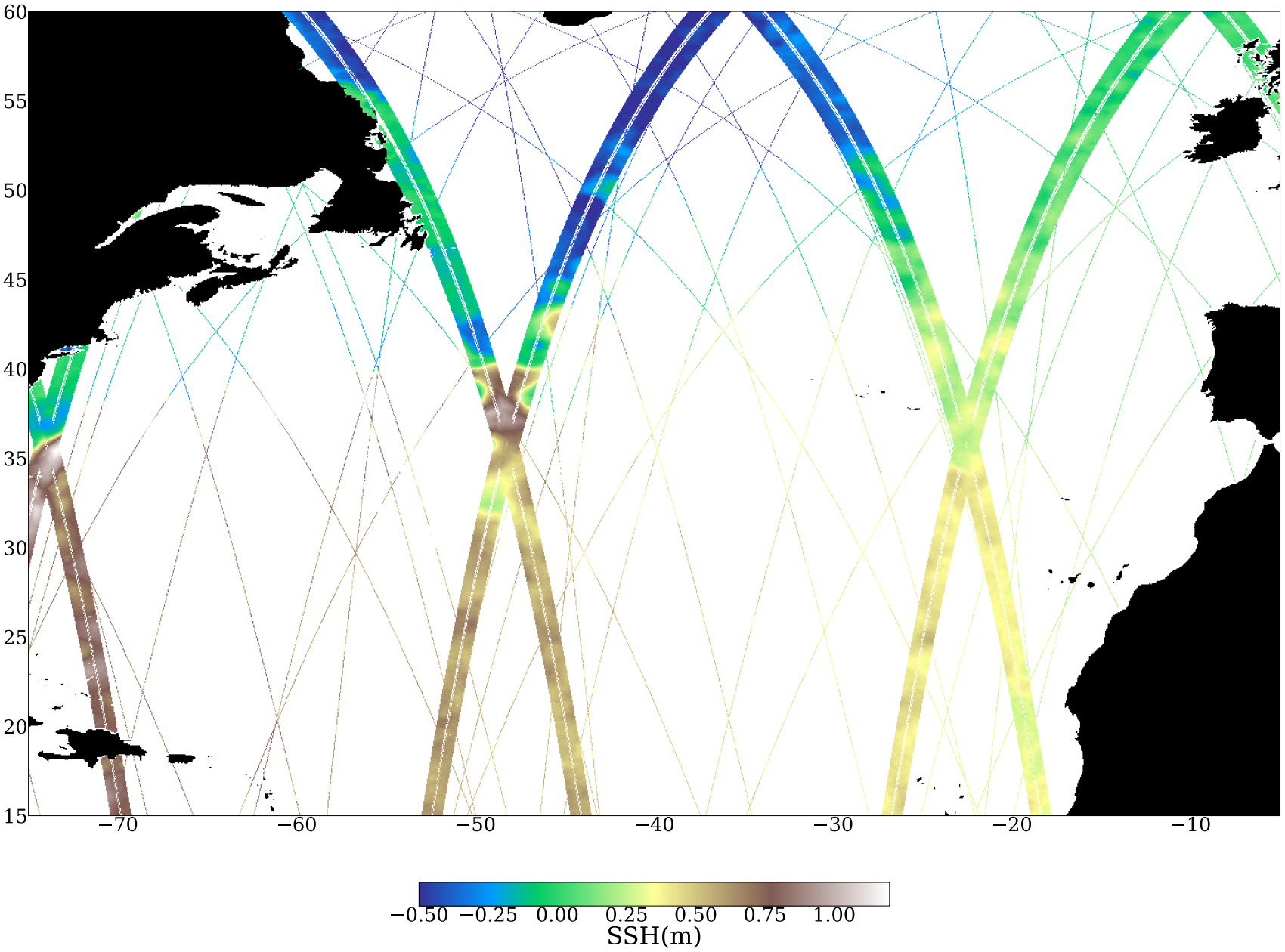


**Fig. 2.4.:** An example of SWOT and nadir pointing altimeters L3 observations, on the North Atlantic. We plot all the data collected by SWOT (wide swath) and the available nadir-pointing satellites (fine tracks) during one day (2023/05/14).

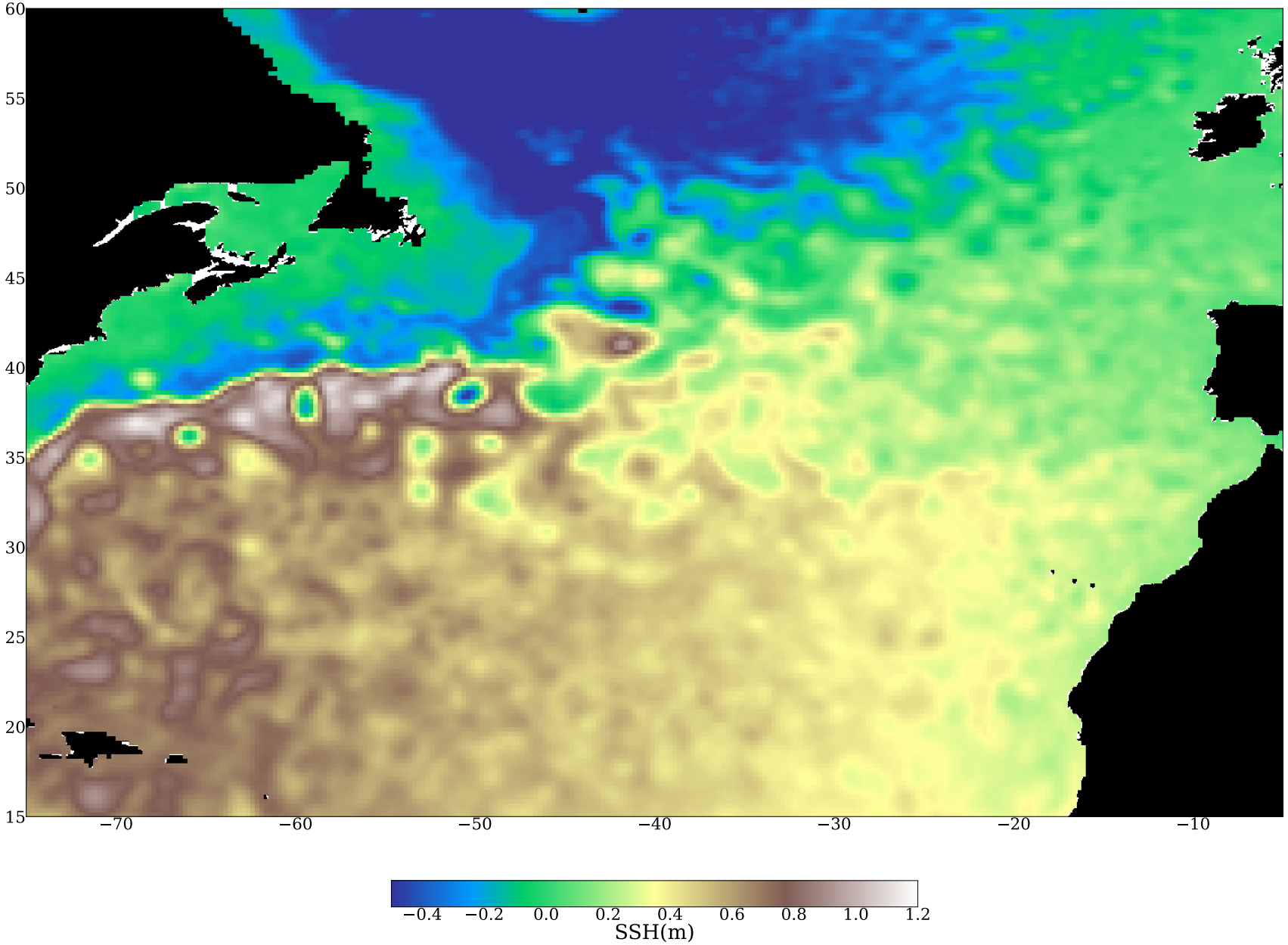


**Fig. 2.5.:** An example DUACS L4 interpolated SSH field on the same day than in Figure **2.4**. The interpolation is done from nadir-pointing altimeter data.

Atmospheric corrections are applied as its composition impacts the speed of light. They are divided into three parts: dry troposphere (composition of all gases except water), wet troposphere (vapor and liquid water), and ionosphere (free electrons). The state of the sea surface also biases the estimation notably because wave troughs better reflect the radar pulse than wave crests. As different shapes of waves can be observed for the same SWH, the value of this bias is hard to estimate, and we usually consider that the resulting error is around 2% of the SWH. Finally, depending on the satellite's orbit, its position can present some uncertainty, which also leads to errors [Martin, 2014, Stammer and Cazenave, 2017].

### SWOT errors

The SWOT errors are usually divided into two spectral domains: errors at wavelengths shorter than 1000 km and errors at higher wavelengths. As the SWOT mission's primary focus is to retrieve SSH frequency variations, the wavelength above 1000 km is estimated through standard nadir altimetry with errors similar to those given in **2.1** [Peral and Esteban-Fernandez, 2018]. However, short wavelengths are recovered through the KaRin instrument and follow a different budget. [Gaultier et al., 2016] introduced a framework to model the KaRin errors, including the altimeter noise (which increases with SWH and distance to nadir), roll, baseline dilatation, phase, and timing errors.

### Other sources of uncertainty

We have listed error sources that can affect the estimation of the range between the satellite and the sea level. However, to retrieve the height above the geoid, some environmental errors should be considered. For instance, tides and sea level pressure induce variations in the sea level that should not be taken into account in SSH calculations. Contrary to the error sources previously mentioned, these corrections are needed to infer geostrophic currents but correspond to real sea level differences and not measurement errors. Usually, these errors are removed using physical models of atmospheric pressure [Chelton et al., 2001] and tides [Le Provost, 2001].

## 2.3 Sea Surface Temperature

The ocean temperature is one of the most important variables for understanding both global climate change and local variability. As the first 3 meters of the ocean have the same heat capacity as the Earth's atmosphere, the ocean temperature is a determining feature for meteorological and climate models. Due to this massive heat storage capacity, the ocean distributes the heat captured near the Equator to the entire planet [Martin, 2014]. Furthermore, SST temporal evolution is linked to oceanic circulation, as currents transport heat through an advection dynamic [Isern-Fontanet et al., 2006].

### 2.3.1 SST Remote sensing

Since the launch of the National Oceanic and Atmospheric Administration (NOAA)-7 weather satellite in 1981, the ocean's surface temperature has been observed from space. During the last four decades, SST data were collected with increasing resolution and quality to provide a deeper understanding of complex ocean phenomena such as climate change, inter-annual variability, or local circulation. To combine SST information on an international level and unify data format and preprocessing, Global High-Resolution SST (GHRSST) was created in 2001, leading to significant improvement in aggregating SST from various source sensors. We hereby give a brief overview of sensing technologies for ocean temperature.

First, infrared radiometers such as the Advanced Very High-Resolution infrared Radiometer (AVHRR) offer a remarkably high spatial resolution, ranging from 1.1 to 4.4 km [Emery et al., 1989]. They can take measurements during both the night and daytime. However, these sensors are cloud-sensitive, which introduces missing data.

Another type of radiometer is the Advanced Microwave Scanning Radiometer (AMSR) instrument. Due to their longer wavelength bands, they capture data through non-raining clouds. However, they have a low resolution compared to infrared ones (30 km) and can not be used too close to the land (75 km away from the seashore). This sensing technology is complementary to AVHRR's, so they are often used conjointly in L4 products.

These two remote sensing data are combined with by *in situ* measurements from commercial ships, buoys, and drifters. These sensors are unequally distributed on the Earth's surface, as the commercial navy is more intensive in the North hemisphere,

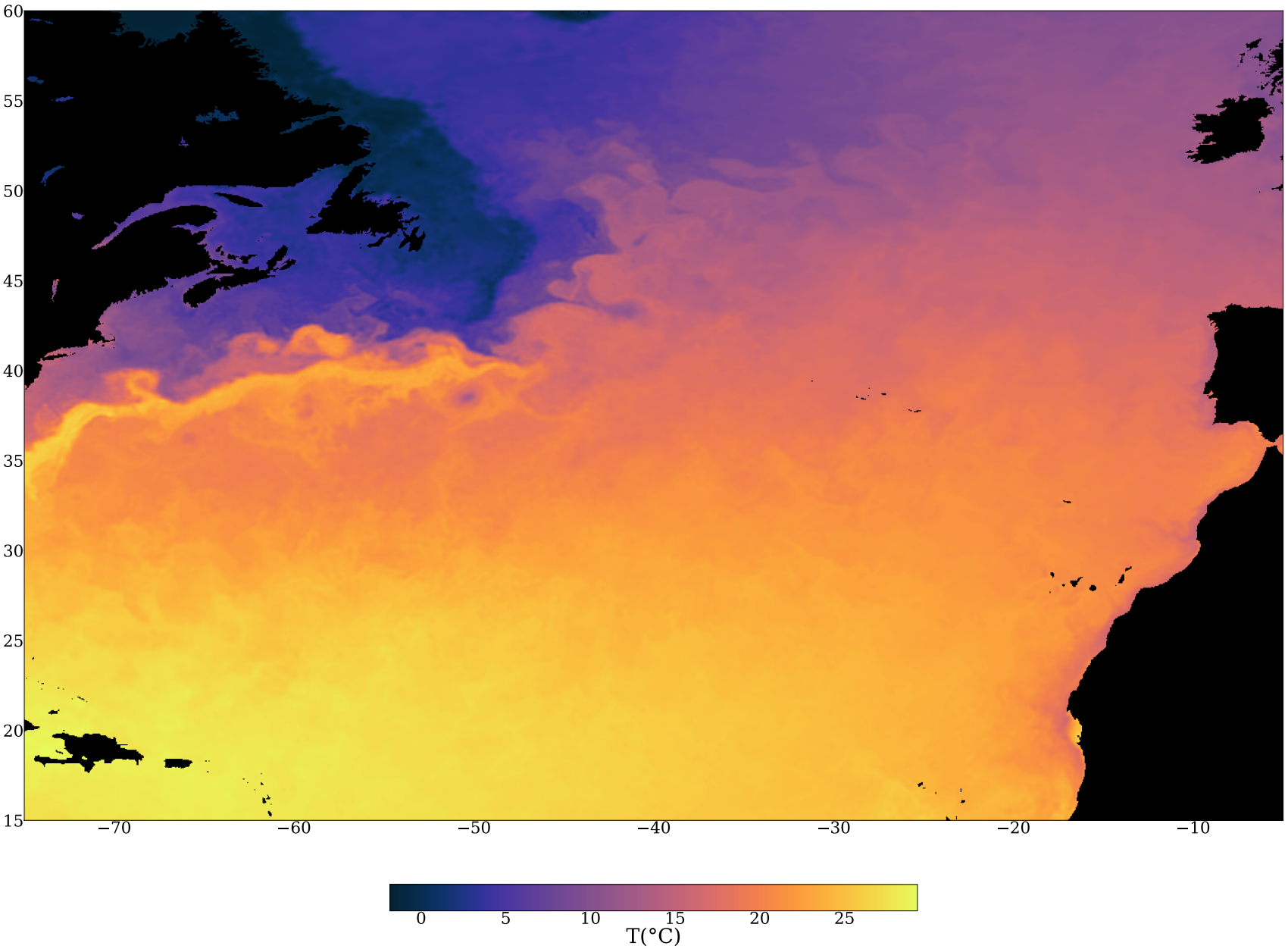


**Fig. 2.6.:** An example of SST OI, from Operational Sea Surface Temperature and Ice Analysis (OSTIA) [CMEMS, 2023b], corresponding to the same day than in Figure**2.4**.

and drifters follow oceanic currents. Nevertheless, *in situ* observations complement infrared and passive microwave remote sensing technologies, specifically in calibration [Martin, 2014]. Finally, the L4 SST complete fields are produced through OI using infrared and microwave radiometers and in *in situ* observations [Donlon et al., 2012, Chin et al., 2017]. Figure **2.6** gives an example of an interpolated product.

### 2.3.2 Validation of SST satellite products

The main source of errors in the infrared radiometers is the instrumental noise, as well as undetected clouds which are interpreted as a small temperature variation. To validate the SST L3 infrared observations, we must compare them to measurements from other sources: usually buoys, ships, or even radiometers [Martin, 2014, Barale et al., 2010]. By comparing AVHRR measurements from every satellite between 1985 and 1998 with matching drifting buoys, [Kilpatrick et al., 2001] found a bias of 0.02°C and a standard deviation of 0.53°C. However, the authors indicate that the small bias found is misleading, as it changes with latitude and seasons.

## 2.4 Physical relationship between SSH and SST

In the following we motivate the inclusion of SST information in SSH reconstruction by describing their physical relationship. We first explain the SSH link to surface currents and then temperature advection.

### 2.4.1 Geostrophic approximation

One of the most important uses of SSH data is to recover oceanic currents through geostrophic approximation. It consists of supposing a static equilibrium between the surface projection of the Coriolis force $\mathbf{F}_{\mathrm{c}}$ and the resultant pressure forces $\mathbf{F}_{\mathrm{p}}$. Near the Equator, the Coriolis force projection is null, and this approximation is not valid. However, a few degrees away (about 2°), geostrophy is a good estimation of circulation [Stewart, 2006]. The geostrophic equilibrium can be written from the volumic pressure and Coriolis forces as follows:

$$\rho f_0 \mathbf{k} \times \mathbf{w}_{\mathrm{g}} = -\nabla p \tag{2.1}$$

$$\Leftrightarrow \mathbf{w}_{\mathrm{g}} = \frac{1}{f_0 \rho} \mathbf{k} \times \nabla p \tag{2.2}$$

where $\rho$ is the water volumic mass, $f_0 = 2\Omega_r \sin(\phi)$ is the Coriolis factor, $\Omega_r$ being the Earth the rotation period, $\phi$ the latitude and $g$ the gravitational acceleration, and $\mathbf{k}$ the unit vector normal to the considered surface, and $\times$ the cross-product. The surface pressure can be expressed as $p = \rho g(h + r)$ where $h$ is the surface height and $r$ is the geoid height. The surface geostrophic circulation $\mathbf{w}_{\text{geo}}$ can be computed following Equation **2.3**:

$$\mathbf{w}_{\text{g}} = \begin{pmatrix} u_{\text{g}} \\ v_{\text{g}} \end{pmatrix} = \begin{pmatrix} -\frac{g}{f_0}\frac{\partial h}{\partial y} \\ \frac{g}{f_0}\frac{\partial h}{\partial x} \end{pmatrix} \tag{2.3}$$

where $u_{\text{g}}$ and $v_{\text{g}}$ are the Eastward and Northward geostrophic currents, $x$ and $y$ the Eastward and Northward coordinates.

### 2.4.2 Quasi-Geostrophic model

Although very useful as a proxy to surface currents, the geostrophic approximation is a static equilibrium in which the currents do not evolve. Quasi-Geostrophic (QG) flow refers to a situation where the circulation is nearly at the geostrophic equilibrium. In this model, the Potential Vorticity (PV) is conserved along the geostrophic streamlines [Lapeyre, 2017]. PV, denoted $q$ can be expressed as:

$$q = \nabla^2\psi - \frac{1}{R_d}\psi \tag{2.4}$$

where $\psi = \frac{g}{f_0}h$ is the stream function, $R_d$ is the Rossby deformation radius, and $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. PV conservation along the geostrophic flow means that its material derivative is null:

$$\frac{dq}{dt} = \frac{\partial q}{\partial t} + \mathbf{w}_{\text{g}} \cdot \nabla q = 0 \tag{2.5}$$

This simplified version of the Navier-Stokes equations is useful for modeling SSH evolution. As its computational complexity is rather low, it can be used as a small physical model in a DA framework (see Section **3.2**). Several studies highlight QG potential in mapping SSH fields [Le Guillou et al., 2020, Ballarotta et al., 2020, Le Guillou et al., 2023].

### 2.4.3 SST advection

After describing the link between the SSH and the surface currents, we explain why temperature also contains information about the circulation. In a first approximation, the surface temperature $T$ can be considered as a passive tracer advected by surface currents [Lapeyre, 2017]. The transport of a scalar quantity in a velocity field is described by the linear advection Equation **2.6**.

$$\frac{\partial T}{\partial t} + \mathbf{w} \cdot \nabla T = 0 \tag{2.6}$$

Combining the geostrophic and the advection Equations (**2.3**,**2.6**), we understand why a time series of SST observations should provide pertinent information for constraining the SSH reconstruction. The SST evolution contains contextual information about the currents, which could be used to recover a more precise SSH. However, the actual physical link between temperature and SSH is more complex, as other phenomena must be considered, such as diffusion, convection, circulation between water depths, atmosphere interactions, and viscosity. [Piterbarg, 2009] theorized a method to correct a background velocity field from a passive tracer and a poorly known forcing. This method was successfully applied to the SSH reconstruction by [Rio and Santoleri, 2018, Ciani et al., 2020, Ciani et al., 2024]. Using the geostrophic currents from L4 SSH products as a background, the authors were able to enhance the circulation thanks to the SST tracer. The SST forcing was set to the temporal derivative of a spatial low-pass filtered SST, which means that the only variations considered had high spatial wavelength (500 km). The resulting currents were evaluated on *in situ* boys currents, showing general improvement, which further motivated the use of SST in SSH reconstruction.

## 2.5 Conclusion

Spaceborne altimetry is a powerful tool to estimate ocean surface currents. However, due to the remote sensing technology employed, the satellite observations are sparse and noisy, as shown in Figure **2.4**. On its behalf, the surface temperature is retrieved through direct imaging technology, leading to broad data coverage and high-resolution data. As temperature is transported by currents, it provides valuable information on the circulation. Using SST contextual information to reconstruct a high-quality SSH field is an exciting idea that motivated this thesis.

# 3 Problem statement and related work

## 3.1 Formulating the reconstruction as an inverse problem

### 3.1.1 Inverse problems

The satellite observation system can be modeled as a function producing observations from the environment state. Let $\mathbf{x} \in \mathbb{R}^n$ be the state of the physical system, from which the operator $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^p$ produces observations $\mathbf{y} \in \mathbb{R}^p$ such as :

$$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \varepsilon \tag{3.1}$$

where $\varepsilon$ is a noise representing instrumental errors. Producing $\mathbf{y}$ from $\mathbf{x}$ is called the *forward problem*, and $\mathcal{F}$ the *forward operator*. The *inverse problem* associated with Equation **3.1** consists in inverting the forward operator, meaning finding a function that outputs the state $\mathbf{x}$ from observations $\mathbf{y}$. According to [Hadamard, 1924], this inverse problem is said well-posed if:

- For every observation $\mathbf{y}$ it exists at least one solution $\mathbf{x}$
- This solution is unique
- The solution depends on the data in a continuous manner, meaning that it exists a positive constant $C$ such as $||\mathbf{x} - \mathbf{x}'|| < C\,||\mathbf{y} - \mathbf{y}'||$

If one of these conditions is not satisfied, the problem is said to be ill-posed. Strictly inverting the problem is difficult, but one can still aim to estimate the state. When the state and observations can be shaped as images, we refer to this problem as an inverse image problem. A wide variety of inverse problems exist, such as denoising, inpainting, super-resolution, deconvolution... The following details how the SSH recontruction can be seen as an image inverse problem.

### 3.1.2 SSH interpolation

Let us say that $\mathbf{X}^{\text{ssh}}$[1] is the SSH state from which derive the observations. The satellite data from nadir-pointing altimeters or SWOT wider swaths are taken on sparse support, leaving unobserved data elsewhere. Let $\mathbf{\Omega} = \{\Omega_i = (t_i, lat_i, lon_i)\,, i \in [0:p]\}$ be the support of this observations. We derive SSH observations $\mathbf{Y}^{\text{ssh}}$ as the $\mathbf{X}^{\text{ssh}}$ on each point of the support as following:

$$\mathbf{Y}^{\text{ssh}} = \mathcal{H}^{\text{ssh}}\left(\mathbf{X}^{\text{ssh}}, \mathbf{\Omega}\right) + \varepsilon \tag{3.2}$$

where $\varepsilon$ is the measurement error and $\mathcal{H}^{\text{ssh}}$ the observation operator. To each coordinate in the support, $\mathcal{H}^{\text{ssh}}$ simply associates the value from the SSH state.

This definition of the problem considers that the SSH state is a field defined everywhere. To formulate it as an image problem, a few additional concerns should be taken into account. First, $\mathbf{X}^{\text{ssh}}$ must be discretized on a regular spatiotemporal grid of shape $T \times H \times W$ to be considered as an image (or a time series of $T$ images). Then, if the support of the observations is not defined in a regular grid (which is the case for L3 products of nadir-pointing altimeters), we must approximate the SSH observations using an interpolation method. In this case, $\mathcal{H}^{\text{ssh}}$ is defined as the interpolation of the SSH state at the locations of each observed coordinate. The interpolation can be chosen arbitrarily, but in this thesis, we use trilinear interpolation. If we also want $\mathbf{Y}^{\text{ssh}}$ to be an image, we must regrid observations. For instance, one can average observations inside each pixel and set the pixel value to zero when no data is taken. We distinguish two types of spatiotemporal interpolations: Delayed Time (DT) interpolations, where we have access to observations in the future compared to the target timestep, and Near Real-Time (NRT) interpolations, where we have only access to past observations. The first one produces the best interpolation possible but is unavailable in real-time. In Chapter **5**, we tackle the interpolation in a DT framework, and we extend this study in NRT in Chapter **6**.

### 3.1.3 SSH downscaling

Many operational oceanography applications require complete SSH fields to estimate currents. The SSH observations are thus interpolated into L4 SSH products, such as DUACS, which estimates SSH fields through Optimal Interpolation (OI) see Section **3.2.1**. However, linear interpolations are known to miss ocean

[1] To differentiate between matrices and vectors, we write matrices and images in capital letters and vectors in lower case.

mesoscales structures [Amores et al., 2018, Stegner et al., 2021], and have coarse resolutions [Ballarotta et al., 2019] as discussed in Section **3.2.1**. Improving the resolutions of satellite L4 products is a challenge in many applications and can also be seen as an image inverse problem.

Let us call $\mathbf{I}^{\mathrm{HR}}$ a high-resolution image and $\mathcal{D}_{\mathrm{ec}}$ the forward operator from Equation **3.1**, which we call a decimation operator. From the high-resolution image, $\mathcal{D}_{\mathrm{ec}}$ produces $\mathbf{I}^{\mathrm{LR}}$, a lower resolution image:

$$\mathbf{I}^{\mathrm{LR}} = \mathcal{D}_{\mathrm{ec}}(\mathbf{I}^{\mathrm{HR}}) \tag{3.3}$$

where $\mathbf{I}^{\mathrm{HR}} \in \mathbb{R}^{H \times W}$ and $\mathbf{I}^{\mathrm{LR}} \in \mathbb{R}^{h \times w}$ with $H = rh$, $W = rw$ where $r$ is the downsampling ratio, usually an integer $> 2$. This inverse problem is named downscaling in oceanography communities and remote sensing and called Super Resolution (SR) in the image processing community. The most common operator for $\mathcal{D}_{\mathrm{ec}}$ is an average pool, i.e. the mean of the pixels values in a $r \times r$ neighboring, but others can be considered. In the case of SSH downscaling, the aim is to recover a high-resolution SSH image, using DUACS coarse L4 product as input. In this setting, $\mathcal{D}_{\mathrm{ec}}$ is not fully known as it derives from the interpolation process. Section **5.1** we will see why this lack of knowledge of the decimation operator is a limitation of seeing SSH remote sensing as a downscaling problem.

### 3.1.4 Studying inverse problems in geosciences

One of the main challenges in geoscience is that the full physical state is never accessible, making the inversion method assessment difficult. Consequently, developing and evaluating reconstructions through indirect means is a crucial aspect of oceanography research. In the following sections, we present two approaches to address this issue.

#### Observing System Simulation Experiment

An Observing System Simulation Experiment (OSSE) is a widely used method where we aim to replicate the observing system on a physical simulation [Zeng et al., 2020]. In this mindset, the physical state is the output variable of a numerical model, upon which we emulate the forward operator. In doing so, we have access to pairs of "ground truth" and associated "pseudo-observations" that we can use to our benefit. This approach is also called *twin experiment* in the Machine Learning community.

The OSSEs have several purposes: first, they can be used to model the observations of a new observing device and estimate its potential [Zeng et al., 2020]. For instance, [Gaultier et al., 2016] introduced the *swot-simulator*, a software emulating the SWOT observations in order to show its potential in retrieving SSH high frequencies. Second, it can be used to assess the inversion method itself. For instance, [Amores et al., 2018, Stegner et al., 2021] used this method to characterize the limits of interpolated altimetry maps. Third, given enough pairs of pseudo-observations and ground truth, OSSEs can be used to train data-driven inversion methods, including deep neural networks as shown in Section **5.4.1**. However, OSSE methodology still presents some limitations. They entirely rely on the realism of their numerical simulation and observation operator, which is possibly different from the real physical state and observing system. This is why it is also possible to work with real data, as presented in the next section.

#### Observing System Experiment

Observing System Experiment (OSE) follow the same objectives as OSSEs but are based on real observations instead of simulation. Unlike OSSEs, OSEs do not face issues related to the realism of simulations. However, using only real observations can make it more challenging to assess reconstruction performance. Typically, some observations are withheld from the inversion process to serve as independent validation data. Additionally, modeling the forward operator is necessary, as it enables comparisons between observations and estimated states, permitting the evaluation process.

## 3.2 Data Assimilation

Data Assimilation (DA) is a set of methods aiming to combine observations "data" with physical knowledge "assimilation", to estimate accurately the state of a system [Asch et al., 2016]. Physical knowledge can be a dynamical model giving information about temporal evolution, but also covariances matrices of observed and unobserved variables. DA is used in many geoscience applications, such as meteorology or oceanography, and constitutes the core of the physical analysis of studied system [Asch et al., 2016]. In the following, we present a non-exhaustive overview of classical DA methods that were used in SSH interpolation problems.

### 3.2.1 Optimal Interpolation: the example of DUACS

#### Optimal Interpolation

Let $\mathbf{x}$, $\mathbf{x}^{\mathrm{b}}$, $\mathbf{x}^{\mathrm{a}} \in \mathbb{R}^n$ be respectively the true state that we want to estimate, the background state, a "first guess" that we have on $\mathbf{x}$ and the analyzed. Let $\mathbf{y} \in \mathbb{R}^p$ be the observations obtained through the observation operator $\boldsymbol{H} \in \mathbb{R}^{p \times n}$ as in:

$$\mathbf{y} = \boldsymbol{H}\mathbf{x} + \varepsilon^o \tag{3.4}$$

where $\varepsilon^o$ are the observation errors and $\boldsymbol{H}$ is supposed linear $\boldsymbol{R} \in \mathbb{R}^{p \times p}$, $\boldsymbol{B} \in \mathbb{R}^{n \times n}$, the covariance matrices of the observation errors and of the background errors, respectively. Usually, OI aims to update the background state with observations [Henderson, 1975] through Equation **3.5**:

$$\mathbf{x}^{\mathrm{a}} = \mathbf{x}^{\mathrm{b}} + \boldsymbol{K}\left(\mathbf{y} - \boldsymbol{H}\mathbf{x}^{\mathrm{b}}\right) \tag{3.5}$$

$$\text{with} \quad \boldsymbol{K} = \boldsymbol{B}\boldsymbol{H}^{\top}(\boldsymbol{H}\boldsymbol{B}\boldsymbol{H}^{\top} + \boldsymbol{R})^{-1} \in \mathbb{R}^{n \times p} \tag{3.6}$$

where $\boldsymbol{K}$ is the optimal gain according to Best Linear Unbiased Estimator (BLUE), in the sense of the least squares [Asch et al., 2016]. When $\mathbf{x}^{\mathrm{a}}$ is obtained through a direct matrix inversion, we refer to this method as OI. But it can also be derived from a variational minimization of the following cost function [Puntanen and Styan, 1989]:

$$\mathcal{J}(\mathbf{x}^{\mathrm{a}}) = \frac{1}{2}\left(\mathbf{x}^{\mathrm{a}} - \mathbf{x}^{\mathrm{b}}\right)^{\top} \boldsymbol{B}^{-1}\left(\mathbf{x}^{\mathrm{a}} - \mathbf{x}^{\mathrm{b}}\right) + \frac{1}{2}\left(\mathbf{y} - \boldsymbol{H}\mathbf{x}^{\mathrm{a}}\right)^{\top} \boldsymbol{R}^{-1}\left(\mathbf{y} - \boldsymbol{H}\mathbf{x}^{\mathrm{a}}\right) \tag{3.7}$$

$$= \frac{1}{2}\left|\left|\mathbf{x}^{\mathrm{a}} - \mathbf{x}^{\mathrm{b}}\right|\right|_{\boldsymbol{B}}^2 + \frac{1}{2}\left|\left|\mathbf{y} - \boldsymbol{H}\mathbf{x}^{\mathrm{a}}\right|\right|_{\boldsymbol{R}}^2 \tag{3.8}$$

where $||x - y||_{\boldsymbol{\Sigma}}^2$ is the Mahalanobis distance between $x$ and $y$ associated with the covariance matrix $\boldsymbol{\Sigma}$. We call 3 Dimensional Variational data assimilation (3D-VAR) the variational optimization of this loss function.

#### DUACS

The Data Unification and Altimeter Combination System (DUACS) is one of the most used interpolated SSH products. Its protocol consists of the acquisition of observations from every nadir-pointing altimeter, production of L3 SSH data, and finally, their interpolation [Pujol et al., 2016, Taburet et al., 2019]. While preprocessing satellite measurements is a difficult task leveraging expert knowledge,

we focus on the last step of the process, i.e., the interpolation. DUACS performs an OI of the satellite tracks, of which we detail the general setting hereafter.

The interpolation method used in DUACS is a special case of the OI previously described, in which we aim to estimate a unique value $v$ from $p$ observations [Bretherton et al., 1976,Traon et al., ,Pujol et al., 2016,Taburet et al., 2019]. This corresponds to estimating the value of one pixel, but to produce a complete field, this estimation will be repeated multiple times, each time for the value of another pixel at a new location. We consider that $\mathbf{x}$ is the vector composed by $v$ and the ground truth values at the locations of the observations, $\mathbf{x} = (v, x_1, ..., x_p) \in \mathbb{R}^{p+1}$. Each observation is then $y_i = x_i + \varepsilon_i$, leading to the observation vector $\mathbf{y} = (x_1 + \varepsilon_1, ..., x_p + \varepsilon_p) \in \mathbb{R}^p$. The observation errors are considered independent from the state: $\forall (i,j), \mathrm{cov}(x_i, \varepsilon_j) = \mathrm{cov}(v, \varepsilon_j) = 0$. The estimator for the unobserved value is given by:

$$\widehat{v} = \sum_{i=1}^{p} \sum_{j=1}^{p} A_{ij}^{-1} C_i y_i \tag{3.9}$$

where $\boldsymbol{A}$ is the covariance matrix between observations $A_{i,j} = \mathrm{cov}(y_i, y_j) = \mathrm{cov}(x_i, x_j) + \mathrm{cov}(\varepsilon_i, \varepsilon_j)$ and $C_i = \mathrm{cov}(v, y_i) = \mathrm{cov}(v, x_i)$ is the covariance between the observations and the point $v$ to estimate.

DUACS OI focuses on SLA interpolation, meaning that the background is the MDT which is already removed from observations. Therefore $\mathbf{x}^{\mathrm{b}} = 0$, and the covariance matrix of the background error $\boldsymbol{B}$ becomes the covariance matrix of $\mathbf{x}$. In this setting the observation matrix $\boldsymbol{H} \in \mathbb{R}^{p \times p+1}$ and the background matrix $\boldsymbol{B} \in \mathbb{R}^{p+1 \times p+1}$ become

$$\boldsymbol{H} = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix} \tag{3.10}$$

$$\boldsymbol{B} = \begin{pmatrix} \mathrm{cov}(v,v) & \mathrm{cov}(v,x_1) & \cdots & \mathrm{cov}(v,x_p) \\ \mathrm{cov}(x_1,v) & \mathrm{cov}(x_1,x_1\rangle & \cdots & \mathrm{cov}(x_1,x_p) \\ \vdots & & \ddots & \vdots \\ \mathrm{cov}(x_p,v) & \cdots & \cdots & \mathrm{cov}(x_p,x_p) \end{pmatrix} \tag{3.11}$$

and the analyzed state from Equation **3.5** becomes $\mathbf{x}^{\mathrm{a}} = \boldsymbol{K}\mathbf{y}$. The matrix $\boldsymbol{A}$ from Equation **3.9** can be identified to $(\boldsymbol{H}\boldsymbol{B}\boldsymbol{H}^\top + \boldsymbol{R})$ where $\mathrm{cov}(x_i, x_j)$ accounts for $\boldsymbol{H}\boldsymbol{B}\boldsymbol{H}^\top$ and $\mathrm{cov}(\varepsilon_i, \varepsilon_j)$ for $\boldsymbol{R}$. In the end, Equation **3.9** is the first coordinate of the analyzed state, where $C$ is the first line of the matrix $\boldsymbol{B}\boldsymbol{H}^\top$.

### Tuning covariance matrices

The described methodology thus requires the state's covariance, $\mathrm{cov}(x_i, x_j)$, and observation errors covariance $\mathrm{cov}(\varepsilon_i, \varepsilon_j)$. In the DUACS product, the SSH state correlation function is given by $C_{corr}$ defined in [Arhan and Verdiére, 1985] as:

$$\mathrm{cov}(x_i, x_j) = C_{corr}(dx, dy, dt) = \left[1 + ar + \frac{(ar)^2}{6} - \frac{(ar)^3}{6}\right] e^{-ar} e^{-dt^2/T^2} \tag{3.12}$$

$$\text{with} \quad r = \sqrt{\left(\frac{dx - C_{px}dt}{L_x}\right)^2 + \left(\frac{dy - C_{py}dt}{L_y}\right)^2} \quad \text{and} \quad a = 3.337 \tag{3.13}$$

where $dx, dy, dt$ are the distance in longitude, latitude, and time between $x_i$ and $x_j$, $L_x$ and $L_y$ are the spatial correlations radius, $C_{px}$ and $C_{py}$ are the spatial propagation speeds and $T$ is the temporal correlation radius. All these values are functions of the latitude and longitude and are tuned on 25 years of observations [Pujol et al., 2016, Taburet et al., 2019].

Concerning the observations error correlation, the DUACS protocol differentiates two scenarios. If two observations $y_i$ and $y_j$ with $i \neq j$ come from different satellites their errors are not correlated: $\mathrm{cov}(\varepsilon_i, \varepsilon_j) = b^2\delta_{i,j}$. On the other hand, if the two observations come from the same satellite, they might have long wavelength errors, therefore: $\mathrm{cov}(\varepsilon_i, \varepsilon_j) = b^2\delta_{i,j} + E_{LW}$. $b^2$ and $E_{LW}$ are the error variance and long wave error correlations, estimated on years of data for each satellite.

### Limitations

We can state several limitations of the DUACS product. [Amores et al., 2018] performed an OSSE, mimicking DUACS interpolation on the North Atlantic and the Mediterranean sea. Using the output data from a physical model, the Nucleus for European Modelling of the Ocean (NEMO), they emulated nadir-pointing along-track observations and used an OI to produce L4 altimetry maps, similarly to DUACS. The hyper-parameters of the interpolation method are adjusted so that the obtained maps are close to the one retrieved by DUACS. This *pseudo-duacs* is then used by the authors to characterize the capabilities of DUACS in resolving mesoscale structures and eddies (see Section **5.4.4** for more details). The authors found that $83\%$ of the eddies were not resolved by the OI in the Atlantic. Among the resolved eddies, the larger ones (radius > 50 km) were over-represented in the *pseudo-duacs* map, as small eddies are merged into wider ones. The same conclusions are drawn by the authors in the Mediterranean Sea.

Using a similar OSSE methodology but with the exact DUACS processing chain, [Stegner et al., 2021] found similar results on the Mediterranean Sea. Their statistical analysis reveals an asymmetry in cyclonic-anticyclonic eddies detection, with less reliable cyclones than anticyclones. Small eddies are missed, and large cyclones (radius $>$ 45 km) are mainly obtained by aggregating two or more smaller cyclones.

Overall, we have strong evidence that DUACS OI produces overly smooth SSH fields, leading to overestimating the large eddies and underestimating smaller ones. This is also confirmed by our analysis (see Section **5.6**), where we find that on OSE data DUACS has an effective resolution of 149 km in the Gulf Stream area, which is too low to retrieve small structures. Enhancing DUACS quality is thus an open problem.

### 3.2.2 Kalman filter

**Sequential data assimilation.** In the last section, we focused on the BLUE analysis of a physical system at a given instant, every data having no temporal dimension. However, in oceanography, we are interested in including this temporal dimension to represent the evolution of the physical system. Let's then consider that the physical state dynamics is given by $\mathcal{M}_{\mathrm{t}}$, the *dynamical model*. Equation **3.4** becomes:

$$\mathbf{x}_{\mathrm{t}} = \mathcal{M}_{\mathrm{t}}\left(\mathbf{x}_{\mathrm{t}-1}\right) + \varepsilon_t^m \tag{3.14}$$

$$\mathbf{y}_{\mathrm{t}} = \mathcal{H}_{\mathrm{t}}\left(\mathbf{x}_{\mathrm{t}}\right) + \varepsilon_t^o \tag{3.15}$$

where index $t$ indicates the time step, and $\varepsilon_t^m$ the model errors associated to covariance matrix $\boldsymbol{Q}_t$ . We consider the operators $\mathcal{M}_{\mathrm{t}}$ and $\mathcal{H}_{\mathrm{t}}$ linear, and are represented by matrices $\boldsymbol{M}_t$ and $\boldsymbol{H}_t$. In this framework, one can apply iteratively the BLUE analysis described earlier in Equations **3.5**, **3.6**, and the forecast Equation **3.14**. This process is named the Kalman filter [Asch et al., 2016, Bertino et al., 2003].

**Analysis.** At time $t$ we have access to the forecast from the previous timestep that will constitute our background $\mathbf{x}_{\mathrm{t}}^{\mathrm{b}}$, with covariance matrix $\boldsymbol{B}_t$. We perform the BLUE analysis following Equation **3.5**:

$$\mathbf{x}_{\mathrm{t}}^{\mathrm{a}} = \mathbf{x}_{\mathrm{t}}^{\mathrm{b}} + \boldsymbol{K}_t\left(\mathbf{y}_{\mathrm{t}} - \boldsymbol{H}_t\mathbf{x}_{\mathrm{t}}^{\mathrm{b}}\right) \tag{3.16}$$

$$\text{with} \quad \boldsymbol{K}_t = \boldsymbol{B}_t\boldsymbol{H}_t^{\top}(\boldsymbol{H}_t\boldsymbol{B}_t\boldsymbol{H}_t^{\top} + \boldsymbol{R}_t)^{-1} \text{ the Kalman gain} \tag{3.17}$$

and the covariance matrix associated to this analysis is $\mathbf{A}_\mathrm{t} = (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{H}_t)\boldsymbol{B}_t$

**Forecast.** Given the analyzed state from the current timestep, we can forecast the next background state as: $\mathbf{x}^\mathrm{b}_{\mathrm{t}+1} = \boldsymbol{M}_{t+1}\mathbf{x}^\mathrm{a}_\mathrm{t}$. Then we update the covariance matrix of the next background state as $\boldsymbol{B}_{t+1} = \boldsymbol{M}_{t+1}\boldsymbol{A}_t\boldsymbol{M}_{t+1}^\top + \boldsymbol{Q}_{t+1}$. The Kalman filtering is presented in Algorithm **1**.

**Algorithm 1** Kalman Filter

**Inputs:** $\mathbf{x}^\mathrm{b}_0$ and its covariance matrix $\boldsymbol{B}_0$, dynamic and observation models $\boldsymbol{M}$ and $\boldsymbol{H}$, observations $\mathbf{y}$, and covariance matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, at every timestep from $0$ to $T$.
**for** $t = 0$ to $T$ **do**
  $\boldsymbol{K}_t \leftarrow \boldsymbol{B}_t\boldsymbol{H}_t^\top(\boldsymbol{H}_t\boldsymbol{B}_t\boldsymbol{H}_t^\top + \boldsymbol{R}_t)^{-1}$ ▷ Compute Kalman gain
  $\mathbf{x}^\mathrm{a}_\mathrm{t} \leftarrow \mathbf{x}^\mathrm{b}_\mathrm{t} + \boldsymbol{K}_t\left(\mathbf{y}_\mathrm{t} - \boldsymbol{H}_t\mathbf{x}^\mathrm{b}_\mathrm{t}\right)$ ▷ Analysis
  $\mathbf{A}_\mathrm{t} = (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{H}_t)\boldsymbol{B}_t$ ▷ Compute analysis covariance matrix
  $\mathbf{x}^\mathrm{b}_{\mathrm{t}+1} \leftarrow \boldsymbol{M}_{t+1}\mathbf{x}^\mathrm{a}_\mathrm{t}$ ▷ Forecast
  $\boldsymbol{B}_{t+1} = \boldsymbol{M}_{t+1}\boldsymbol{A}_t\boldsymbol{M}_{t+1}^\top + \boldsymbol{Q}_{t+1}$ ▷ update background covariance matrix
**end for**

### 3.2.3 4D-VAR

In the 4 Dimensional Variational data assimilation (4D-VAR) setting, the observing and dynamic models are not necessarily linear, and we consider a window of observations from time $0$ to $T$. We assume a perfect dynamical model, meaning that the model error $\varepsilon^m_t = 0$. This situation is called strong constrained 4D-VAR [Dimet and Talagrand, 1986]. The 3D-VAR cost function (Equation **3.8**) can be extended to:

$$\mathcal{J}(\mathbf{x}_0) = \frac{1}{2}\left|\left|\mathbf{x}_0 - \mathbf{x}^\mathrm{b}\right|\right|^2_{\boldsymbol{B}} + \frac{1}{2}\sum_{t=0}^{T}||\varepsilon^o_t||^2_{\boldsymbol{R}_t} \tag{3.18}$$

$$= \frac{1}{2}\left|\left|\mathbf{x}_0 - \mathbf{x}^\mathrm{b}\right|\right|^2_{\boldsymbol{B}} + \frac{1}{2}||\mathcal{H}_0(\mathbf{x}_0) - \mathbf{y}_0||^2_{\boldsymbol{R}_0} + \frac{1}{2}\sum_{t=1}^{T}||\mathbf{y}_\mathrm{t} - \mathcal{H}_\mathrm{t}\circ\mathcal{M}_{0\to\mathrm{t}}(\mathbf{x}_0)||^2_{\boldsymbol{R}_t} \tag{3.19}$$

where $\mathcal{M}_{0\to\mathrm{t}} = \mathcal{M}_0 \circ \mathcal{M}_1 ... \circ \mathcal{M}_{\mathrm{t}-1}$ is the dynamic model applied at each time step between $1$ and $t$. Because the dynamic is fully known, estimating the initial

condition determines the full trajectory. The control parameters are in $\mathbf{x}_0$, and we can compute the gradient of $\mathcal{J}$ as :

$$\nabla_{\mathbf{x}_0}\mathcal{J} = \boldsymbol{B}^{-1}(\mathbf{x}_0 - \mathbf{x}^{\mathrm{b}}) + \frac{\partial \mathcal{H}_0(\mathbf{x}_0)}{\partial \mathbf{x}_0}^{\top} \boldsymbol{R}_0^{-1}\varepsilon_0^o + \sum_{t=1}^{T}\left(\frac{\partial(\mathcal{H}_{\mathrm{t}} \circ \mathcal{M}_{0\to \mathrm{t}}(\mathbf{x}_0)}{\partial \mathbf{x}_0}\right)^{\top} \boldsymbol{R}_t^{-1}\varepsilon_t^o \tag{3.20}$$

Computing this gradient requires the adjoint models of the linear tangent models of $\mathcal{M}$ and $\mathcal{H}$, which can be difficult to obtain in complex physical models. Nowadays, it is common to use automatic differentiation software such as Tapenade, Tensorflow, and Pytorch, which provide the adjoint when the direct model is coded. Compared to Kalman filters that only required covariances matrices and direct models, these can be seen as a drawback for the 4D-VAR method. The variational optimization of 4D-VAR is given in Algorithm **2**.

**Algorithm 2** Strong Constrained 4D-VAR

**Inputs:** $\mathbf{x}^{\mathrm{b}}$, background state, $\boldsymbol{B}$ and $\boldsymbol{R}$, background and observations error covariance matrices, $\mathcal{M}$ and $\mathcal{H}$, dynamic and observation models, and $\mathbf{y}$ observations between $t = 0$ to $T$.
Initialize $\mathbf{x}_0$
**while** not converged **do**
  $\mathcal{J} \leftarrow \left|\left|\mathbf{x}_0 - \mathbf{x}^{\mathrm{b}}\right|\right|^2_{\boldsymbol{B}} + ||\mathbf{y}_0 - \mathcal{H}_0(\mathbf{x}_0)||^2_{\boldsymbol{R}_0}$
  **for** t=1 to T **do**
    $\mathbf{x}_{\mathrm{t}} \leftarrow \mathcal{M}_{\mathrm{t}-1}(\mathbf{x}_{\mathrm{t}-1})$
    $\mathcal{J} \leftarrow \mathcal{J} + ||\mathbf{y}_{\mathrm{t}} - \mathcal{H}_{\mathrm{t}}(\mathbf{x}_{\mathrm{t}})||^2_{\boldsymbol{R}_t}$
  **end for**
  $\mathbf{x}_0 \leftarrow \mathbf{x}_0 - \alpha\nabla_{\mathbf{x}_0}\mathcal{J}$
**end while**

### 3.2.4 Nudging methods

#### Principle

Nudging methods are simple data assimilation techniques aiming to push the model trajectory toward observations [Asch et al., 2016]. It implies introducing a *nudging* or *feedback* term in the dynamic model equations, which is proportional to the error between the model and observations. Let $\mathbf{x} \in \mathbb{R}^n$ be the state vector defined as a function of $t \in [0, T]$. The evolution of the system is given by the dynamical model $\mathcal{M}$ as follows:

$$\frac{d\mathbf{x}}{dt} = \mathcal{M}(\mathbf{x}(t)), \quad \mathbf{x}(t = 0) = \mathbf{x}_0 \tag{3.21}$$

Let us say that we have access to a set of observations distributed over time $\mathbf{y} \in \mathbb{R}^p$ and to the observation operator $\mathcal{H}$. The nudging method consists of modifying the dynamical equations as follows:

$$\frac{d\mathbf{x}}{dt} = \mathcal{M}(\mathbf{x}(t)) - \boldsymbol{K}(\mathcal{H}(\mathbf{x}(t)) - \mathbf{y}(t)), \quad \mathbf{x}(t=0) = \mathbf{x}_0 \tag{3.22}$$

where $\boldsymbol{K} \in \mathbb{R}^{n \times p}$ is the nudging coefficient matrix. If these coefficients are very large the trajectory will fit the observations, but if $\boldsymbol{K}$ is small the trajectory is purely driven by the model dynamics. $\boldsymbol{K}$ is therefore a non-trivial experimental parameter that will require accurate tuning. Looking at Equation **3.22**, this approach can be seen as a degraded Kalman filter, where the analysis step is performed with a sub-optimal fixed gain, which is a drawback of the method. However, nudging is very light to implement as it does not require covariances matrices of the Kalman filter, nor the adjoints models of 4D-VAR.

### Back and Forth Nudging

Back and Forth Nudging (BFN) is an iterative method introduced by [Auroux and Blum, 2008] extending the nudging principle. It consists of a forward scheme, as in Equation **3.22**, and a backward scheme, which corresponds to the model going back in time. We write the backward equation using the final condition instead of the initial one:

$$\frac{d\mathbf{x}}{dt} = \mathcal{M}(\mathbf{x}(t)), \quad \mathbf{x}(t=T) = \mathbf{x}_{\mathrm{T}} \tag{3.23}$$

Given the variable change $t' = T - t$, we rewrite it as :

$$\frac{d\mathbf{x}}{dt'} = -\mathcal{M}(\mathbf{x}(t')), \quad \mathbf{x}(t'=0) = \mathbf{x}_{\mathrm{T}} \tag{3.24}$$

and the nudging equation becomes

$$\frac{d\mathbf{x}}{dt'} = -\mathcal{M}(\mathbf{x}(t')) - \boldsymbol{K}'(\mathcal{H}(\mathbf{x}(t')) - \mathbf{y}(t')), \quad \mathbf{x}(t'=0) = \mathbf{x}_{\mathrm{T}} \tag{3.25}$$

which we rewrite:

$$\frac{d\tilde{\mathbf{x}}}{dt} = \mathcal{M}(\tilde{\mathbf{x}}(t)) + \boldsymbol{K}'(\mathcal{H}(\tilde{\mathbf{x}}(t)) - \mathbf{y}(t)), \quad \tilde{\mathbf{x}}(t=T) = \mathbf{x}_{\mathrm{T}} \tag{3.26}$$

where $\tilde{\mathbf{x}}$ is the state function going backward in time [Asch et al., 2016]. This method requires "inverting" the dynamical model, which is generally ill-posed, for instance, when $\mathcal{M}$ contains a form of diffusion. In the backward scheme, the nudging term

has a supplementary role compared to its forward equivalent, as it also stabilizes the temporally inverted model.

BFN (Algorithm **3**) applies successively forward and backward nudging until convergence. The initialization starts from $\mathbf{x}^{\mathrm{b}}$, an arbitrarily chosen first guess. The forward nudging then generates a new model trajectory, and its last time step is used as the initial condition of the backward. This process is repeated until the algorithm converges, meaning that the back-and-forth trajectories are close.

**Algorithm 3** Back and Forth Nudging (BFN)

**Inputs:** $\mathbf{x}^{\mathrm{b}}$, background state, $\mathcal{M}$ and $\mathcal{H}$, dynamic and observation models, and $\mathbf{y}$ observations between $t = 0$ to $T$.
$\tilde{\mathbf{x}} \leftarrow \mathbf{x}^{\mathrm{b}}$
$\mathbf{x} \leftarrow +\infty$
**while** not converged **do**

$\mathbf{x}(0) \leftarrow \tilde{\mathbf{x}}(0)$
**for** $t \in [0 : T-1]$ **do**
$\mathbf{x}(t+1) \leftarrow \mathbf{x}(t) + \Delta t \left[\mathcal{M}(\mathbf{x}(t)) - \boldsymbol{K}\left(\mathcal{H} \circ \mathcal{M}(\mathbf{x}(t)) - \mathbf{y}(t+1)\right)\right]$
**end for**

$\tilde{\mathbf{x}}(T) \leftarrow \mathbf{x}(T)$
**for** $t \in [T : 1]$ **do**
$\tilde{\mathbf{x}}(t-1) \leftarrow \tilde{\mathbf{x}}(t) + \Delta t \left[\mathcal{M}(\tilde{\mathbf{x}}(t)) + \boldsymbol{K'}\left(\mathcal{H} \circ \mathcal{M}(\tilde{\mathbf{x}}(t)) - \mathbf{y}(t-1)\right)\right]$
**end for**
**end while**

## 3.3 Deep Learning

Machine Learning (ML) is an extensive set of statistical methods able to learn complex tasks leveraging a dataset [Mahesh, 2018]. The base principle in ML is to automatically adjust an algorithm to perform a task by tuning parameters (weights of linear regression, filter values ...) and hyper-parameters (number of iterative steps, input variables ...), using data-driven knowledge. Among these techniques, Deep Learning (DL) has emerged as one of the leading methods [Goodfellow et al., 2016, Lecun et al., 2015]. DL is an ensemble of techniques in which the parameters are organized in an Artificial Neural Network (ANN), mimicking the brain structure. An ANN is composed of neurons, which take a series of inputs, ponder it with weights, and apply a non-linear *activation* function, indicating if the neuron is activated or not. The results are passed to the next neurons, regrouped in layers, in a so-called Multi-Layer Perceptron (MLP) [Popescu et al., 2009]. DL extend this principle

by stacking large numbers of neurons, modifying their different elements, such as their activation functions, connections between layers, regularization processes, and optimization algorithms. DL has shown a significant capacity to solve ill-posed inverse problems [Goodfellow et al., 2016, Ongie et al., 2020, Jam et al., 2021, Qin et al., 2021].

This section first discusses the different training methodologies used in DL and how we can use them in SR and interpolation. Then, we give a non-exhaustive overview of the neural network architectures. While not all of them were used in our studies, they have been in use in the field and are included in the manuscript to facilitate the discussion of architectural choices in Chapters **4**, **5**, **6** as well as prepare the perspectives section of this thesis.

### 3.3.1 Deep Neural network to solve ill-posed inverse problems

#### Supervised and unsupervised learning

We aim to inverse the forward operator $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ from Equation **3.1** using a function $f : \mathcal{Y} \to \mathcal{X}$, parameterized by $\theta \in \mathbb{R}^k$. In this part, $\mathcal{X}$ is the state space, and $\mathcal{Y}$ is the observation space, which can be a vector, an image, or a time series of images without lack of generality. The neural network $f_\theta$ estimates the state $\mathbf{X}$ from observations $\mathbf{Y}$ as:

$$\widehat{\mathbf{X}} = f_\theta\left(\mathbf{Y}\right) \tag{3.27}$$

The DL strategy is to fit $\theta$ using the knowledge of a dataset $\mathcal{D}$. We can distinguish two different learning settings: supervised and unsupervised learning.

In a **supervised learning** context, we have access to pairs of observations and associated ground truth ($\mathcal{D} = \{\mathbf{Y}_\mathrm{i}, \mathbf{X}_\mathrm{i}\}$). Fitting $\theta$ usually requires defining $\mathcal{L} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, a loss function that is a distance between the true state and its estimation. The optimal value of $\theta$ is found by minimizing $\mathcal{L}$ on $\mathcal{D}$:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}} \sum_{(\mathbf{X},\mathbf{Y})\in\mathcal{D}} \mathcal{L}\left(\mathbf{X}, f_\theta(\mathbf{Y})\right) \tag{3.28}$$

In practice, this minimization is performed through stochastic gradient descent, a variant of gradient descent where the loss is computed on a subset of the dataset (a batch) [Bottou, 2010]. The parameters are updated using the back-propagation algorithm [Hecht-Nielsen, 1992] by computing the loss gradient on this small batch

of data. The weights are modified starting from the last layer by applying the chain rule layer by layer. This process is repeated until convergence.

In an **unsupervised learning** context, only observations are available ($\mathcal{D} = \{\mathbf{Y}_i\}$), and in this case, other strategies must be considered. However, when the forward operator $\mathcal{F}$ is known, observations $\mathbf{Y}$ can serve as a proxy for the ground truth. In this setting, called **self-supervised learning**, we can train neural networks from observations only given a proper modification of the loss function [Ongie et al., 2020]. As the forward model is known, we can apply it on the estimated state $\widehat{\mathbf{X}}$. Equation **3.28** becomes:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{\mathbf{Y} \in \mathcal{D}} \mathcal{L}\left(\mathbf{Y}, \mathcal{F}(f_\theta(\mathbf{Y}))\right) = \underset{\theta}{\operatorname{argmin}} \sum_{\mathbf{Y} \in \mathcal{D}} \mathcal{L}\left(\mathbf{Y}, \mathcal{F}(\widehat{\mathbf{X}})\right) \tag{3.29}$$

where the loss function is now $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. This strategy was used, for instance, to train denoising auto-encoders [Tamir et al., 2019, Yaman et al., 2019].

### Deep Image Prior

While in supervised and unsupervised settings the neural network learns on a dataset, it is also possible to fit a neural network on a **single example** of observations. This methodology, called Deep Image Prior (DIP) [Ulyanov et al., 2017], consists of starting from a random vector $\mathbf{Z}$, sampled once for each image, with low dimension, as an input of a generator. Similarly to self-supervised methods, the network produces an estimate of the state $\widehat{\mathbf{X}}$ on which is applied the forward operator. The optimization is then performed on the network weights on a single example as:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}\left(\mathbf{Y}, \mathcal{F}(f_\theta(\mathbf{Z})\right) = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}\left(\mathbf{Y}, \mathcal{F}(\widehat{\mathbf{X}})\right) \tag{3.30}$$

DIP optimization is presented in Algorithm **4**. Given some conditions on the optimization, [Ulyanov et al., 2017] showed that this method produced accurate images for diverse image inverse problems, such as denoising, inpainting, and super-resolution. To the authors, the interpretation of this surprising result is that neural architectures produce spatially correlated images, and thus promote certain solutions over others. For example, in the denoising case study, the neural network has great trouble fitting an uncorrelated noise image where it can more easily fit a natural one. Therefore, if an early stopping is applied to the optimization, i.e., stopping the gradient descent before convergence, the natural smooth image will be reconstructed, but the noise errors will not. Neural network architecture has a prior on the images and acts as a

regularization. Different architectures bring different inductive biases [Cohen and Shashua, 2017, Rahaman et al., 2019], further discussed in Section **3.3.2**.

**Algorithm 4** Deep Image Prior

**Inputs:** observations $\mathbf{Y}$, learning rate $\alpha$
Initialize $\theta$ and $\mathbf{Z}$ randomly
**while** not converged **do**
$\widehat{\mathbf{X}} \leftarrow f_\theta(\mathbf{Z})$
$\widehat{\mathbf{Y}} \leftarrow \mathcal{F}(\widehat{\mathbf{X}})$
$\theta \leftarrow \theta - \alpha \nabla_\theta \left|\left|\widehat{\mathbf{Y}} - \mathbf{Y}\right|\right|^2$
**end while**

### Generative deep learning

In this section, we take a step back from inverse problem-solving to present deep generative networks. We then explain how these models can be used in image inpainting or super-resolution. In generative modeling, we aim to generate samples of a complex, intractable, high-dimension probability distribution $\mathcal{X}$ from a tractable distribution $\mathcal{Z}$ [Ruthotto and Haber, 2021]. The so-called generator $g : \mathcal{Z} \to \mathcal{X}$ is a function mapping a sample $\mathbf{Z} \sim \mathcal{Z}$ to a corresponding sample from the intractable distribution $\mathbf{X} \sim \mathcal{X}$. In the following, we detail two generative frameworks that received a lot of attention in the last years: Generative Adversarial Network (GAN) and Denoising Diffusion Probabilistic Models (DDPM).

**Generative Adversarial Networks** were introduced by [Goodfellow et al., 2014], and represent a very significative contribution to image synthesis, SR and inpainting, among others. They require training simultaneously two networks: a generator $g_\theta$ trying to fool a discriminator $d_\phi$. Starting from a random vector $\mathbf{Z}$, the generator outputs a sample that should belong to the target distribution $\mathcal{X}$. Then, the discriminator takes the generated sample and a real sample of the distribution drawn from the training dataset and must decide which of the two examples is fake. The discriminator should produce a scalar value close to $1$ if the sample is real and $0$ if not. With this mindset, the two networks are "adversarial" and focus on the same loss function:

$$\mathcal{L}_{\text{GAN}}(g_\theta, d_\phi) = \mathbb{E}_{\mathbf{X}\sim\mathcal{X}}[\log d_\phi(\mathbf{X})] + \mathbb{E}_{\mathbf{Z}\sim\mathcal{Z}}[\log(1 - d_\phi \circ g_\theta(\mathbf{Z}))] \tag{3.31}$$

which the discriminator tries to maximize and the generator to minimize. The training process of the GAN is described in Algorithm **5**.

**Algorithm 5** Generative Adversarial Network Training

**Inputs:** Dataset $\mathcal{D}$, Learning rate $\alpha$, distribution $\mathcal{Z}$
Initialize discriminator parameters $\phi$ and generator parameters $\theta$ randomly
**while** not converged **do**
  Sample $\mathbf{X}$ from $\mathcal{D}$
  Sample $\mathbf{Z}$ from $\mathcal{Z}$
  $\phi \leftarrow \phi + \alpha \nabla_\phi \left[\log d_\phi(\mathbf{X}) + \log(1 - d_\phi \circ g_\theta(\mathbf{Z}))\right]$
  $\theta \leftarrow \theta - \alpha \nabla_\theta \left[\log(1 - d_\phi \circ g_\theta(\mathbf{Z}))\right]$
**end while**

**Denoising Diffusion Probabilistic Models (DDPM)** is another kind of generative model introduced by [Ho et al., 2020] that ended GAN hegemony in image synthesis and many other generative tasks [Yang et al., 2023]. DDPM use two different Markov chains: the *forward process* and the *backward process*. The first process progressively adds Gaussian noise to the data $\mathbf{X}_0$ such as :

$$q(\mathbf{X}_\mathrm{t}|\mathbf{X}_{\mathrm{t}-1}) = \mathcal{N}(\mathbf{X}_\mathrm{t}; \sqrt{1-\beta_t}\mathbf{X}_{\mathrm{t}-1}, \beta_t \boldsymbol{I}), \quad q(\mathbf{X}_{1:\mathrm{T}}|\mathbf{X}_0) = \prod_{t=1}^{T} q(\mathbf{X}_\mathrm{t}|\mathbf{X}_{\mathrm{t}-1}) \tag{3.32}$$

where $t$ is the current noising step, and $\beta_t$ is the variance of the added noise at time $t$. The *inverse process* is a step-by-step denoising parametrized by a neural network with weights $\theta$.

$$p_\theta(\mathbf{X}_{\mathrm{t}-1}|\mathbf{X}_\mathrm{t}) = \mathcal{N}(\mathbf{X}_{\mathrm{t}-1}; \mu_\theta(\mathbf{X}_\mathrm{t}, t), \beta_t \boldsymbol{I}) \quad p_\theta(\mathbf{X}_{0:\mathrm{T}}) = p(\mathbf{Z}) \prod_{t=1}^{T} p_\theta(\mathbf{X}_{\mathrm{t}-1}|\mathbf{X}_\mathrm{t}) \tag{3.33}$$

One interesting feature of the forward process is that if $\beta_t$ is small, the $\mathbf{X}_\mathrm{t}$ can be expressed from $\mathbf{X}_0$ directly [Ho et al., 2020]. If we write $\alpha_t = 1 - \beta_t$ and $\alpha_{1:t} = \prod_{i=1}^{t} \alpha_i$. in this case, $q(\mathbf{X}_\mathrm{t}|\mathbf{X}_0) = \mathcal{N}(\mathbf{X}_\mathrm{t}; \sqrt{\alpha_{1:t}}\mathbf{X}_0, (1-\alpha_{1:t})\boldsymbol{I})$. This is a very important feature, as it means that the training of the denoising network can be performed one step at a time as it is possible to obtain a sample for $\mathbf{X}_\mathrm{t}$ and $\mathbf{X}_{\mathrm{t}-1}$ directly from the data. In practice, the neural network tries to estimate the input noise $\mathbf{Z}$ and is called $z_\theta$ of which we present the training method in Algorithm **6**. To generate a new sample from a trained DDPM, we start from a random sample and compute the $T$ steps of the inverse process as shown in Algorithm **7**.

**Conditional generation.** To use these generative frameworks to solve inverse problems, one must find a way to inform the generation with observations $\mathbf{Y}$. This is called conditional generation, as giving $\mathbf{Y}$ to generative models makes it learn the

**Algorithm 6** DDPM training

**Inputs:** Dataset $\mathcal{D}$, $\alpha_t$ schedule, learning rate $\alpha$
Initialize $\theta$ randomly
**while** not converged **do**
  Sample $\mathbf{X}_0$ from $\mathcal{D}$
  $t \sim$ Uniform$(\{1, \dots, T\})$
  $\mathbf{Z} \sim \mathcal{N}(0, \boldsymbol{I})$
  $\theta \leftarrow \theta - \alpha \nabla_\theta \left|\left|\mathbf{Z} - z_\theta \left(\sqrt{\alpha_{1:t}}\mathbf{X}_0 + \sqrt{1-\alpha_{1:t}}\mathbf{Z}, t\right)\right|\right|^2$
**end while**

**Algorithm 7** DDPM Generation

$\mathbf{X}_\mathrm{T} \sim \mathcal{N}(0, I)$ ▷ $\mathbf{X}_\mathrm{T}$ plays the role of z for the first step
$\mathbf{Z} \leftarrow 0$
**for** $t = T$ **to** 1 **do**
  $\mathbf{X}_{\mathrm{t}-1} = \sqrt{\frac{1}{\alpha_t}}\left(\mathbf{X}_\mathrm{t} - \frac{1-\alpha_t}{\sqrt{1-\alpha_{1:t}}} z_\theta(\mathbf{X}_\mathrm{t}, t)\right) + \sqrt{1-\alpha_t} z$
  $\mathbf{Z} \sim \mathcal{N}(0, I)$
**end for**
**return** $\mathbf{X}_0$

conditional distribution $\mathbf{X}$ knowing $\mathbf{Y}$. Adapting this principle to GAN, [Mirza and Osindero, 2014] introduced Conditional Generative Adversarial Network (CGAN). Both the generator and the discriminator are informed of the observations and $\mathcal{L}_{\mathrm{GAN}}$ from Equation **3.31** becomes:

$$\mathcal{L}_{\mathrm{CGAN}}(g_\theta, d_\phi) = \mathbb{E}_{\mathbf{X}\sim\mathcal{X}}[\log d_\phi(\mathbf{X}|\mathbf{Y})] + \mathbb{E}_{\mathbf{Z}\sim\mathcal{Z}}[\log(1 - d_\phi \circ g_\theta(z|\mathbf{Y}))] \tag{3.34}$$

One of the benefits of generative modeling to solve ill-posed inverse problems is that the solution is not unique. Therefore, by sampling multiple times the latent variable, we have access to multiple solutions, enabling a more accurate estimation and uncertainty measures [Patel and Oberai, 2021]. These networks have been used in SR applications [Ledig et al., 2017, Wang et al., 2018, Wang et al., 2021], and in inpainting [Jam et al., 2021, Qin et al., 2021]. DDPM can also use conditions to guide image reconstruction. [Lugmayr et al., 2022] proposed an inpainting diffusion model that replaces known pixels with their values in the noise image. In doing so, the network progressively reconstructs the missing values only. GANs and DDPMs can even be used together to push further their performances [Xiao et al., 2024].

### Domain Adaptation

In the previously described settings, the data came from a single domain, i.e. observation space $\mathcal{Y}$, a state space $\mathcal{X}$, and a joint probability distribution of state

and observations, $p(\mathbf{Y}, \mathbf{X})$. In practice, data can be collected from several sources, constituting different domains. For instance, in geosciences, we could use data from a physical simulation and from real observations. Transfer learning refers to methods able to adapt the learning of a task on a source domain to a target domain, with possibly different tasks [Pan and Yang, 2010]. Domain adaptation is a special case of transfer learning, where the source and target domains are related but different, and the tasks are identical. We consider the situation where we have access to a source dataset $\mathcal{S} = \{(\mathbf{Y}_\mathrm{i}, \mathbf{X}_\mathrm{i}), i \in [1, m]\}$ and $\mathcal{T} = \{\mathbf{Y}_\mathrm{i}, i \in [1, n]\}$ a target dataset. As only the source dataset is labeled, this framework is called unsupervised domain adaptation [Patel et al., 2015]. While domain adaptation usually focuses on classification tasks [Patel et al., 2015, Farahani et al., 2021], regression and inverse problem domain adaptation have been considered [Cortes and Mohri, 2011, Han et al., 2018, Singhal and Majumdar, 2020]. In Section **5.5** we give an example of domain adaptation applied to SSH interpolation from simulated data to real observations.

#### Ensemble estimation of neural networks

The solution proposed by a group of estimators can be better than the solution of one individual. Applying the *crowd wisdom* principle to machine learning is called *ensemble estimation* [Goodfellow et al., 2016]. The idea is to use several models simultaneously and combine their predictions to provide the final estimation. In image inverse problems, the most simple yet efficient way to combine estimations is to average the images from each member. The resulting image is a more accurate solution and often outperforms even the best model. The reason for this improved performance is that the individual errors are compensated by the other ensemble members.

There are several ways to compute ensembles: training on different datasets, using various architectures... [Mohammed and Kora, 2023]. However, one of the most straightforward approaches is to train several networks with different weight initializations [Hinton and Dean, 2015]. The optimization of neural networks is a stochastic process that typically converges to a sub-optimal local minimum, making it sensitive to the initial weights. This sensitivity can be leveraged to compute an ensemble.

### 3.3.2 Neural architectures

Through the years, many deep learning architectures have been proposed, each one presenting different advantages and drawbacks. In the following, we list some of the most widely used architectures, which we will refer to in our contributions.

#### Convolutionnal neural networks

**Convolutional Neural Network (CNN)**, one of the most used deep learning architectures in image tasks, learns convolution operations able to identify features over space and/or time. These convolutions operation kernels usually have $2$ or $3$ dimensions, and regroup their trainable parameters in a kernel $W$ and a bias $b$. If $I^i$ is an input image of $2$ (eventually $3$) dimensions, the output image is usually computed as: $I^o = W \star I^i + b$, where $\star$ is the convolutional product [Lecun et al., 2015, Albawi et al., 2017]. The convolution operation can detect some features inside an image independently of their locations. Due to the translation invariance of the convolutions, the CNN architectures have a spatial invariant (or spatio-temporal for 3D convolutions) inductive bias [Battaglia et al., 2018].

A variant of the 3D convolutions is the *Conv2PD1* introduced by [Tran et al., 2018] and showed the same performances as Conv3D with lesser computing cost. First, a 2D convolution is performed in the spatial dimensions, followed by a Rectified Linear Unit (ReLU) activation function, and finally a 1D convolution in the time dimension.

**Autoencoders and Encoder Decoder.** An autoencoder is a neural architecture where the input data is compressed to a small latent space before being decoded [Baldi, 2012, Goodfellow et al., 2016]. Depending on the dimensionality of the inputs, the latent space could be a vector, small images, or a time series of small images. Two main benefits can be found in these architectures: first, as the latent vector has lower dimensionality than the initial data, it represents only the most significant information from the input. It is then possible to train denoising autoencoders in a self-supervised way, by using the distance between the input and the output as a loss function. As only the most significant information is encoded, the input noise will be decreased. Second, as spatial dimensions are reduced, it is possible to train multi-resolution features. In CNNs for example, the perceptive field of a filter is spatially limited. In a $3 \times 3$ filter, the information to compute a pixel only comes from the same pixel and its neighbors. Even if it is possible to stack filters to increase the

spatial propagation of information, it is much more efficient to perform convolutions at different resolutions.

**Super-Resolution CNNs.** Super-resolution CNNs are neural architectures able to enhance the resolution of images, providing a higher level of detail. These networks typically employ alternate feature extraction and up-sampling layers until they reach target resolution. The first SRCNN was introduced by [Dong et al., 2014] and is composed of only two layers, one extracting features from the low-resolution image and one mapping the features to the high-resolution image. The upsampling is performed at the beginning of the network by a bicubic interpolation, and the network only learns to refine the details of the upsampled image. Over the years, the architectures evolved toward different upsampling strategies [Wang et al., 2021]. Residual skip connections were introduced [Kim et al., 2016, Lim et al., 2017] (see next paragraph), as well as subpixel convolution [Shi et al., 2017a]. Its principle is to perform the convolution, not in the original high-resolution image $I_{hr} \in \mathbb{R}^{C \times H \times H}$, but in a smaller space with more channels $I_{lr} \in \mathbb{R}^{r^2 C \times H/r \times H/r}$. The upsampling is then performed at the end of the network by a pixel-shuffling operator $P$, which maps all the pixels from $I_{lr}$ to $I_{hr}$ as illustrated in Figure **3.1**. This strategy has two

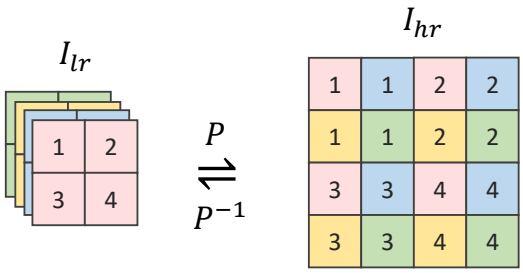


**Fig. 3.1.:** Graphical view of the pixel shuffling operator $\mathcal{P}$ introduced by [Shi et al., 2017a].

main advantages: first, the upsampling is trainable, unlike the bicubic interpolation, which allows learning more complex patterns [Wang et al., 2021]. Second, for the same computational cost, performing the convolution on the subpixel space allows having more parameters than the standard convolution [Shi et al., 2017a, Shi et al., 2017b], enabling video SR, which was its initial case study. To enhance the reconstruction, it is possible to use these architectures as the conditional generator of a CGAN [Ledig et al., 2017].

### Residual and skip connections

To learn the very complex relationship between data the deep learning community developed deeper and deeper neural networks. However, as the parameters are modified following the chain rule, stacking layers and activation functions result in small gradients. This makes it difficult to train the first weights of very deep neural networks. This problem is called the *vanishing gradient problem* [Bengio et al., 1994, Glorot and Bengio, 2010]. A solution to address this problem is the use of Residual neural Network (ResNet) [He et al., 2016]. Residual learning incorporates shortcut connections where the input of a block is combined with its output, facilitating the flow of information through the network. A residual block is composed of a small number of layers and activation function $F$ which is applied to the input $x_i$. The output of the resblock is obtained through $x_o = F(x_i) + x_i$ which forces the network to learn a small modification of its input and avoid vanishing gradient. Another solution is to concatenate the output of intermediate layers with inputs. This so-called *skip connection*, allows shortcuts between layers so that the gradient can better back-propagate during training. One of the most famous networks using this principle is the U-Net [Ronneberger et al., 2015]. The architecture is similar to an autoencoder but introduces skip connections between encoding and decoding blocks, preserving high-resolution signal while also managing to expand the perceptive field of CNNs.

### Recurrent Neural Networks

Recurrent Neural Network (RNN) is a family of neural networks well suited to temporal data representations. Let $\mathbf{y}_t$ be the input sequence and $\mathbf{x}_t$ the target sequence at time $t$. A RNN cell usually takes the input at the current time step ($\mathbf{y}_t$) and the output of the same RNN cell at the previous timestep ($\mathbf{x}_{t-1}$) as shown in Figure **3.2**.

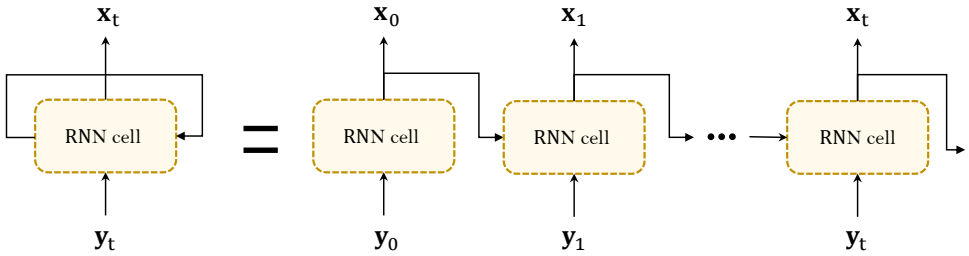


**Fig. 3.2.:** RNN computational graph. An RNN cell can be unrolled into feed-forward network with shared weights.

This recurrent structure is not compatible with back-propagation, and to perform Back Propagation Through Time (BPTT), we usually represent RNN as an unrolled feedforward network with share weights [Goodfellow et al., 2016]. The basic version of RNN uses a linear combination of $\mathbf{x}_{t-1}$ and $\mathbf{y}_t$ followed by an activation function. But this implementation suffers from the vanishing gradient problem: as the same cell is applied a high number of times, the gradient magnitude becomes small, which leads to difficulties in learning long-term dependencies [Bengio et al., 1994].

The Long-Short Term Mermory (LSTM) network introduced by [Hochreiter and Schmidhuber, 1997, Gers et al., 2000] proposes a way to deal with the gradient vanishing problem. It uses a hidden state $\mathbf{c}_t$ which is modified at each timestep, to store or erase information and helps the network to estimate its output. The cell's shape is presented in Figure **3.3**.

$$\mathbf{f}_t = \sigma\left(W_f[\mathbf{x}_{t-1}, \mathbf{y}_t] + b_f\right) \tag{3.35}$$
$$\mathbf{i}_t = \sigma\left(W_i[\mathbf{x}_{t-1}, \mathbf{y}_t] + b_i\right) \tag{3.36}$$
$$\tilde{\mathbf{c}_t} = \tanh\left(W_c[\mathbf{x}_{t-1}, \mathbf{y}_t] + b_c\right) \tag{3.37}$$
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}_t} \tag{3.38}$$
$$\mathbf{o}_t = \sigma\left(W_o[\mathbf{x}_{t-1}, \mathbf{y}_t] + b_o\right) \tag{3.39}$$
$$\mathbf{x}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right) \tag{3.40}$$

We detail hereafter the LSTM equations:

- **The forget gate**: first, the network looks at the output of the last cell $\mathbf{x}_{t-1}$ and the current input $\mathbf{y}_t$, and decides which information to erase from the memory of the last state $\mathbf{c}_{t-1}$. The erasing mechanism is implemented as a sigmoïd layer in Equation **3.35**, meaning that a weight matrix $W_f$ is multiplied to the vector $[\mathbf{x}_{t-1}, \mathbf{y}_t]$ and that a bias $b_f$ is added before using the sigmoid function. As the output of the sigmoid is between 0 and 1, multiplying $\mathbf{f}_t$ with $\mathbf{c}_{t-1}$ will select relevant information to keep in the cell state.

- **The input gate**: we then need to decide which information to add in the hidden state. The estimated state $\tilde{\mathbf{c}_t}$ is calculated from $[\mathbf{x}_{t-1}, \mathbf{y}_t]$ through a hyperbolic tangent layer in Equation **3.37**, and the information to be added to the memory is chosen with a sigmoid layer in Equation **3.36**.

- **State update**: the new hidden state is computed in Equation **3.38**. The irrelevant past information is removed from the memory and the relevant new information is added.

- **Output**: now that we have computed $\mathbf{c}_t$ we can use it together with $[\mathbf{x}_{t-1}, \mathbf{y}_t]$ to estimate the new output of the cell. Once again, a selection of $\mathbf{c}_t$ is computed with a sigmoid layer as in Equation **3.39**, and $\mathbf{x}_t$ is finally estimated from Equation **3.40**.

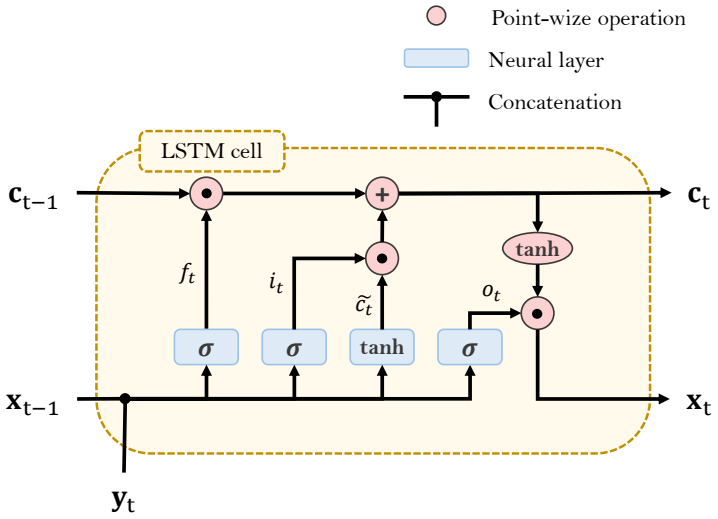


**Fig. 3.3.:** LSTM cell. The red functions correspond to point-wise operations, whereas the blue functions represent the trainable operations.

The LSTM principle is very flexible, and many variants can be implemented. First, a natural thought would be that the choice of the information to erase or to store in the memory should be linked. This is known as the coupled gate variation. In this variant, the input gate $\mathbf{i}_t$ is replaced by $1 - \mathbf{f}_t$ [Greff et al., 2017]. Another idea would be to allow the three gates (forget, input, and output) to access the hidden state. This means replacing the $[\mathbf{x}_{t-1}, \mathbf{y}_t]$ vector by $[\mathbf{x}_{t-1}, \mathbf{y}_t, \mathbf{c}_{t-1}]$ in Equations **3.35** and **3.36** and by $[\mathbf{x}_{t-1}, \mathbf{y}_t, \mathbf{c}_t]$ in Equation **3.39**. This version is called the peephole LSTM [Gers et al., 2000, Yu et al., 2019].

**Convolutional LSTM**. LSTM network is fully connected and has major drawbacks when dealing with image data. Adapting it to images requires flattening 2D or 3D images into high-dimension vectors. This means exploding weight numbers, computational cost, and no spatial correlation between pixels. To use LSTM on high-dimension data, [Shi et al., 2015] introduced the Convolutional Long-Short Term Mermory (ConvLSTM) which was originally designed for rain nowcasting. The

linear operations on weights are replaced by convolution operations which leads to the following Equations:

$$\mathbf{F}_{\mathrm{t}} = \sigma\left(W_{xf} \star \mathbf{X}_{\mathrm{t}-1} + W_{yf} \star \mathbf{Y}_{\mathrm{t}} + W_{cf} \star \mathbf{C}_{\mathrm{t}-1} + b_f\right) \tag{3.41}$$

$$\mathbf{I}_{\mathrm{t}} = \sigma\left(W_{xi} \star \mathbf{X}_{\mathrm{t}-1} + W_{yi} \star \mathbf{Y}_{\mathrm{t}} + W_{ci} \star \mathbf{C}_{\mathrm{t}-1} + b_i\right) \tag{3.42}$$

$$\tilde{\mathbf{C}}_{\mathrm{t}} = \tanh\left(W_{xc} \star \mathbf{X}_{\mathrm{t}-1} + W_{yc} \star \mathbf{Y}_{\mathrm{t}} + b_c\right) \tag{3.43}$$

$$\mathbf{C}_{\mathrm{t}} = \mathbf{F}_{\mathrm{t}} \odot \mathbf{C}_{\mathrm{t}-1} + \mathbf{I}_{\mathrm{t}} \odot \tilde{\mathbf{C}}_{\mathrm{t}} \tag{3.44}$$

$$\mathbf{O}_{\mathrm{t}} = \sigma\left(W_{xo} \star \mathbf{X}_{\mathrm{t}-1} + W_{yo} \star \mathbf{Y}_{\mathrm{t}} + W_{co} \star \mathbf{C}_{\mathrm{t}} + b_o\right) \tag{3.45}$$

$$\mathbf{X}_{\mathrm{t}} = \mathbf{O}_{\mathrm{t}} \odot \tanh\left(\mathbf{C}_{\mathrm{t}}\right) \tag{3.46}$$

### Attention mechanism

The attention mechanism is a wide set of trainable functions that aim to emphasize important information while neglecting irrelevant ones. It is inspired by human brain behavior: to read a text, we must remember the last words and some significant words seen earlier, while others can be forgotten. Another example is human vision: we are usually more sensitive to the center of our vision field or moving objects. Focusing on a few important features and forgetting others is the core of the attention mechanism in deep learning. This principle is widely used in Natural Language Processing (NLP), or in many computer vision tasks. A general formulation of the attention mechanism is to consider two functions $f$ and $g$ to be applied to an input $x$ as follows:

$$Attention = f\left(g(x), x\right) \tag{3.47}$$

In this formulation, $g$ is the function that generates the attention based on the input information, and $f$ is the function that selects information in $x$ based on the attention $g(x)$ [Guo et al., 2021]. Various approaches can be considered, depending on how information is discriminated by the attention mechanism. If the model focuses on specific time steps, spatial locations, or channels above others, we call it *temporal*, *spatial*, or *channel* attention, respectively. We present hereafter some widely used attention modules that we will use or refer to in this thesis.

The **Convolutional Block Attention Module (CBAM)** is an attention-based convolution proposed by [Woo et al., 2018] for 2D images. Let $x \in \mathbb{R}^{C \times H \times W}$ be an image

with $C$ channels. CBAM is composed of two attention mechanisms performed sequentially: channel attention (Equation **3.48**) and spatial attention (Equation **3.49**).

$$x_c = g_c(x) \odot x \tag{3.48}$$

$$x_s = g_s(x_c) \odot x_c \tag{3.49}$$

The channel attention $g_c$ (see Equation **3.50**) first computes both the average and the maximum on the entire spatial dimension for each channel, producing two vectors of size $C$. A shared MLP is calculated on the two channel descriptors, and a sigmoid activation function $\sigma$ is applied sum of the two vectors. The selection function $f$ from Equation **3.47** is simply the element-wise product $\odot$ between the input and the computed attention.

$$g_c(x) = \sigma\left(MLP(\underset{H,W}{\text{AvgPool}}(x)) + MLP(\underset{H,W}{\text{MaxPool}}(x))\right) \tag{3.50}$$

Following channel attention, spatial attention $g_s$ (see Equation **3.51**) starts by averaging and max-pooling operations across the channels of $x_s$. Then a convolution operation is performed on the concatenation of the two feature maps, aggregating every channel to one image, followed by a sigmoid function.

$$g_s(x_c) = \sigma\left(\mathbf{W}^{2\times7\times7}(\underset{C}{\text{AvgPool}}(x_c); \underset{C}{\text{MaxPool}}(x_c))\right) \tag{3.51}$$

To retrieve the final output, another element-wise product between the spatial attention and $x_c$ is performed.

**Self attention**. Let us consider a sequence of $N$ inputs each of dimension $D$ and the associated input matrix $x \in \mathbb{R}^{N\times D}$. The *self-attention* mechanism consists in comparing a matrix $\mathbf{Q}$ of "queries", to a matrix of "keys" $\mathbf{K}$, and to return the "values" from a matrix $\mathbf{V}$ which have a high correspondence to the keys. In self-attention the matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V} \in \mathbb{R}^{N\times D}$ are generated from the input $x$ using a learnable linear combination, i.e. $\mathbf{Q} = x\mathbf{W}_{\text{q}}$, $\mathbf{K} = x\mathbf{W}_{\text{k}}$ and $\mathbf{V} = x\mathbf{W}_{\text{v}}$ where $\mathbf{W}_{\text{q}}, \mathbf{W}_{\text{k}}, \mathbf{W}_{\text{v}} \in \mathbb{R}^{D\times D}$ are the trainable parameters. The model is thus able to take inputs of variable length, as the sequence length $N$ does not appear in the weight matrix size. The attention $g(x)$ is then computed as $g(x) = \text{SoftMax}(\mathbf{Q}\mathbf{K}^\top)$ and the selection $f(x, g(x)) = g(x)\mathbf{V}$ [Guo et al., 2021]. A variant of this mechanism is the Multi-Head Self Attention (MHSA), where many matrices $\mathbf{W}_{\text{q}}, \mathbf{W}_{\text{k}}, \mathbf{W}_{\text{v}}$ are used in parallel to learn more representations of the same sequence and using GPU high parallel computing capabilities. When the keys and values come from a different

input than the queries, this mechanism is called Multi-Head Cross Attention (MHCA).

**Transformers and Vision Transformers**. Using MHSA in a sequence-to-sequence encoder-decoder, [Vaswani et al., 2017] introduced the very famous transformer architecture, represented in Figure **3.4**. Its principle relies on two blocks: first, the encoder block uses MHSA on the input sequence $\mathbf{y}$, and a feed-forward network with intermediate residual connections and Batch Normalization (BN) layers. The decoder starts with the outputs generated by the transformer at the previous timestep ($\mathbf{x} \in \mathbb{R}^{Nx-1 \times D}$) and performs MHSA on this tensor. Then, a MHCA block is computed using the keys and values from the decoder and queries from the encoder. A final feed-forward network is applied to generate the next output. Before being passed to the neural network, a positional encoding is applied to $\mathbf{y}$ and $\mathbf{x}$, allowing learning position-specific attention between embedded features.

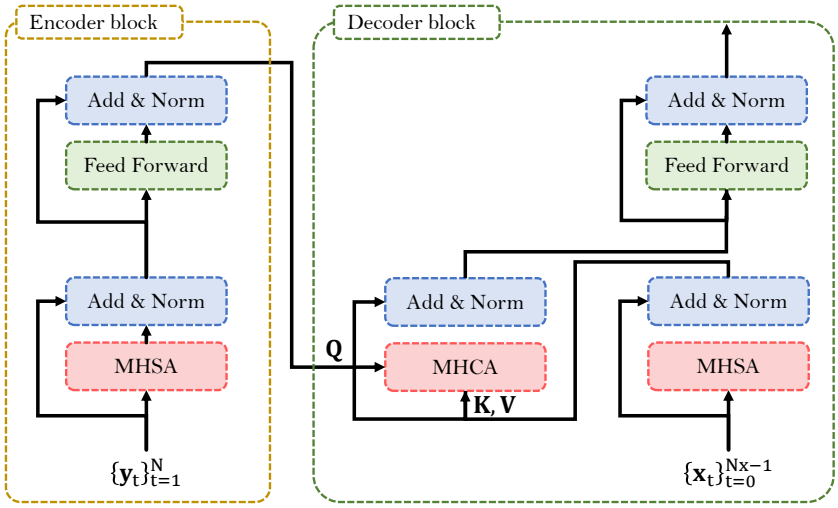


**Fig. 3.4.:** Transformer architecture (reproduced from [Vaswani et al., 2017]).

Transformers have been widely used in many sequence tasks, especially in Natural Language Processing (NLP) recently achieving very impressive results [Patwardhan et al., 2023]. An extension of this idea to computer vision was proposed by [Dosovitskiy et al., 2021], introducing the Vision Transformer (ViT) in which the input sequence is replaced by an image. To be comparable with the original transformer architecture, the image is spatially separated into $N$ patches, and each patch is embedded and mapped to a vector. The obtained vectors are then concatenated, and using positional encoding, they form a sequence comparable to a NLP sentence. ViT was originally introduced for the image classification task, but since then, many other applications were tackled, from image generation to super resolution [Khan et al., 2022], for spatiotemporal image inpainting [Feichtenhofer et al., 2022], on remote

sensing data [Cong et al., 2022], or on physical oceanography applications [Wang et al., 2024].

# 4 Sea Surface Height Downscaling on numerical simulations

In this chapter, we address the problem of SSH downscaling (or Super-Resolution), in which we aim to estimate high-resolution SSH from lower-resolution inputs. In an operational framework, it would correspond to enhancing the resolution of an interpolation product, such as the DUACS L4 map. Our goal is to develop a neural network that leverages very high-resolution SST fields to improve SSH resolution. This work serves as a proof of concept, demonstrating the feasibility of enhancing SSH resolution using SST. However, we did not examine the direct applicability of this downscaling method to real observations. In this chapter, we exclusively use simulated data to showcase the potential of our method. This approach provides both benefits and limitations: we have access to a high-resolution reference for evaluation, but there are no assurances that the methods will perform similarly on real data. The chapter is structured as follows: Section **4.1** introduces the NATL60 simulation data, and Section **4.2** details our downscaling method and architectures. Section **4.3** presents our findings on SSH reconstruction and derived surface currents. Finally, Section **4.4** discusses the limitations of this work.

## 4.1 The NATL60 simulation

Supervised neural network training requires a dataset of inputs/outputs, but we lack fully gridded high-resolution satellite data to perform such training. Therefore, we propose to perform a twin experiment on a high-resolution simulation: the NATL60 simulation. The NATL60 simulation is a very high-resolution model of the North Atlantic Ocean extending from 26°N to 66°N with a resolution of 1/60°. It is based on the Nucleus for European Modelling of the Ocean (NEMO) version 3.6, a global circulation ocean model [Madec et al., 2017], with atmospheric forcing [Dussin and Barnier, 2014], and initial conditions taken from the Global Physical Reanalysis (GLORYS) [Lellouche et al., 2018, CMEMS, 2020] and no tide model [Ajayi et al., 2020]. NEMO 3.6 is integrated from these initial conditions with an atmospheric

forcing, but without assimilation. The output data is, therefore, physically realistic regarding NEMO 3.6 equations but does not represent a day that occurred. Its very fine grid allows NATL60 to model physics at an effective resolution of about 10–15 km. We focus on the Gulf Stream, an open ocean area from 26.8°N to 44.3°N and from -64.4°W to -42.3°W, which corresponds to a square image in terms of pixel number, and far enough from the Equator so that the geostrophic approximation is valid. As NATL60 was extremely computationally intensive to run (16McpuH, on 14,000 cores), it is not possible to generate as much data as needed. In our case we have access to a complete year (from October 1, 2012, to October 1, 2013) and to 4 separate months (March, June, September, and December 2008). We retrieve from the simulation the SSH, the SST, and the Northward and Eastward currents, U and V, respectively. A visual overview of the data is presented in Figure **4.1** where *Rx* denotes the resolution $x$ minutes of arc (1/60°). Therefore, the native resolution of NATL60 is *R01*.

As seen in Figure **4.1** the studied area can be divided into two parts: the very energetic north, and the south which is less energetic. The north contains a frontier between cold and warm waters with a very dominant eastward current. On the other hand, the south is much more homogenous and has lower kinetic energy. By separating the area at latitude 35°N, we compute the histogram of the distributions of the NATL60 simulation in Figure **4.2**, and the mean and standard deviation values in Table **4.1**. The studied values are SSH, SST, and Kinetic Energy (KE) defined as follows:

$$KE = \frac{1}{2}\rho_0(u^2 + v^2) \tag{4.1}$$

where $\rho_0$ is the water volumic mass.

By differentiating these two regions, we see the "border" effect in the north, where the data distribution is more evenly spread, regarding SSH, SST, and KE. The southern area has a higher average SSH between $0$ and $1$ m, whereas the northern SSH ranges from $-0.5$ to $1$ m. This difference in repartition is also visible in the temperature. Globally the most energetic area is the north, by far, with 85% of the total KE.

To artificially generate low-resolution data, we compute an "average pool" on the reference data. It consists of taking the average of the pixel values inside a square with a predefined size and using this mean as the next pixel value. In our case, we use a square of size $3 \times 3$, which we apply recursively to obtain the data at resolutions R03, R09, R27, and R81, which correspond to approximately $4.5, 14, 41,$ and $122$ km per pixel, respectively. The input SSH image at $R81$ has a very low resolution even compared to DUACS, (1.35° per pixel against 0.25° per pixel for DUACS). However, as

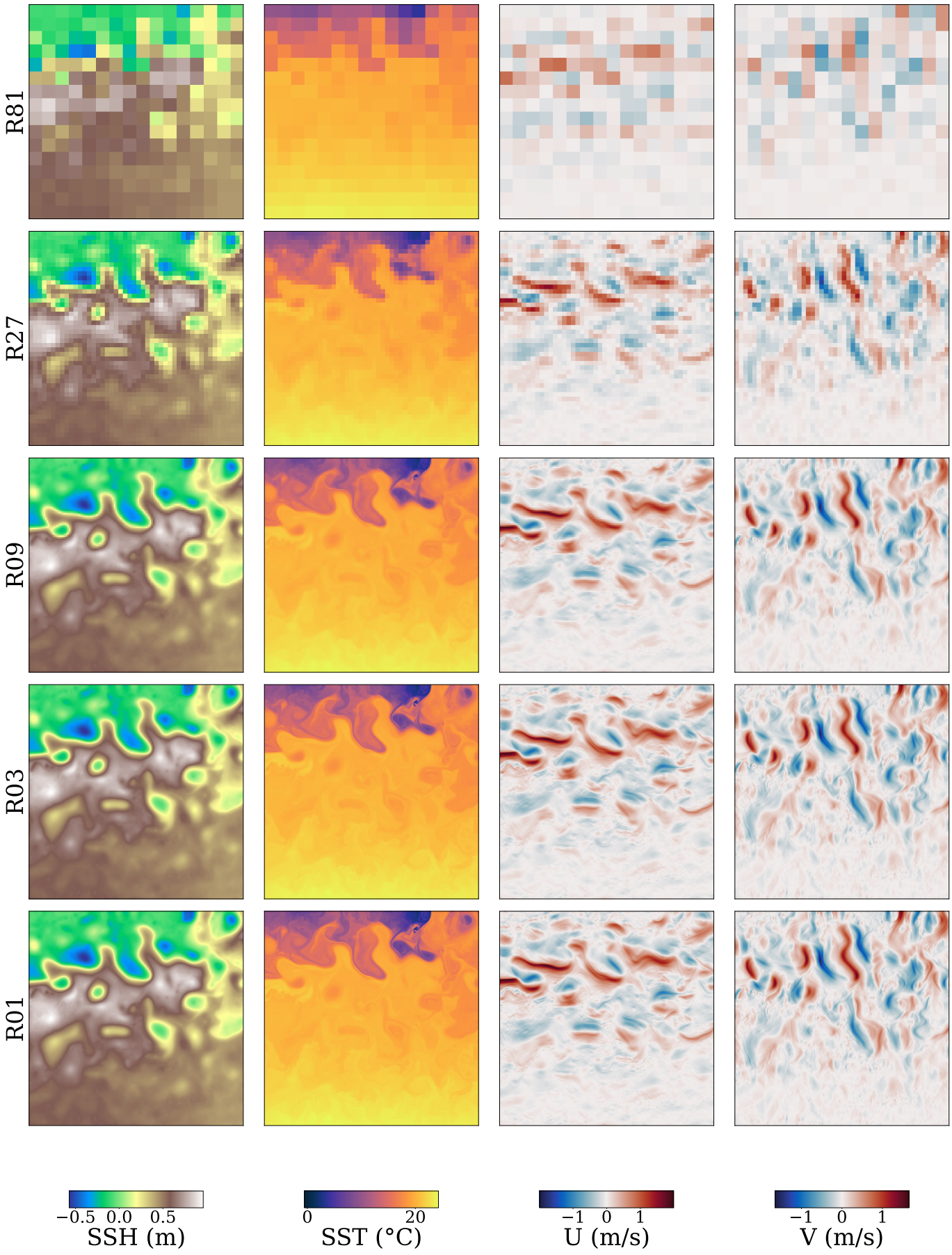


**Fig. 4.1.:** Example of one day (January 8, 2012) of the NATL60 simulation at different resolutions.

the effective resolution of DUACS is lower than its pixel resolution (see Section **5.6.1**), starting from a very coarse SSH image ensures that all the high-frequency structures are lacking from the input and must be estimated.

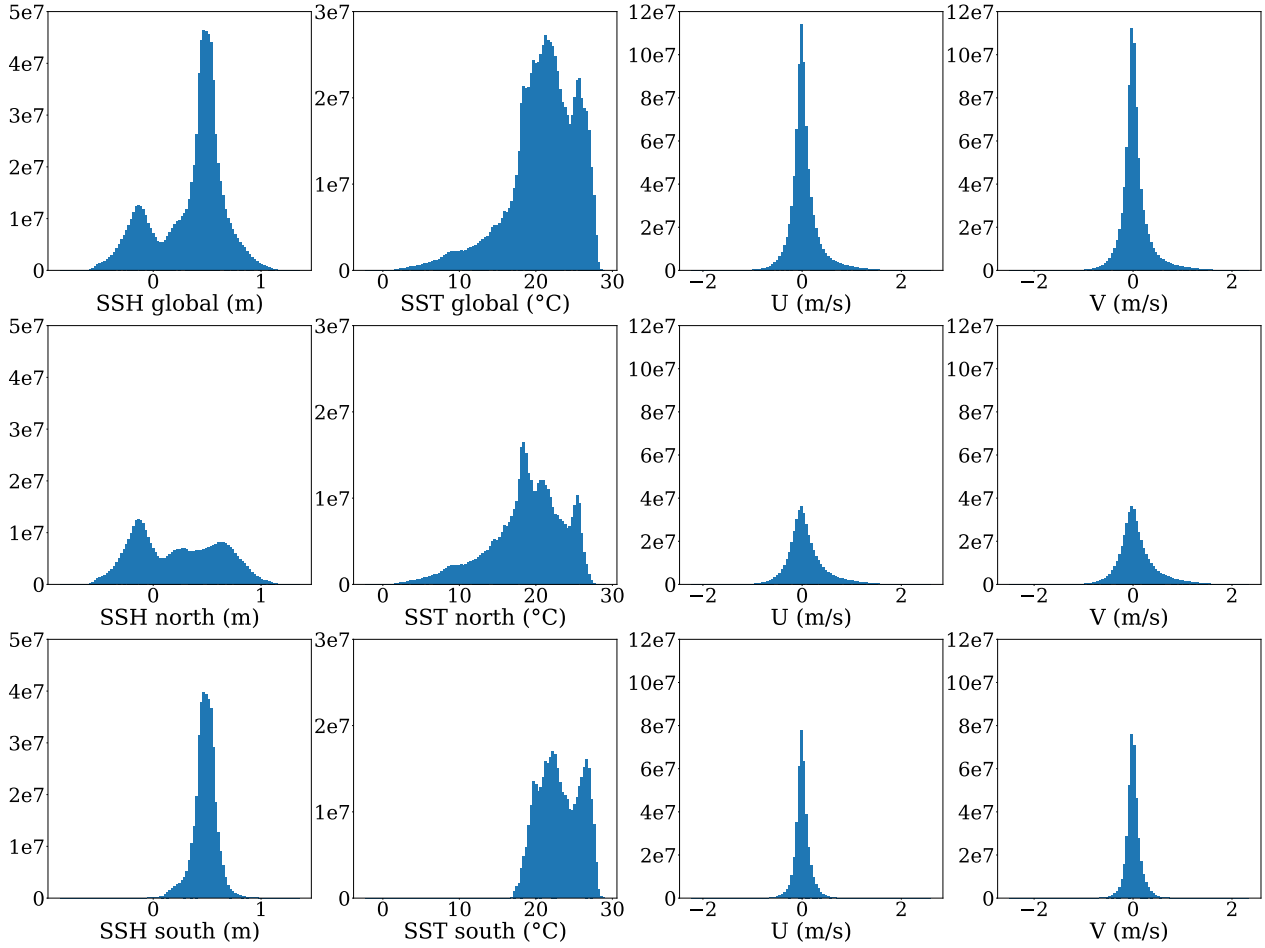


**Fig. 4.2.:** Histograms of SSH, SST, U, and V in the 3 defined areas at resolution R01.

| Area | SSH (m) | | SST (°C)) | | U (m/s) | | V (m/s) | | KE (J/m³) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | mean | std | mean | std | mean | std | mean | std | mean | std |
| Global | 0.35 | 0.32 | 20.9 | 4.5 | 0.04 | 0.3 | 0.0 | 0.27 | 81 | 188 |
| North | 0.23 | 0.39 | 18.9 | 4.9 | 0.07 | 0.39 | 0.0 | 0.34 | 136 | 245 |
| South | 0.48 | 0.11 | 23.1 | 2.7 | 0.0 | 0.15 | 0.0 | 0.14 | 20 | 36 |

**Tab. 4.1.:** Mean and standard deviation statistics on the 3 defined areas at resolution R01.

## 4.2 Proposed methods

### 4.2.1 Stage by Stage downscaling: RESAC

We propose to train a downscaling neural network from $\mathrm{SSH}_{R81}$ to $\mathrm{SSH}_{R03}$ (with a total upsampling ratio of $\times 27$), using higher resolution SST. In the early development of our method, we observed that performing the downscaling stage by stage improved the network training, by stabilizing it. Given the high upsampling ratio, we divided the SR task into lower upsampling tasks, each one resolved by a different neural network. It is also possible in the final stage to use another network to estimate the currents in addition of SSH. We call this approach REsolution by Stages of Altimetry and Currents (RESAC). The training is end-to-end, meaning that the upsampling networks are trained together using a common loss function which we detail later. In our first work, we performed two downscaling stages, each one of upsampling ratio $\times 3$, and the estimation of the currents [Thiria et al., 2023]. In our latter, we pushed upsampling a step further with an improved architecture but without current estimation [Archambault et al., 2022]. In the following, we present close but new results combining the features of the two works: 3 upsampling steps and current estimation, including extensive architecture comparison, SST and multi-scale loss ablation studies, and noise impact.

Let's now detail RESAC training method, represented in Figure **4.3**. The SR starts from $\mathrm{SSH}_{\mathrm{R81}}$ and progressively downscales to $\mathrm{SSH}_{\mathrm{R27}}$, $\mathrm{SSH}_{\mathrm{R09}}$, $\mathrm{SSH}_{\mathrm{R03}}$. Each downscaling block $f_\theta$ takes as input a low-resolution SSH, (from the data or the previous downscaling block) and eventually a higher-resolution SST. It outputs the improved resolution SSH which is used to compute the loss at the current resolution, and then fed to the next neural network. After the full SSH downscaling, another neural network $g_\theta$ can be used to derive $U$ and $V$.

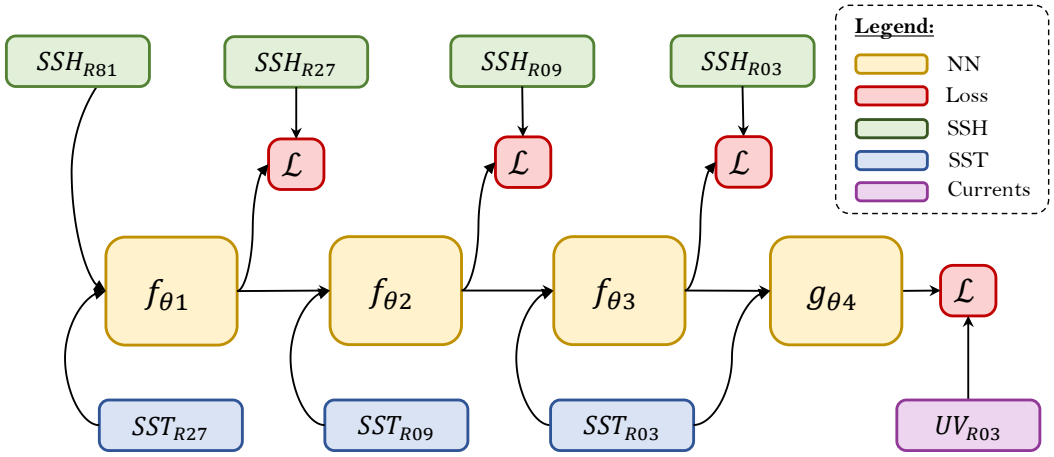


**Fig. 4.3.:** REsolution by Stages of Altimetry and Currents (RESAC) training computational graph.

The loss function to be minimized is the multi-scale Mean Squared Error (MSE):

$$\begin{aligned}\mathcal{L} =& \text{MSE}\left(\widehat{\text{SSH}}_{R27}, \text{SSH}_{R27}\right) + \text{MSE}\left(\widehat{\text{SSH}}_{R09}, \text{SSH}_{R09}\right) + \text{MSE}\left(\widehat{\text{SSH}}_{R03}, \text{SSH}_{R03}\right) \\ &+\text{MSE}\left(\widehat{\text{UV}}_{R03}, \text{UV}_{R03}\right), \\ &\text{where} \\ &\widehat{\text{SSH}}_{R27} = f_{\theta 1}(\text{SSH}_{R81}, \text{SST}_{R27}) \\ &\widehat{\text{SSH}}_{R09} = f_{\theta 2}(\widehat{\text{SSH}}_{R27}, \text{SST}_{R27}) \\ &\widehat{\text{SSH}}_{R03} = f_{\theta 3}(\widehat{\text{SSH}}_{R09}, \text{SST}_{R03}) \\ &\widehat{\text{UV}}_{R03} = g_{\theta 4}(\widehat{\text{SSH}}_{R03}, \text{SST}_{R03})\end{aligned} \tag{4.2}$$

where the MSE loss function is defined in Appendix **A**.

The different terms of the loss have comparable magnitudes, as all data are normalized (see Section **4.2.3**). In a general setting, the different parts of this loss could be weighted to give more importance to some tasks than others. In our work, we decided not to do so, as tuning the ponderation between losses requires training a significant quantity of neural networks in different settings, which we did not consider a priority. However, in Section **4.3.1**, we compare three loss function variants. In the first one, denoted *multiscale MSE +UV*, we use all the terms presented in Equation **4.2**. The second one is an ablation study of the UV block, as we compute the SSH MSE at each resolution but not the UV MSE. We call this variant *Muliscale MSE*. The last one, simply entitled *MSE*, is an ablation study of the currents and the multiscale control where we compute only the MSE at the target resolution $R03$.

### 4.2.2 Architectures

We now detail the neural architectures $f$ and $g$. In [Thiria et al., 2023], we used a fully convolutional network, applying first bicubic upsampling followed by convolutions denoted RESAC. In [Archambault et al., 2022] we improved the architecture using a subpixel convolution residual network. Subpixel convolution (See Section **3.3.2**) is an efficient convolutional method, as for the same computational cost, it has more parameters than regular convolutions. It also makes the upsampling strategy learnable, whereas the method used in vanilla RESAC was performing convolutions in pre-upsampled images. Also, the residual addition improves the training by introducing shortcuts in the gradient flow, which helps it to fit the first weights of deep networks (See Section **3.3.2**). Residual learning is well adapted to downscaling, as it forces the network to learn progressive small modifications of its input, which is particularly suited to our direct operator. We adjusted the standard residual block to our needs, where we have an SSH image of low resolution to improve and an SST image with higher resolution. In the following, we describe four neural networks: first, the RESAC and RESACSUB downscaling block, playing the role of $f$ in Figure **4.3**. Then the RESAC and RESACSUB current estimation blocks, playing the role of $g$ in Figure **4.3**. A visual overview of the architectures is presented in Figure **4.4**.

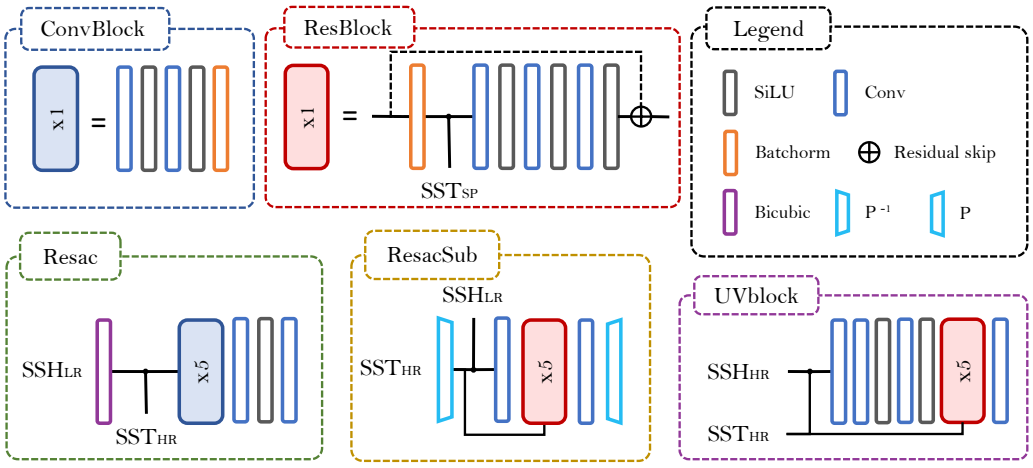


**Fig. 4.4.:** REsolution by Stages of Altimetry and Currents (RESAC) architecture. Sigmoid Linear Unit (SiLU) is defined as $\mathrm{SiLU}(x) = x\sigma(x)$, with $\sigma(x)$ the sigmoid function. The pixel-shuffling operators $P$ and $P^{-1}$ are defined in Figure **3.1**.

The **RESAC downscaling block** starts by upsampling the SSH image using the bicubic interpolation and concatenating it with the SST higher resolution image on the channel dimension. Then a block composed of two convolutions + SiLU activations followed by a batch normalization is applied $5$ times. Finally, the resulting

tensor is passed through an extra convolution + SiLU, and a final linear combination between the obtained channels produces the output. All the filter kernels have a $3 \times 3$ shape and $1$ padding pixel to conserve the image dimension, except for the last linear layer with a kernel of size $1$ and no padding.

The **RESACsub downscaling block**, on the other hand, does not perform any SSH upsampling beforehand but applies the pixel shuffling operator $P$ on the SST image. As a consequence, the low-resolution SSH can be concatenated with the high-resolution SSH as they have the same spatial dimensions, but with one channel for the SSH and $9$ for the SST. A first convolution is performed to increase the number of channels and then $5$ residual blocks are applied. These blocks are specially designed to benefit from the good backpropagation of the gradient of the residual learning and from the SST information. They start with a batch normalization, followed by a concatenation with the sub-pixel SST. Then, three convolutions + SiLU are applied, where the last one outputs a tensor with the same number of channels as the input (before concatenation with SSH), and performs the residual skip connection. In the end, a last convolution brings the total number of channels to $9$, without activation, and the pixel shuffling operator $P$ is applied to retrieve the super-resolved image. As RESAC, all convolutions use $3 \times 3$ kernels and padding of $1$ except the last one, which has a kernel size of $1 \times 1$ and a padding of 0.

The **RESACsub UVblock** is an optional block that we use for the estimations of currents. We first describe the architecture used in RESACsub. The block uses residual learning but no subpixel convolutions (as no super-resolution task is performed in this stage). In [Thiria et al., 2023] we initially designed a fully convolutional network implemented as the RESAC block without bicubic upampling. Here, we present a modified version more suited to the estimation of currents from the SSH estimated from RESACsub. As shown in Figure **4.7** and discussed further, subpixel convolution introduces checkerboard artifacts, which are a serious drawback for current estimation. To overcome this issue, we start the UVblock by convolutions + SiLU with wide kernels of $7 \times 7$ and padding of $3$ to help the artifact removal. Then, the residual block previously described is applied $5$ times, and a last layer with no activation and a kernel size of $1\times$ outputs the two current components.

The **RESAC UVblock** is identical to the one described for RESACsub but without the residual skip connection.

Of course, all the described architectures can also take SSH alone and not output U and V currents. In the following, we denote RESAC-ssh-sst and RESAC-ssh, the fully convolutional architectures using SST or not, RESACsub-ssh-sst and RESACsub-ssh, their subpixel equivalents.

### 4.2.3 Training details

#### Dataset split

We split the dataset into $3$ subsets: train, validation, and test. We take 366 daily examples from October 1, 2012, to October 1, 2013, for training. In doing so, the network sees an entire year during training and does not overfit some seasons. The 4 extra months (in 2008) are split in two: March and June for validation and September and December for testing. As the test, validation, and training set are separated by months or even several years, there is no data leakage from training to evaluation data.

#### Normalization

All the data are normalized similarly: we center and reduce the data by calculating their mean and standard deviation on the training set. The same statistics are then applied to validation and test sets so that no information on these datasets is used in the normalization of the training.

#### Training hyperparameters

We use an ADAptative Moment estimation optimizer (ADAM) [Kingma and Ba, 2014] with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The starting learning rate is $1.10^{-4}$ for RESAC and $2.10^{-3}$ for RESACSUB, and is multiplied at each epoch by $\gamma = 0.95$. The starting learning rates and the $\gamma$ were hand-tuned separately on the RESAC-ssh-sst and RESACSUB-ssh-sst and applied to their SSH-only versions. The fact that we can use a higher learning rate in RESACSUB can be explained by the stabilization brought by residual skip connections. An early stopping is used with a patience of 10 epochs and the stopping criterion is the RMSE of the $\mathrm{SSH}_{R03}$ on the validation set.

#### Ensemble

As mentioned in Section **3.3.1** the neural network training is sensitive to weights initialization. In the following, we train $5$ networks for each experiment with different initializations to construct an ensemble. We refer to *ensemble estimation* as the mean of the estimations of the $5$ networks. It is then possible to compute

the *ensemble score*, i.e., the score of the ensemble estimation, and the *mean score*, i.e., the average of each member score in the ensemble. In the following, if $m$ is a metric of reconstruction, it always refers to the *mean score*, and $\overline{m}$, on the other hand, refers to the *ensemble score*.

## 4.3 Results

### 4.3.1 SSH reconstruction

#### Comparison of architectures

In the following, we compare the downscaling of the two architectures, RESAC and RESACSUB, to show their advantages and drawbacks. To do so, we train RESAC and RESACSUB networks with SSH or SSH + SST inputs, and we use the bicubic upsampling as a baseline. We aggregate the results in Table **4.2**, in which we provide the mean SSH RMSE $\mu$ and its ensemble score $\overline{\mu}$. We test $3$ losses: the MSE on $\mathrm{SSH}_{R03}$, the multiscale MSE on $\mathrm{SSH}_{R27}$, $\mathrm{SSH}_{R09}$, $\mathrm{SSH}_{R03}$, and the multiscale MSE, with an additional control on currents, as described in Equation **4.2**. In the first two settings, there is no current estimation block.

| | MSE | | | | Multiscale MSE | | | | Multiscale MSE+UV | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SSH | | SSH + SST | | SSH | | SSH + SST | | SSH | | SSH + SST | |
| | $\mu$ | $\overline{\mu}$ | $\mu$ | $\overline{\mu}$ | $\mu$ | $\overline{\mu}$ | $\mu$ | $\overline{\mu}$ | $\mu$ | $\overline{\mu}$ | $\mu$ | $\overline{\mu}$ |
| RESAC | 6.69 | 5.85 | 5.03 | 4.4 | 5.87 | 5.51 | 4.34 | 3.95 | 6.93 | 5.96 | 5.08 | 4.47 |
| $\mathrm{RESAC}_{\mathrm{SUB}}$ | 5.65 | 5.38 | 3.94 | 3.57 | 5.62 | 5.37 | 3.98 | 3.61 | 6.27 | 5.77 | 3.9 | **3.36** |
| BICUBIC | 6.79 | | | | | | | | | | | |

**Tab. 4.2.:** Reconstruction RMSE ($\mu$ in cm) on $\mathrm{SSH}_{R03}$ for different downscaling methods. The scores are computed on the set (September and December 2008).

As expected, the SST has a significant impact on the reconstruction of every method. When trained using multiscale MSE, SST leads to an improvement of $1.53$ cm$(\approx 26\%)$ and $1.64$ cm$(\approx 29\%)$, in mean score for RESAC and RESACSUB respectively. Comparing architectures, it seems that the subpixel architecture has a clear advantage over standard convolution, especially in the SSH+SST setting. In the SSH-only scenario, the two methods significantly outperform bicubic upsampling by around 1 cm. Another advantage of using subpixel convolutions is that for the same weight number, it is much more computationally efficient than standard convolutions. For

instance, on our training set of 366 examples one epoch takes $\approx 2.5$ s and $\approx 15.4$ s for RESACssh-sst and RESACSUBssh-sst respectively.[1]

The comparison of losses highlights another advantage of the residual skip connections. Specifically, the multiscale loss significantly enhances the RESAC reconstruction across all configurations, whereas it results in nearly equivalent performances for RESAC SUB. We interpret this finding as an indication that fully convolutional networks are prone to the vanishing gradient problem, which complicates the training of very deep networks. This issue underscores the benefit of constraining the network at multiple resolutions. On the other hand, the residual network does not benefit from the multiscale loss as it is less impacted by gradient vanishing (see Section **3.3.2**). Using skip connections makes the multiscale control unnecessary.

Additionally, regulating the networks based on currents (last row) leads to a notable performance decline for RESAC and RESACSUB-ssh, but results in a slight improvement for RESACSUB-ssh-sst. This performance drop can be attributed to the increased complexity and noise in the current estimation process, which introduces additional errors and makes the training process more challenging, at least for SSH reconstruction. However, in Section **4.3.2**, we will see that it leads to a significant current improvement compared to geostrophy.

#### Impact of the ensemble estimation

Also, in every setting, the ensemble reconstruction has a lower SSH RMSE ($\mu > \overline{\mu}$), which is expected in ML (see Section **3.3.1**). However, finding the right ensemble size is always a trade-off between computational cost and score improvement. In Figure **4.5**, we plot the ensemble reconstruction score as a function of the ensemble size. We see that even with only two members, the ensemble significantly reduces the RMSE compared to the mean RMSE and that with $5$ members, we reach $\approx 10\%$ of RMSE gain (to be compared with the $\approx 12.5\%$ with $50$ members). This empirically justifies our choice of using $5$ member ensembles in the following experiments.

#### Visual comparison

We present a visual comparison of the ensemble reconstructions in Figure **4.6** and of their difference to ground truth in Figure **4.7**. The SSH fields estimated using only SSH inputs appear much smoother than those using SST. This smoothing mainly affects the small structures of the SSH, as in the absence of SST high-resolution

[1] Performances calculated on 100 epochs, with batch size of $8$ on NVIDIA RTX A4000.

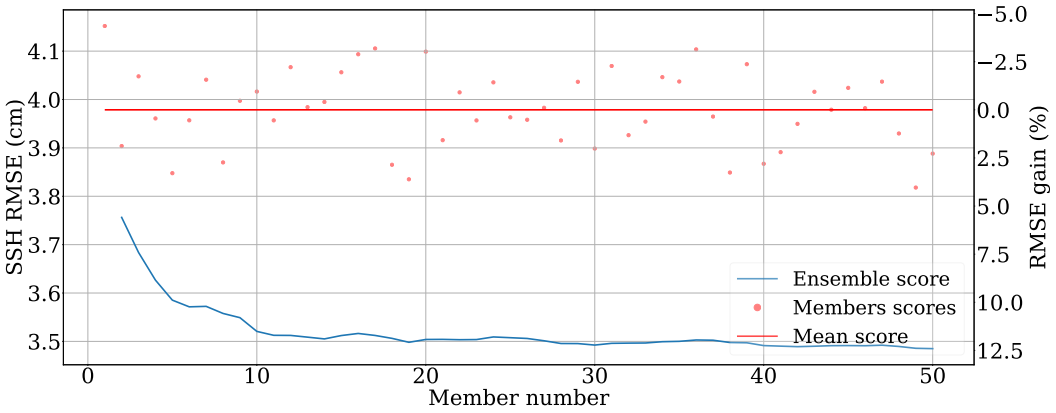


**Fig. 4.5.:** Impact of the ensemble size on the SSH RMSE ($\overline{\mu}$)

information, they cannot be guessed by the neural network. However, SSH is quite a smooth field, and visually assessing the quality of the reconstructions on SSH directly might be difficult. It is easier to compare difference fields as in Figure **4.7** or relative vorticities maps (see next section).

In SSH differences maps to ground truth, we see once again the smoothing of small structures. However, we also see that even if RESACSUB has overall lower error maps, it also produces checkerboard artifacts, that the fully convolutional RESAC does not. This is not very important for SSH reconstruction itself, but it will lead to a very bad geostrophic approximation, as it requires estimating the spatial gradient of the image. In the next section, we present a solution to still correctly estimate currents from SSH presenting such issues. These artifacts can be explained by the subpixel convolution operation: before upsampling spatial neighbors are estimated by different kernels and stored in an image with more channels but smaller spatial dimensions. When the pixel shuffling operator is applied, channel-neighbor pixels become spatial-neighbor pixels in the high-resolution image. The kernels producing the sub-pixel channels are initialized differently and converge to different values during training. Their output could be close, but slightly different, which causes both the trainable upsampling and the checkerboard artifacts.

### Impact of the SSH noise

In the previously described experiments, we trained neural networks to inverse a noise-free decimation operator. The resolution is then reduced without changing the mean of the pixels compared to ground truth. However, interpolation methods such as DUACS do not produce a perfect SSH image, even at very low resolutions. To get

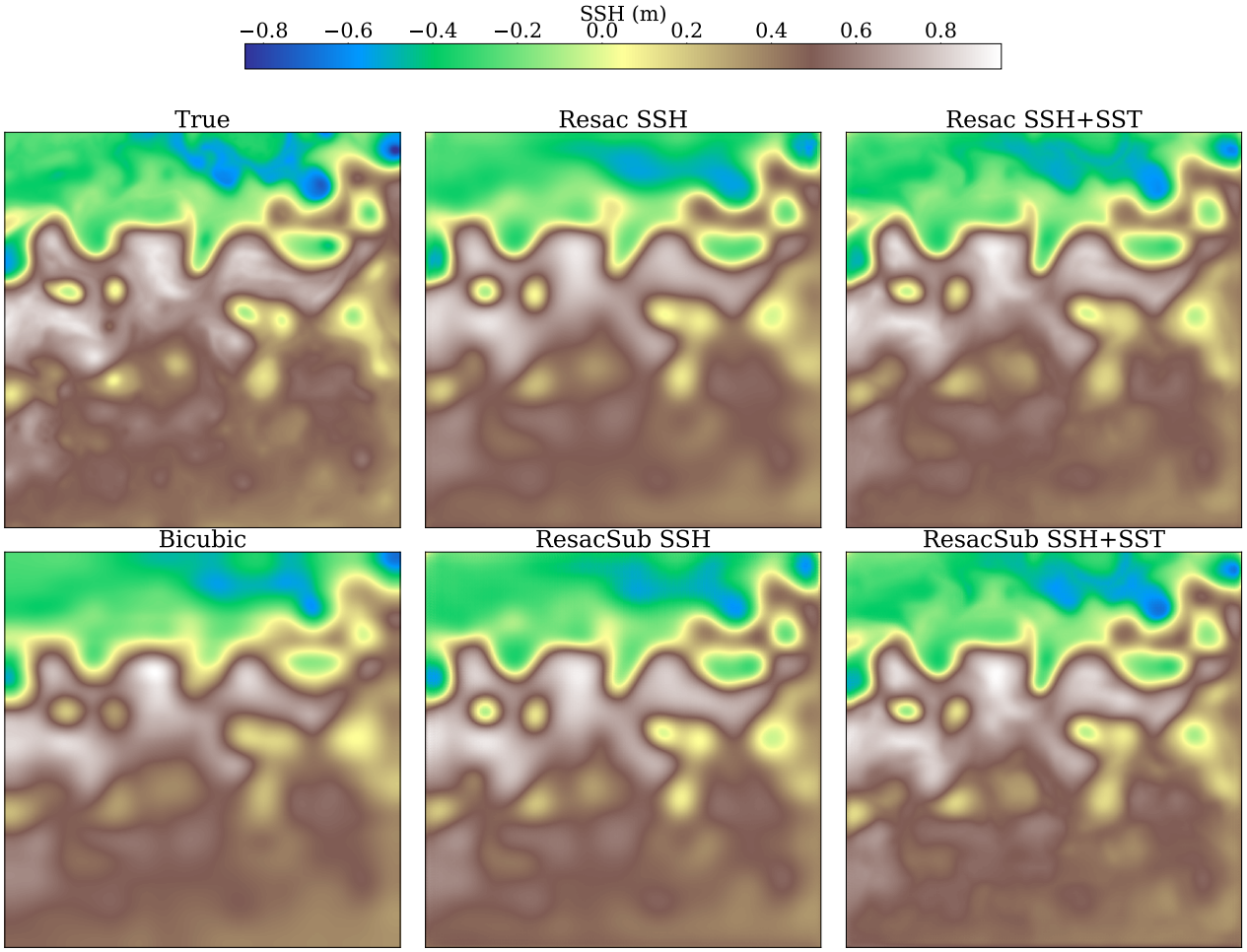


**Fig. 4.6.:** Reconstructed SSH fields

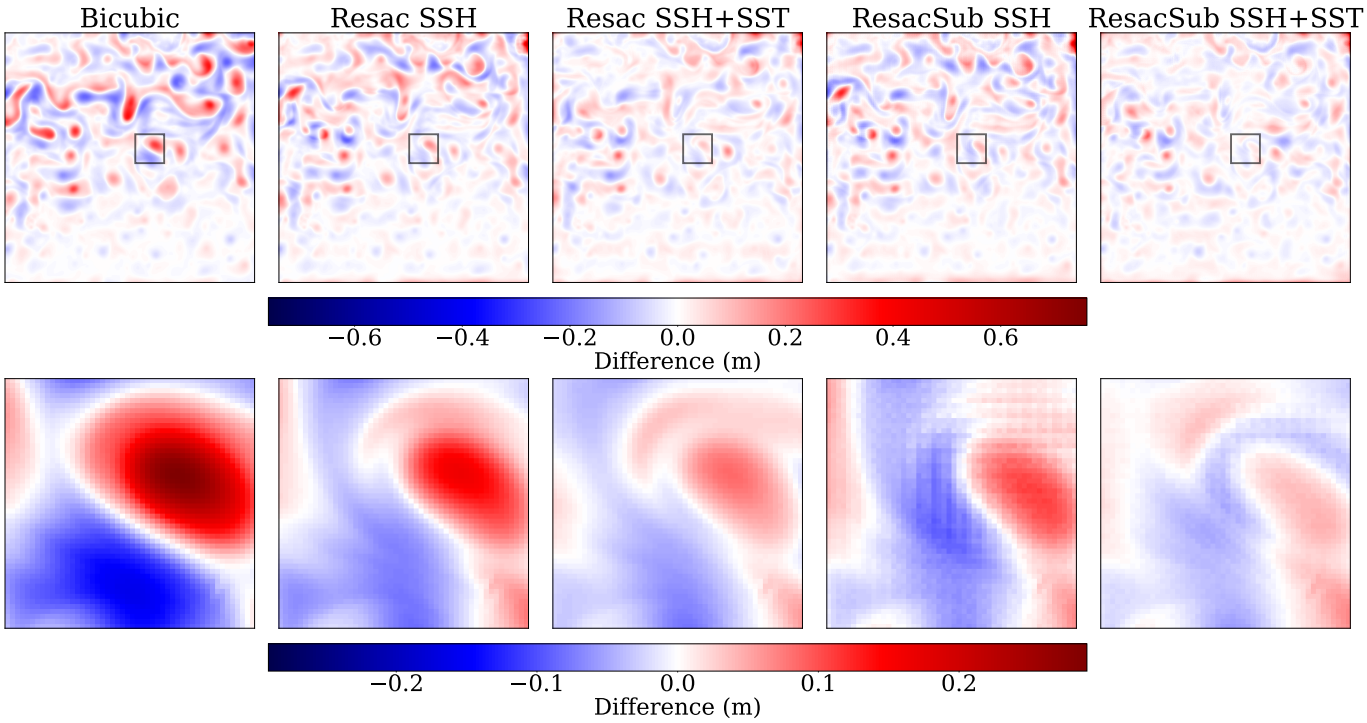


**Fig. 4.7.:** Error map between the reconstructed SSH and the ground truth

a step closer to a realistic low-resolution image, we add a random Gaussian noise of standard deviation $\sigma$ in addition to the decimation:

$$\mathbf{I}^{\mathrm{LR}} = \mathcal{D}_{\mathrm{ec}}(\mathbf{I}^{\mathrm{HR}}) + \varepsilon \quad \text{where} \quad \varepsilon \sim \mathcal{N}(0, \sigma \boldsymbol{I}) \tag{4.3}$$

In Figure **4.8**, we study the impact of the standard deviation of $\varepsilon$ on the RMSE of RESAC$_{\mathrm{SUB}}$. To do so, we train an ensemble of 5 members for different noise values (fixed at the beginning of the training).

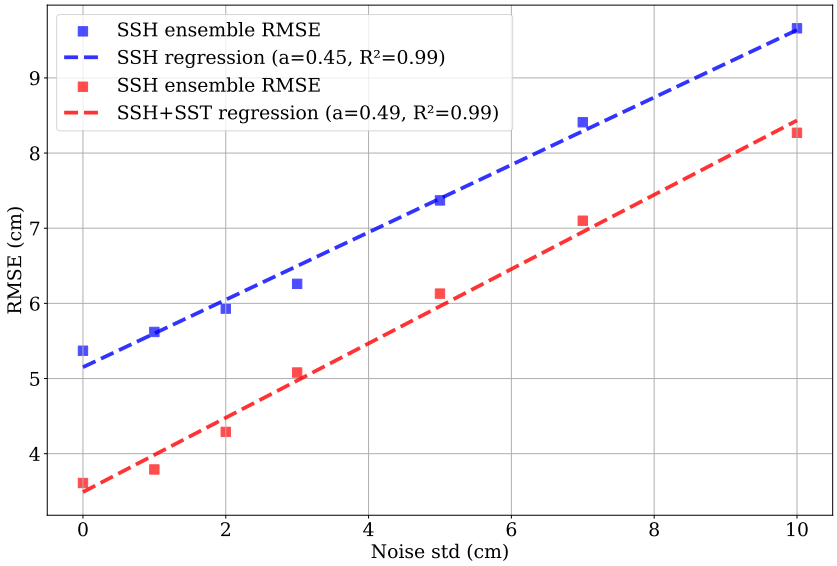


**Fig. 4.8.:** RMSE as a function of the noise standard deviation for RESAC$_{\mathrm{SUB}}$.

Even though the SST-using networks produce lower errors, they are still very impacted by the SSH noise. We find that the RMSE increases linearly with the noise standard deviation (high $R^2$ of 0.99). The RMSE growth rates are $\approx 0.45, 0.49$ for SSH-only downscaling and SSH + SST, respectively.

### 4.3.2 Joint Super-Resolution of SSH and estimation of currents

As the most important use of the SSH field is to estimate surface currents, we are interested in assessing the currents that can be derived from our super-resolved maps. The velocity field is usually derived from SSH through the geostrophic approximation (Equation **2.3**), but it can also be directly estimated using the neural network as described in Section **4.2.2**. We use two variants of the current block in the following: a fully convolutional block for RESAC or a residual block for RESACSUB. This last block is specifically designed to get rid of the checkerboard

artifacts visible in Figure **4.7** produced by the subpixel convolutions. It starts with wide-kernel convolutions on the SSH estimation, which can detect and remove the artifacts. We compare the currents directly estimated by the neural networks and the geostrophic currents derived from their SSH estimations to the NATL60 surface currents. We use the geostrophic currents of the bicubic interpolation as a baseline and the geostrophic currents computed on the ground truth SSH to give an upper bound performance of the geostrophic approximation. We aggregate the scores in Table **4.3**. The considered metrics are $\mu_u$, $\mu_v$, $\mu_n$ (in cm/s) the RMSE of the $u, v$ components independently, and on the norm of the current vector, i.e. total velocity. We add to this the Absolute Angular Error (AAE) (in °) to measure the error of the current direction. However, for low-intensity currents, a small error in either of the two components leads to a large angle error. Therefore, we also include, $\text{AAE}_{25}$, $\text{AAE}_{50}$, $\text{AAE}_{75}$, the angular errors for currents above $25, 50, 75$ (cm/s), respectively. For more clarity, we only compute the score on the ensemble estimations of the currents.

| | Input data | $\overline{\mu_u}$ | $\overline{\mu_v}$ | $\overline{\mu_n}$ | $\overline{\text{AAE}}$ | $\overline{\text{AAE}}_{25}$ | $\overline{\text{AAE}}_{50}$ | $\overline{\text{AAE}}_{75}$ |
|---|---|---|---|---|---|---|---|---|
| NATL60 geo | GT SSH | 5.3 | 4.4 | 5.5 | 17 | 5 | 3 | 3 |
| Bicubic geo | SSH | 18.4 | 16.7 | 21.1 | 42 | 25 | 19 | 15 |
| RESAC geo | SSH | 18.1 | 19.9 | 20.9 | 44 | 25 | 18 | 14 |
| RESAC geo | SSH + SST | 14.5 | 17.5 | 17.7 | 40 | 17 | 10 | 7 |
| RESAC UV | SSH | 15.5 | 15.2 | 16.8 | 43 | 24 | 17 | 13 |
| RESAC UV | SSH + SST | 11.6 | 11.1 | 12.3 | 40 | 15 | 8 | 6 |
| $\text{RESAC}_{\text{SUB}}$ geo | SSH | 22.8 | 24.9 | 24.5 | 53 | 35 | 27 | 22 |
| $\text{RESAC}_{\text{SUB}}$ geo | SSH + SST | 15.6 | 15.0 | 15.7 | 45 | 20 | 11 | 8 |
| $\text{RESAC}_{\text{SUB}}$ UV | SSH | 15.4 | 15.2 | 16.7 | 45 | 24 | 16 | 12 |
| $\text{RESAC}_{\text{SUB}}$ UV | SSH + SST | **10.1** | **9.6** | **10.9** | **33** | **12** | **6** | **5** |

**Tab. 4.3.:** Current reconstruction comparison. The included metrics are $\overline{\mu_u}$, $\overline{\mu_v}$, $\overline{\mu_n}$ (in cm/s) the RMSE of the $u, v$, and of the current vector, and $\overline{\text{AAE}}$, $\overline{\text{AAE}}_{25}$, $\overline{\text{AAE}}_{50}$, $\overline{\text{AAE}}_{75}$ (in °), the Absolute Angular Error of the total currents, and the currents above $25, 50, 75(cm/s)$.

First, it seems that the geostrophy can be considered a good approximation in this area, as applying it to ground truth SSH produces small errors: 5.3 and 4.4 (cm/s) for $\mu_u$ and $\mu_v$ respectively. This error is small compared to even the best mapping methods, and to the ground truth standard deviation of 39 and 34 (cm/s) (see Table **4.1**). The impact of SST is even greater on the currents than on SSH, with $35\%$ and $38\%$ of RMSE improvement for $u$ and $v$ respectively, using $\text{RESAC}_{\text{SUB}}$ UV. As we supposed, the geostrophic currents from $\text{RESAC}_{\text{SUB}}$ are worse than the ones of RESAC, even if its SSH is better reconstructed. This is a consequence of the checkerboard artifacts described earlier, as computing spatial gradients of a SSH with high-frequency errors produces strongly noised gradients.

In Figure **4.9** we plot the SSH maps and associated currents of the ground truth, the one of RESAC and $\text{RESAC}_{\text{SUB}}$ using SST, $\text{RESAC}_{\text{SUB}}$ UV SSH and $\text{RESAC}_{\text{SUB}}$ UV SSH+SST. We also plot the Relative Vorticity (RV) $\xi$ of the currents defined as follows:

$$\xi = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \tag{4.4}$$

Relative vorticity is an important quantity in the analysis of surface circulation as it highlights areas of significant rotational vector field. It is positive in counterclockwise rotations and negative in clockwise rotations, whereas a value close to zero indicates no direction changes. Here, we normalize relative vorticity fields by dividing by the Coriolis factor $f$. By comparing the RV of the geostrophic currents of RESAC and $\text{RESAC}_{\text{SUB}}$, we see that the artifacts lead to important high-frequency orientation mistakes. On the other hand, the currents estimated by $\text{RESAC}_{\text{SUB}}$UV SSH+SST present little to no high-frequency artifacts, and their relative vorticity maps look closer from the ground truth than all the other methods. Specifically, we notice the improvement brought by the SST in circulation estimation, as $\text{RESAC}_{\text{SUB}}$UV SSH relative vorticity map is much much smoother, and fine structures are not well recovered.

## 4.4 Conclusion

This chapter addressed the SSH downscaling challenge using a deep neural network and data from the NATL60 physical simulation [Ajayi et al., 2020]. We began by under-sampling SSH fields to very coarse resolutions of approximately 120 km per pixel. The downscaling process is conducted in three consecutive steps, each applying a scaling factor of 3. Each step utilized networks with identical architectures but distinct weights, processing low-resolution SSH images and, in some cases, high-resolution SST images. Our comparison involved two architectures: a fully convolutional CNN (RESAC) and a subpixel convolution residual network ($\text{RESAC}_{\text{SUB}}$). Additionally, we explored including a final CNN block to jointly estimate SSH fields and surface currents in the network's final stage.

We demonstrate the importance of incorporating high-resolution SST images as contextual data to enhance the downscaling process. By utilizing temperature data, all tested networks successfully reconstructed more accurate SSH fields, capturing finer resolution structures. This enhancement is particularly evident in current estimations, where the RMSE is significantly lower. Visual comparisons further

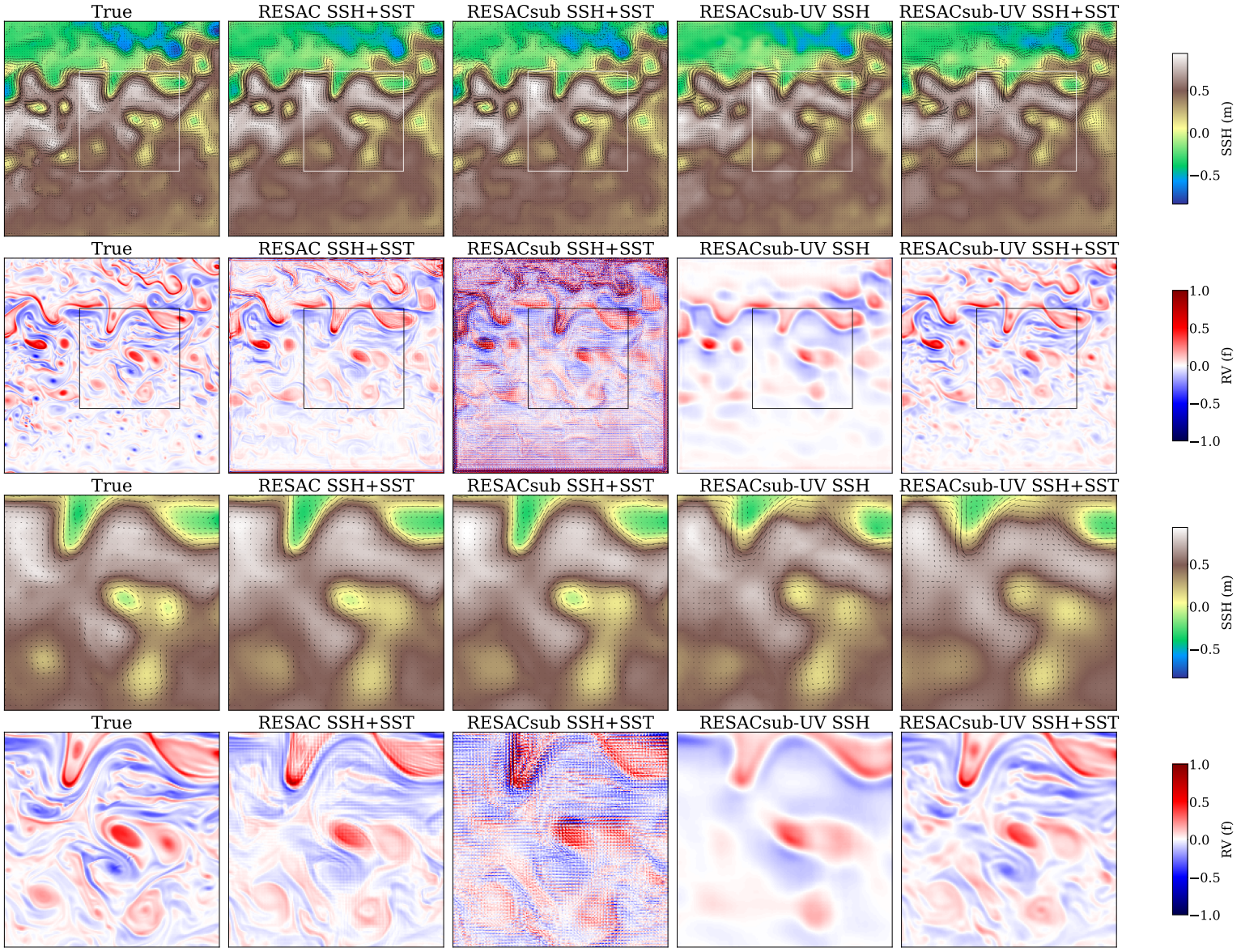


**Fig. 4.9.:** Plot of the SSH fields and associated currents and relative vorticities. We present the fields from the ground truth, RESAC-ssh-sst, RESACSUB-ssh-sst geostrophy, and RESACSUB-ssh and RESACSUB-ssh-sst estimated currents (though an additional CNN)

highlight the detailed, high-resolution currents achieved through SST-using methods. We show that RESACSUB outperforms its fully convolutional counterpart, as skip connections effectively address the vanishing gradient problem, and subpixel convolution serves as a trainable up-sampling method. Notably, while controlling the network at multiple resolutions significantly enhances RESAC performance, it is not required for RESACSUB. This underscores the need for innovative training strategies for very deep networks when dealing with substantial downscaling ratios, such as $\times 27$, whether through multi-resolution losses or adapted architectures.

However, this study is limited to simulation data. In the next chapter, we discuss these difficulties and motivate the transition from the downscaling approach to the interpolation approach.

# 5 Sea Surface Height interpolation

## 5.1 Introduction

### 5.1.1 From downscaling to interpolation

In the previous chapter, we tackled the SSH downscaling problem and showed that SST enabled the reconstruction of high spatial frequencies in the images. It corresponds to a setting where we aim to increase the quality of an existing SSH L4 product, such as DUACS, with low effective resolution [Amores et al., 2018, Stegner et al., 2021]. However, applying it in operational contexts is difficult because the decimation estimator is hard to model, as it would require replicating DUACS product. Because of this, applying a DL downscaling algorithm seems challenging as an inaccurate simulation of the low-resolution input resolves in an inaccurate super-resolved SSH field. This is a typical domain gap issue where the simulated and target tasks are similar but not the data distribution (see Section **3.3.1**). To overcome this limitation, performing an OSSE replicating the DUACS mapping on simulated data is possible. This approach was explored by [Nardelli et al., 2022] using one year of *pseudo-duacs* product on the Mediterranean Sea. However, as we do not have access to the full DUACS product chain, we cannot simulate as much data as required.

Instead of simulating DUACS L4 product, we decided to step down one level in the SSH data preprocessing stages. In the following chapter, we no longer work with low-resolution L4 SSH products but with sparse L3 SSH observations before any interpolation operation. Moving from a downscaling task to an interpolation task presents major advantages: first, the observation operator is easier to simulate, as no interpolation process occurs. It implies selecting the data in the region observed by the satellite and adding noise to mimic instrumental error. Twin experiments will be more accurate as pseudo-observations will be closer to their real-world equivalent. Second, this new observation operator has some interesting properties, allowing training neural networks from observations alone, thereby reducing the domain

gap problem. This chapter focuses on Delayed Time (DT) interpolations, meaning that we consider that we have access to observations in the future of the evaluation day.

### 5.1.2 Existing OSSE and OSE for SSH interpolation

As described in Section **3.1.4**, comparing interpolation methods requires a shared dataset and metrics. Through the years, the ocean altimetry community has developed "Ocean data challenges", providing twin experiments of the interpolation problem. We describe hereafter the Ocean Data Challenge 2020 (ODC2020) [CLS/MEOM, 2020] and the Ocean Data Challenge 2021 (ODC2021) [CLS/MEOM, 2021], twin experiments on a simulation and real observations, respectively. The interpolations considered in the following challenges focus on SSH-only, Delayed Time (DT) interpolations, but it is still possible to adapt them to multivariate interpolations.

#### Ocean data challenge 2020 OSSE

The ODC2020 is a one-year OSSE on the NATL60 simulation [Ajayi et al., 2020] (see Section **4.1**). The included data are the ground truth SSH and a simulation of nadir-pointing and SWOT observations. Given that the NATL60 model also outputs SST and ocean currents fields, we retrieved and used these variables, even though they were not included in the official depository of the challenge. The studied area is a portion of the Gulf Stream area (43° to 33° North and -65° to -55° East). The nadir-pointing altimeters considered are Topex-Poseidon, Jason 1, Geosat, and Envisat (between 2003 and 2005). The instrumental errors of both the altimeters and SWOT are simulated through the *swot-simulator* software [Gaultier et al., 2016]. On the one year of observations (from 2012-10-01 to 2013-09-30), the test period is 42 consecutive days between 2012-10-22 to 2012-12-01. For data requiring training from complete fields, the data between 2013-01-02 to 2013-09-30 can be used, but leaving aside the data from 2012-12-02 and 2013-01-02 for decorrelation purposes.

#### Ocean data challenge 2021 OSE

The ODC2021 provides one year of data in the same Gulf Stream area, but unlike ODC2020, it focuses on real observations. The SSH measurements were taken in 2017 from nadir-pointing satellites: SARAL/Altika, Jason 2, Jason 3, Sentinel 3A,

Haiyang-2A, and Cryosat-2. The Cryosat-2 data must not be used in the interpolation to compare reconstruction methodologies and will serve as independent testing data. This validation data presents instrumental errors (see Section **2.2.3**), leading to overestimating the interpolation error. As we cannot access a fully gridded reference, all the reconstruction scores must be computed on the along-track measurement from Cryosat-2, limiting the metrics' choice. As in the ODC2020, no SST data was officially included in the repository of the challenge, nor current estimations. However, as this challenge focuses on real-world data, it is possible to use the satellite SST of the corresponding days and area, as well as *in situ* current measurements. After some experiments, we decided to use the Multiscale Ultrahigh Resolution SST (MUR SST) L4 product [NASA/JPL, 2019] (described in Section **5.5.1**) and global Ocean-Delayed Mode *in situ* Observations of surface drifters measurements from SEANOE [SEANOE, 2024].

### Metrics

The ODC2020 and ODC2021 define several metrics to compare several interpolations.

- $\mu$ and $\sigma_t$ (in cm) are the RMSE of the SSH and the temporal standard deviation of this RMSE. In the data challenge, the RMSE is normalized by the Root Mean Square of the target SSH to take into account the intensity of the variations of the target signal. The normalized RMSE is defined as follows:

$$\mu_{nor} = 1 - \frac{\mu}{\mathrm{RMS}(\mathbf{X}^{\mathrm{ssh}})} \tag{5.1}$$

- $\lambda_x$ (in degrees) and $\lambda_t$ (in days) are two spectral metrics, introduced by [Le Guillou et al., 2020]. We compute respectively the spatial and temporal power spectrum of the error, $\lambda_x$ is then the smallest spatial wavelength where the power spectrum of the error is equal to the power spectrum of the signal and $\lambda_t$ its temporal equivalent. The two metrics can be seen as the effective resolution of the reconstruction, as they correspond to the smallest wavelength, which is at least half resolved, in space and time. For further information, we refer the reader to [Le Guillou et al., 2020].

- $\mu_u$ and $\mu_v$ (in cm/s) are the RMSE between the currents and the geostrophic currents of the estimation.

In ODC2020, as we can access complete references, we can compute all these metrics on the entire maps. However, on the ODC2021, the reference data is left

aside from measurements from one satellite. To be able to compute these metrics, we first linearly interpolate the reconstruction method along the path of the reference satellite and then compute the metrics. This is why, in ODC2021, it is not possible to compute the temporal effective resolution of the reconstructions, as there are not enough consecutive points in time and space. Also, the estimation of currents is evaluated on drifters only, using the same process of interpolating the estimated current map at the location of the drifter.

### State-of-the-art interpolation methods

We present the state-of-the-art interpolation methods applied to at least one of the data challenges.

- DUACS is the operational linear optimal interpolation leveraging covariance matrix tuned on 25 years of data [Taburet et al., 2019] as described in Section **3.2.1**.
- DYMOST or Dynamic Optimal Interpolation (DOI) [Ubelmann et al., 2016, Ballarotta et al., 2020]: a dynamical OI, where a Quasi-Geostrophic (QG) model (see Section **2.4.2**) is used forward and backward to predict the evolution of SSH. Compared to DUACS, the spatiotemporal Gaussian kernel used to compute covariances matrices is replaced by a non-linear kernel dynamically propagating the SSH contributions through the QG model. The obtained covariance model includes statistical and physical information and is then used in a standard BLUE analysis. For more details about dynamical mapping, we refer to the reader to [Ubelmann et al., 2016].
- MIOST or Multiscale and multivariate Optimal Interpolation (MOI) [Ardhuin et al., 2020]: an OI where DUACS covariance is replaced by a wavelet decomposition. Each wavelet variance is determined by looking at their variance in nadir-altimetry measurement. Originally, this method was developed to simultaneously interpolate SSH and ageostrophic currents, leveraging future Spaceborne Doppler data, but the version provided in the ODC2020 and ODC2021 uses no information from the currents.
- Back and Forth Nudging Quasi-Geostrophy (BFN-QG) [Le Guillou et al., 2020]: a data assimilation method that performs a back and forward nudging of a QG model (see Section **3.2.4**).
- 4 Dimensional Variational Network (4DVarNet) [Fablet et al., 2021]: deep learning model supervised using reference field. Its architecture relies on

the variational formulation of the interpolation problem of 4D-VAR (see Section **3.2.3**). But unlike 4D-VAR, 4DVarNet does not use a physical model to constrain the inversion. Instead, it simultaneously learns the dynamical model, the prior, and the solver of the variational problem using the ground truth of a simulation. In inference, it performs several gradient steps of its solver (typically 15), by using the automatic differentiation implemented in the solver architectures to fit accurately the input observations. This is thus a hybrid method between standard neural network training and variational data assimilation, where the models are trained, but a variational minimization is applied to new examples instead of a direct neural network inference. [Fablet et al., 2021] showed its ability to learn SSH interpolation on OSSEs, and [Fablet et al., 2023] extended its study by showing the reconstruction improvement brought by the SST. When applied to OSE data, 4DVarNet exhibits interesting performances, but to the best of our knowledge, no real-world application using SST was achieved by the authors.

- Convolutional Long-Short Term Mermory (ConvLSTM): [Martin et al., 2023] introduced a neural architecture composed Convolutional Long-Short Term Mermory (ConvLSTM)-based spatiotemporal encoder decoder. It is composed of CNN spatial encoder and decoder, compressing or decompressing SSH and SST information one timestep at a time. A Convolutional Long-Short Term Mermory (ConvLSTM) is used as a temporal encoder, using encoder information from each timestep. This network was trained on observations directly by training the neural network to estimate SSH data removed from its output. To our knowledge, it was only applied to real-observations, unlike 4DVarNet.

### 5.1.3 Objectives and contributions

The most important motivation for this work is to get toward an operational DL framework able to produce high-resolution SSH fields using SSH observations and SST contextual variables. One of the key features to achieve this objective is to find a way to train the network directly on observations, thereby avoiding the domain gap problem.

With this objective in mind, we first explore in Section **5.2** the unsupervised inversion in a Deep Image Prior (DIP) strategy. Through these experiments, we show that it is possible to use the inductive bias of a neural network to perform the interpolation on a single example of observations. We test this on the ODC2020, a simulation,

but without using the reference field. This work was published in [Filoche et al., 2022].

In Section **5.3**, we adapt the DIP idea to be able to include SST contextual information and multiple examples. We test our idea on ODC2020 pseudo-observations and the ODC2021 observations. This work was published in [Archambault et al., 2023].

However, these two methods have an important drawback: they must be refitted to new examples by gradient descent as standard variational DA frameworks. This leads to high computational costs, which limits their operational use. Usually, neural networks are trained offline and applied to new data by a single inference, which is less computationally intensive than performing many gradient backpropagations. To address this issue, we are interested in training a neural network so that it can be directly applied to unseen data but still in an unsupervised manner. To this end, in Section **5.4**, we introduce a new OSSE, well suited to DL training, and compare supervised and unsupervised methods. This work was published in [Archambault et al., 2024b].

While comparing unsupervised and supervised training on our OSSE, we remarked that unsupervised methods had significantly lower performances than their supervised equivalents. Learning the physical relationship and correlation between variables using only observations is a hard task. To improve the reconstructions, we present a method to adapt the physical knowledge embedded in the simulation to real observations. This method involves pre-training on an OSSE and fine-tuning on an OSE, and leads to improved performances and is presented in Section **5.5**. This work was published in [Archambault et al., 2024a] and in [Archambault et al., 2024b].

## 5.2 Training without complete SSH reference

### 5.2.1 Inspiration: Deep Image Prior

To train on real-world observations only, we are interested in defining a loss function that does not require complete SSH fields. Deep Image Prior (DIP) [Ulyanov et al., 2017], tackle the image inpainting problem by fitting a neural network on a single example (see Section **3.3.1**). In this setting, the network $g_\theta$ starts from a random vector $z$, and estimates the SSH field as $g_\theta(z) = \widehat{\mathbf{X}}$. Before applying the loss function, the observation operator is applied to the estimated state, which results in $\widehat{\mathbf{Y}}$ a vector comparable to observations $\mathbf{Y}$. $g_\theta$ is fitted by gradient descent on a single example,

as 3D-VAR would be, but without explicit covariance matrix (see Section **3.2.1**). Instead, the neural network architecture itself forces correlation between the output pixels. To adapt this idea in the context of SSH interpolation, we now consider that the observations $\mathbf{Y}^{\text{ssh}}$ are a time series of 32 images (one per day), and the observation operator $\mathcal{H}^{\text{ssh}}$ simply masks the pixels where no data is observed. The loss to be optimized is the following:

$$\mathcal{L}(\mathbf{Y}^{\text{ssh}}, g_\theta(z)) = \text{MSE}(\mathbf{Y}^{\text{ssh}}, \widehat{\mathbf{Y}}^{\text{ssh}}) = \text{MSE}(\mathbf{Y}^{\text{ssh}}, \mathcal{H}^{\text{ssh}} \circ g_\theta(z)) \tag{5.2}$$

where $\mathcal{H}^{\text{ssh}}$ is the masking operator corresponding to the observations support. Formally, $\mathcal{H}^{\text{ssh}}$ can be written as the Hadamard product between the input and the observation mask $\Omega$ ($1$ if the pixel is observed, $0$ if not).

$$\mathcal{H}^{\text{ssh}}(\mathbf{X}) = \mathcal{H}_\Omega(\mathbf{X}) = \mathbf{X} \odot \Omega \tag{5.3}$$

The DIP principle is summarized in Figure **5.1**.

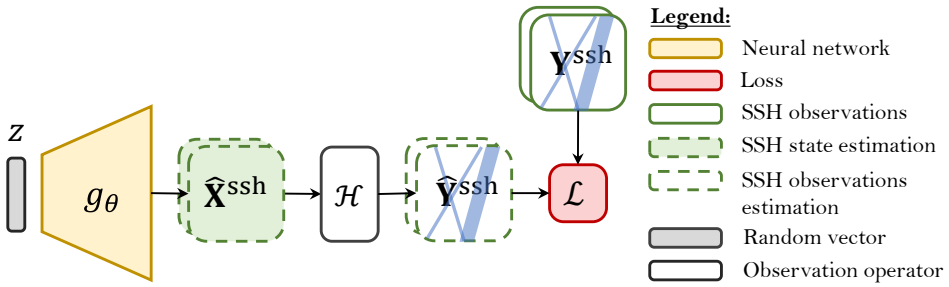


**Fig. 5.1.:** Computational graph of the Deep Image Prior SSH interpolation.

## 5.2.2 Application to SSH interpolation

### Experiment

We test this principle on NATL60 OSSE provided by [CLS/MEOM, 2020], on a small area from 36.5° to 43° North and -65° to -58.5° East. We focus on interpolation from SWOT and nadir-pointing observations. In Figure **5.2**, we show an example of reference SSH and the observations, where we only represent one day out of four. Due to its $\sim 10$ days return time, SWOT data is not available on a daily basis as observations were only made on days 4, 16, 24, and 28.

DIP relies on the idea that using a well-suited neural network to generate the solution of a variational problem can act as a handcrafted regularization, leveraging

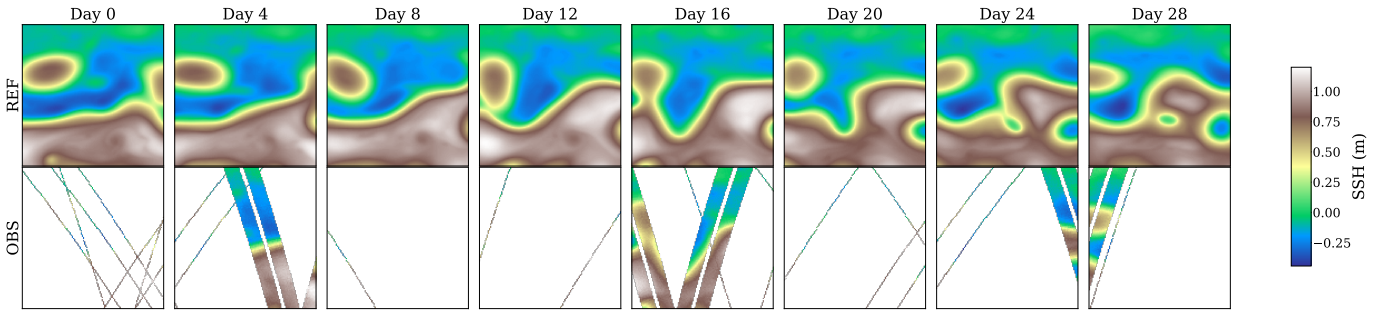


**Fig. 5.2.:** The times series of the NATL60 SSH reference and the associated observations.

spatial and, eventually, temporal bias induced by the architecture. The choice of architecture is thus a crucial component of the method, as it must perform operations with appropriate invariance properties. 2D convolutions are invariant by spatial translation, and 3D convolutions extend this invariance to the third dimension, in our case, the temporal dimension (see Section **3.3.2**). We design our architecture to take advantage of this inductive bias. It is largely inspired by generative convolutional architectures introduced in [Radford et al., 2016], but we replaced deconvolution operations with trilinear upsampling, as described in [Odena et al., 2016] to avoid checkerboard artifacts. Finally, to ensure spatiotemporal coherence of the generated solution, we used *Conv2D+1* introduced by [Tran et al., 2018](see Section **3.3.2**) instead of standard 2D convolutions. The architecture is summarized in Figure **5.3**. We fit the network on one example during 2000 epochs with an ADAM optimizer with parameters $\beta_1 = 0.9$, $\beta_2 = 0.99$, and learning rate $1.10^{-2}$.

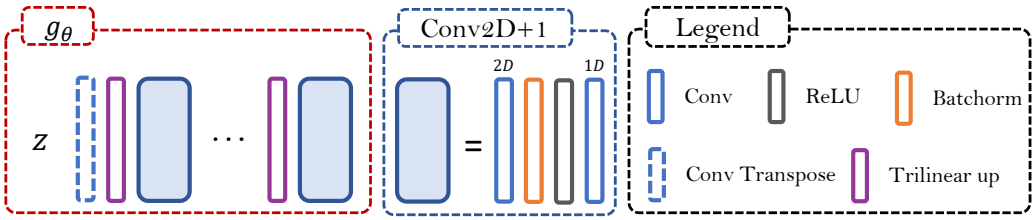


**Fig. 5.3.:** Architecture of the spatio-temporal Deep Image Prior generator $g_\theta$.

### Results

DUACS is a OI with tunned covariance matrices described in Section **3.2.1**, meaning that each pixel is estimated from the observations by weighting the contributions of each measurement according to their covariance to the target point. In DIP, on the other hand, the covariance between observed and unobserved pixels is ensured by the internal structure of the network. Unlike DIP, if the two methods require

only observations, DUACS needs tuning covariance matrices (on 25 years of data). Because of these similarities, comparing DIP and DUACS is a natural thought.

We display in Figure **5.4**, the reference SSH field, with DIP and DUACS estimations, and in Figure **5.5** we plot their error maps. We see that the two methods produce similar fields and associated error maps, missing the same kind of structures, even if DIP errors seem visually a bit less intense.

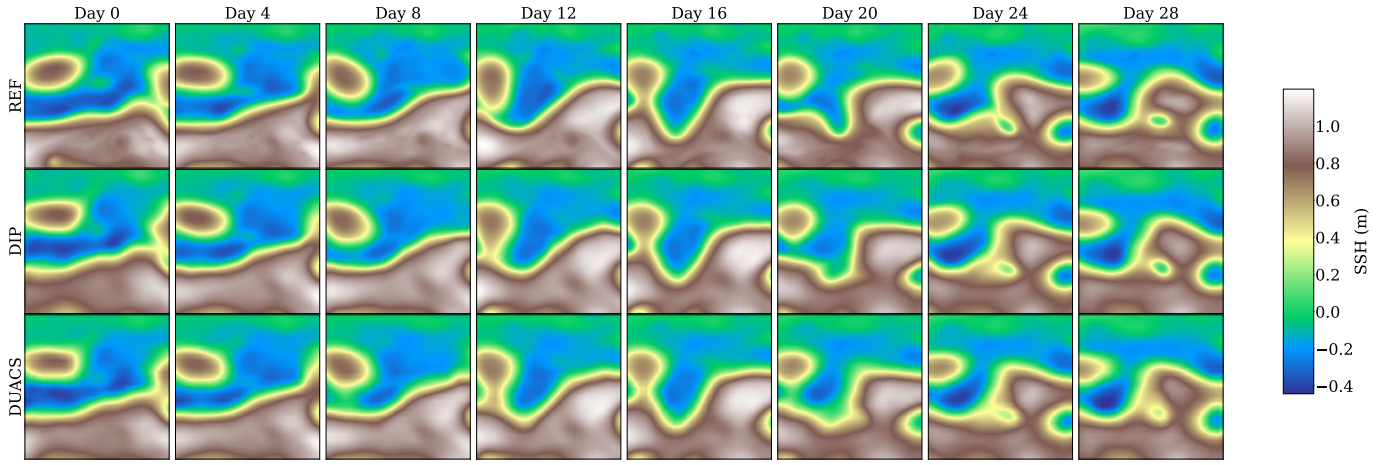


**Fig. 5.4.:** Plot of the SSH reference, DIP and DUACS estimations from the inputs showed in Figure **5.2**.

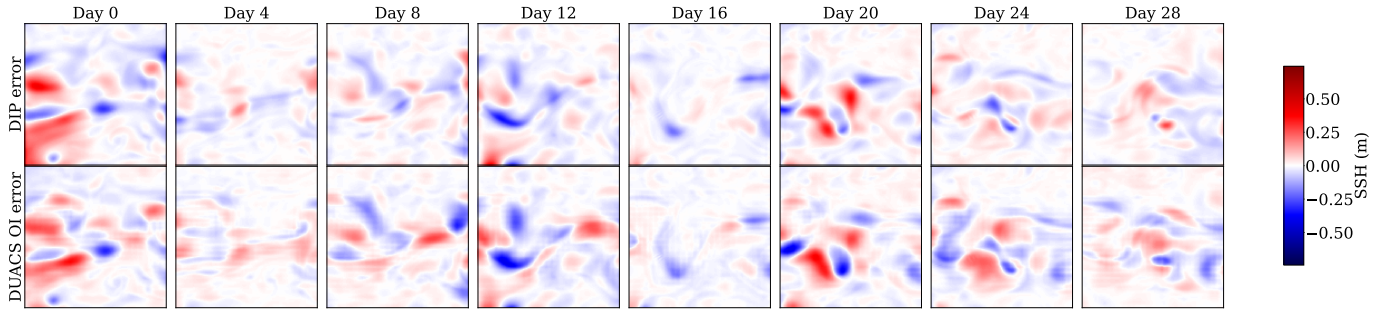


**Fig. 5.5.:** Plot of the error maps of DIP and DUACS from the inputs showed in Figure **5.2**.

To show the importance of the neural architecture inductive bias, we present an ablation study of the *Conv2PD1* by replacing it with standard 2D convolutions. We computed an ensemble of 30 DIP estimations on the same example with the two architectures. We plot in Figure **5.6** the scores along the 32 days for *Conv2PD1* (upper part), *Conv 2D*, (lower part), and DUACS. Without the temporal invariance embedded in the architecture, we see that daily images are not correlated between timesteps, and DIP completely fails to interpolate the SSH. This result highlights

the interest in choosing 3D convolutions over 2D when dealing with spatiotemporal interpolation.

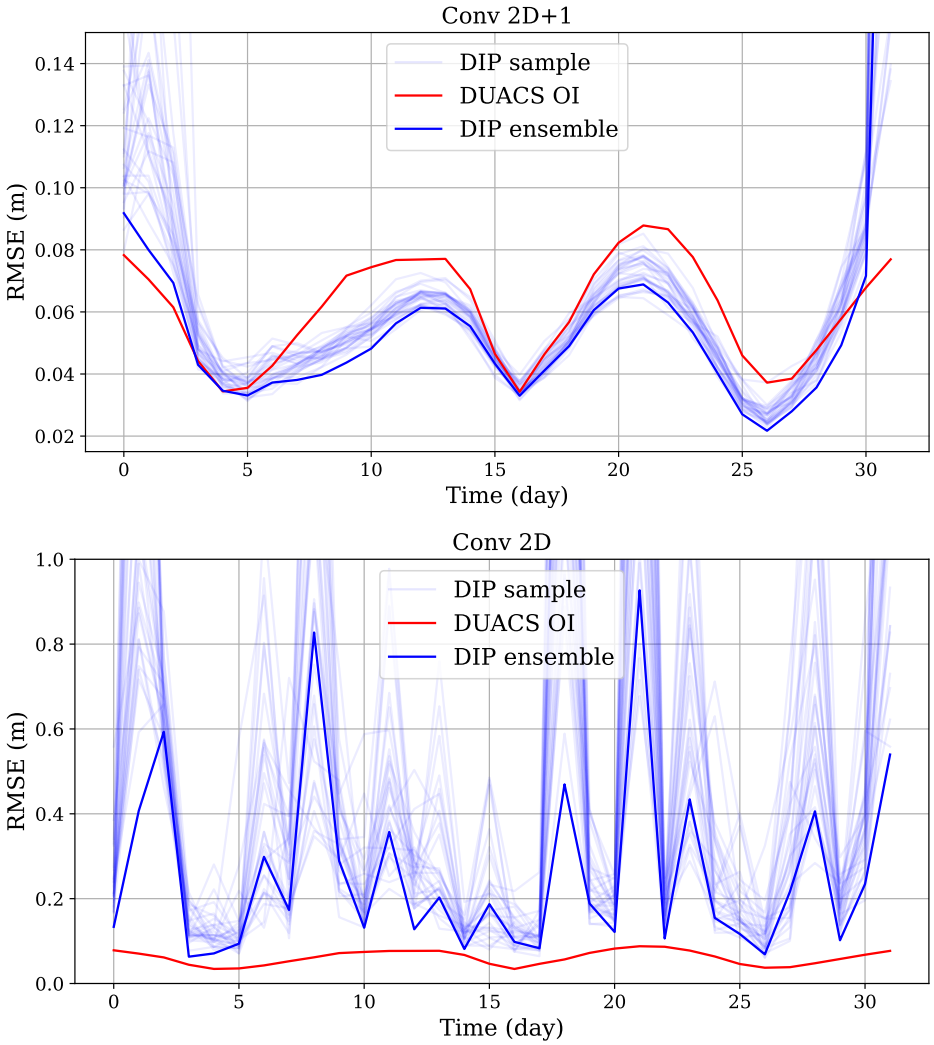


**Fig. 5.6.:** RMSE along one window for *Conv2D+1* and *Conv2D*

### Comparison of DIP and DUACS on ODC2020 and ODC2021.

To compare the errors on the entire dataset, extend DIP to the area of the data challenges spatially and temporarily. In space, we perform a last linear upsampling before computing the loss so that the size of $\widehat{\mathbf{X}}^{\text{ssh}}$ covers the entire reference. We use a sliding window approach, where we fit one DIP by timestep and average DIP estimations from matching instants. However, to mitigate border effects, we exclude the initial and final 8 days of each window. In Figure **5.7**, we plot the RMSE of DUACS and DIP using nadir-pointing and SWOT pseudo-observations, where in

Figure **5.7** we plot the same methods but for nadir-pointing only. We see that the long SWOT return time leads to a periodic cycle in the reconstruction error, which does not appear when using only nadirs. In the two settings, DIP systematically outperforms DUACS, which suggests that its internal covariance is better suited to SSH reconstruction than the linear optimal interpolation covariance matrices.

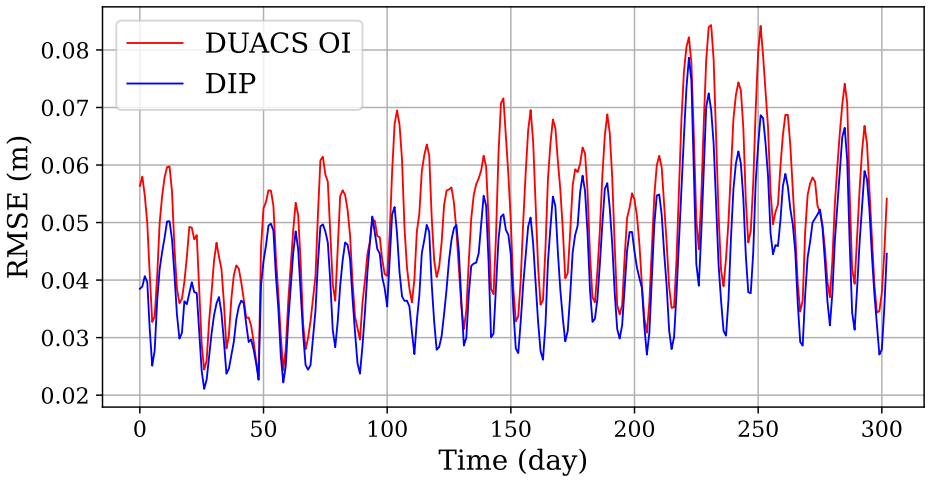


**Fig. 5.7.:** RMSE comparison between DUACS and DIP on the ODC2020, using nadir-pointing and SWOT observations.

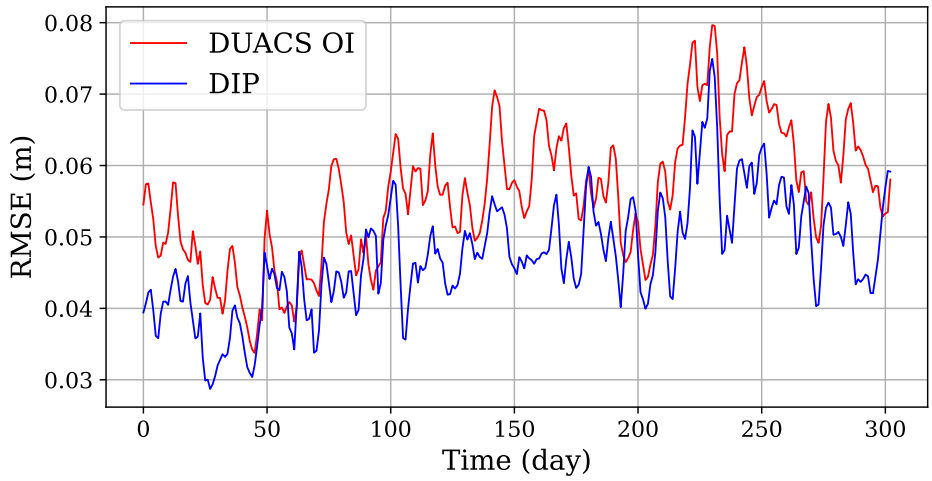


**Fig. 5.8.:** RMSE comparison between DUACS and DIP on the ODC2020, using only nadir-pointing observations.

In Table **5.1**, we compare the scores of DIP and DUACS reconstructions on ODC2020, and in Table **5.2**, we do the same on the real observations from ODC2021. On ODC2020, DIP decreases the reconstruction of 1 cm (∼21%) in the nadir only setting and of 0.91 cm (∼20%) SWOT + nadir setting, respectively. It also leads to a better

estimation of the geostrophic surface currents, $\mu_u$ and $\mu_v$, as shown in Table **5.1**. The SSH reconstruction improvements are similar in the real-world setting with a decrease of 1.02 cm of RMSE. However, $\mu_v$ is higher for DIP, contrary to the simulation setting. Anyway, in these three comparable benchmarks, it seems that it is possible to replace OI with DIP with similar or even enhanced performances.

| Method | $\mu(norm)$ | $\mu(cm)$ | $\sigma_t(cm)$ | $\lambda_x(°)$ | $\lambda_t(days)$ | $\mu_u(cm/s)$ | $\mu_v(cm/s)$ |
|---|---|---|---|---|---|---|---|
| DUACS nadir | 0.916 | 4.89 | 3.02 | 1.42 | 12.1 | 16.7 | 16.2 |
| DUACS swot + nadir | 0.922 | 4.56 | 2.85 | **1.22** | 11.3 | 16.0 | 15.0 |
| DIP nadir | 0.933 | 3.88 | **1.96** | 1.36 | **9.6** | **13.2** | 14.4 |
| DIP swot + nadir | **0.937** | **3.65** | 2.01 | **1.22** | 9.8 | 13.3 | **13.0** |

**Tab. 5.1.:** Comparison between DUACS and DIP errors, on ODC2020.

| Method | $\mu(norm)$ | $\mu(cm)$ | $\sigma_t(cm)$ | $\lambda_x$(km) | $\mu_u(cm/s)$ | $\mu_v(cm/s)$ |
|---|---|---|---|---|---|---|
| DUACS nadi r | 0.877 | 7.31 | 2.37 | 149 | 20.6 | **20.0** |
| DIP nadir | **0.890** | **6.29** | **2.03** | **128** | **20.1** | 21.6 |

**Tab. 5.2.:** Comparison between DUACS and DIP errors, on ODC2021.

## 5.2.3 Conclusion

### Summary and conclusions

With this proof of concept, we show that it is possible to replace DUACS covariance matrices tuned on 25 years of observations by the inductive bias of a well-chosen neural architecture fitted on only one example. In doing so, we even benefit from a significant performance gain. This is a strong result showcasing the interest of neural architectures as handcraft regularization, but also a promising perspective to training neural networks with only observations. However, DIP has some drawbacks that limit its operational use, and that we detail in the next paragraph.

### Limitations

The first obvious drawback of this method is that it cannot, in its current form, include the information from the SST in the reconstruction. As shown in Chapter **4**, SST is a key variable to recover high-frequency details of the SSH state. This motivated us to find a Multi-variate Unsupervised Spatio-Temporal Interpolation (MUSTI) method, inspired by DIP, but enabling SST use. This work is described in the next section.
Another drawback of DIP is that we need to fit a neural network for each sample. Even if over-fitting observations on a single example is quicker than standard training,

doing it for each new time step and area leads to a computational burden. We tackle this problem in Section **5.4** by training a neural network on a 20-year OSSE and to observations in an unsupervised manner. Once trained, the network interpolates new examples by a simple inference without refitting, significantly reducing the computational cost.

## 5.3 Multi-variate Unsupervised Spatio-Temporal Interpolation

### 5.3.1 Reshaping the Deep Image Prior idea to include temperature data

In DIP, the neural networks starts from a random vector with no information, as it is chosen randomly. To adapt this idea to include SST information, we must find a way to introduce knowledge in the latent space $z$. This can no longer be called a Deep Image Prior strategy, as DIP exclusively relies on the prior brought by the neural architecture. Similarly to Section **5.2**, all the data considered here are spatio-temporal 3D images corresponding to a time window of observations. This approach called Multi-variate Unsupervised Spatio-Temporal Interpolation (MUSTI), was published in [Archambault et al., 2023], but under the term "Multimodal" instead of "Multivariate." Afterward, we decided to use this last term as it is better suited to designate observations from multiple physical variables.

As suggested by the manifold hypothesis [Fefferman et al., 2016], physical data can be seen as high-dimension observations taken from the same underlying representation. This means that the ocean state can be parsimoniously encoded in a vector, $z$, of a much smaller dimension than the observations. In DL, this is the core principle of auto-encoders: neural networks capable of synthesizing information in a small number of variables and decoding it (see Section **3.3.2**). Considering the above arguments, using an encoder-decoder framework seems appropriate. We use a deep neural network to encode SST observations $\mathbf{Y}^{\text{sst}}$ in the latent space $z$ as in Equation **5.4**. Then, a decoder estimates the SSH image $\widehat{\mathbf{X}}^{\text{ssh}}$ using the information encoded in $z$ as in Equation **5.5**. Hereafter the encoder will be denoted $f_{\theta_1}$, the decoder $g_{\theta_2}$ and the encoder-decoder network $h_\theta = g_{\theta_2} \circ f_{\theta_1}$. Other architectures can be used as well through a direct multi-variate information transfer from $\mathbf{Y}^{\text{sst}}$ to $\mathbf{X}^{\text{ssh}}$ as long as they bring an inductive bias helping the reconstruction. To fit the neural

network using only observations, similarly to DIP, we apply the masking operator $\mathcal{H}^{\text{ssh}}$ to the estimation (Equation **5.6**) before computing the loss (Equation **5.7**).

$$\textbf{Encoding:} \qquad z = f_{\theta_1}\left(\mathbf{Y}^{\text{sst}}\right) \tag{5.4}$$

$$\textbf{Decoding:} \qquad \widehat{\mathbf{X}}^{\text{ssh}} = g_{\theta_2}\left(z\right) \tag{5.5}$$

$$\textbf{Masking:} \qquad \widehat{\mathbf{Y}}^{\text{ssh}} = \mathcal{H}^{\text{ssh}}\left(\widehat{\mathbf{X}}^{\text{ssh}}\right) \tag{5.6}$$

$$\textbf{Loss:} \qquad \mathcal{L}\left(\mathbf{Y}^{\text{ssh}}, \widehat{\mathbf{Y}}^{\text{ssh}}\right) = \text{MSE}(\mathbf{Y}^{\text{ssh}}, \widehat{\mathbf{Y}}^{\text{ssh}}) \tag{5.7}$$

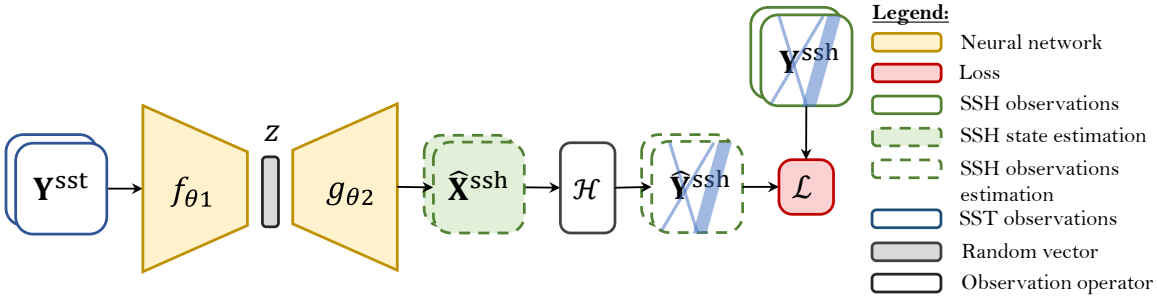


**Fig. 5.9.:** MUSTI computational graph.

## 5.3.2 Experiments

Considering the setting described in the last section, we can distinguish two manners of fitting MUSTI. First, we can fit it in a standard DIP way, over-fitting SSH tracks a single example, and compute a spatiotemporal window to produce a complete dataset. But as the latent variable $z$ now carries information about the SST state, it is now possible to share the neural network's weights over multiple examples. In the following, we describe the Spatio-Temporal Auto-Encoder architecture used in MUSTI and the two training settings.

### Architecture

We extend the principle of the spatiotemporal decoder described in Figure **5.3** to build the spatiotemporal encoder-decoder presented in Figure **5.10**. It still relies on *Conv2D+1* operations as well in the encoder than in the decoder. The decoder alternates *Conv2D+1* with 2D max-pooling, each dividing spatial dimension by two (and not temporal dimension). The latent variable obtained has a shape $T \times h_z \times h_z$ and is given to the decoder. Except for the first deconvolution, the decoder is the same as the one used in DIP.

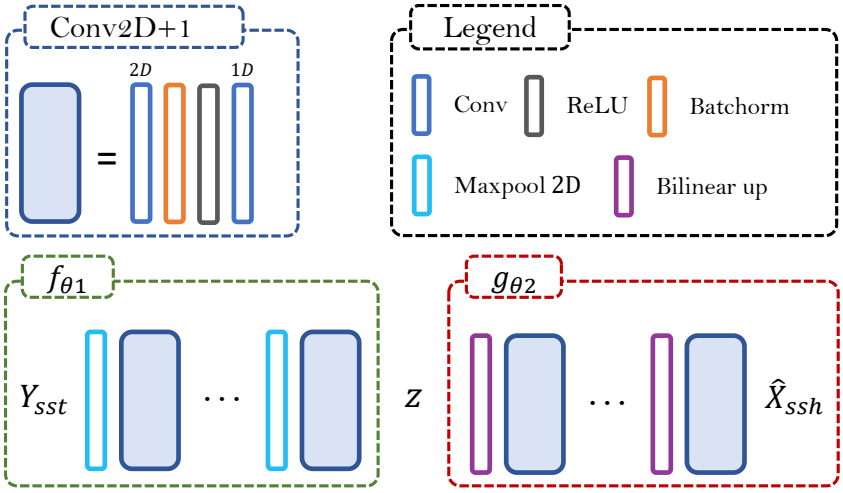


**Fig. 5.10.:** Architecture of the Spatio-Temporal Auto-Encoder used in MUSTI.

#### MUSTI on one example

The first way to train MUSTI is fitting one example at a time on a time series of 32 days, similar to DIP. We denote hereafter this training mode by $\text{MUSTI}_{\text{1BY1}}$. We keep the same training hyperparameters as in the DIP experiment: 2000 epochs with an ADAM optimizer with parameters $\beta_1 = 0.9$, $\beta_2 = 0.99$, and learning rate $1.10^{-2}$. The loss function is the MSE given in Equation **5.7**. The same training procedure is used in the 3 data settings: ODC2020 SWOT +nadir, ODC2020 nadir only, ODC2021 real nadir observations. The temporal sliding window is computed in the same way as DIP, still excluding 8 starting and final days, to avoid border issues.

#### MUSTI on a dataset

In the second method, we train MUSTI on a dataset with standard neural network training. Each dataset example is seen and backpropagated once during an epoch, and we repeat this process until convergence. In doing so, the neural network must learn SST encoding and decoding to SSH on a dataset of about one year. However, this neural network will not generalize well to unseen data as its principle still relies on overfitting SSH tracks. For each data scenario, we tune the window length $T$ and the stopping epoch on the validation dataset, as described in Table **5.3**. In the real-world scenario, there is no ground truth to serve as a validation dataset, so we leave aside the observations from a satellite (Jason-2g) to tune the model's hyper-parameters. Once these hyper-parameters are found on validation observations, we train another network with this set of parameters on the training and validation

set, but still without Cryosat-2 independent data. We compute an ensemble of 10 members to be comparable to DIP and the other MUSTI training procedure.

| Dataset | ODC2020 SWOT+nadir | ODC2020 nadir | ODC2021 nadir |
|---|---|---|---|
| T | 7 | 5 | 5 |
| Epoch | 94 | 57 | 50 |

**Tab. 5.3.:** MUSTI Optimal hyper-parameters on the validation dataset for each dataset.

### 5.3.3 Results

#### Performances on ODC2020 and ODC2021 datasets

We compute the reconstruction on ODC2020 and ODC2021 with the two training modes of MUSTI. We compare it to DIP, which is its SST-agnostic equivalent. We first present the scores in Tables **5.4** and **5.5**. The scores on ODC2020 support using SST in MUSTI with an error reduction of 1 cm when using SWOT+nadir data. The improvement brought by $\text{MUSTI}_{\text{1BY1}}$, on the other hand, is more contrasted. In the SWOT+nadir inversion, it brings a slight improvement compared to DIP, but has similar scores when using nadir-only data, and significantly lower performance on ODC2021. It shows that the gain brought by the SST also comes from the fact that it is now possible to perform the inversion on multiple examples. One of the most salient features of computing the inversion on the entire dataset using SST is that the effective resolutions of the estimated fields are much more accurate (at least on ODC2020). It supports further the conclusions of Chapter **4** that the SST is very helpful in retrieving high frequencies of the SSH field. This resolution improvement is less visible in real observation settings as the SST also presents some noise and smoothing because of the processing and cloud-gaps interpolation.

#### Visual comparison

In Figure **5.11**, we present an example of SSH and SST reference, as well as DIP and MUSTI estimated fields, alongside their respective error maps, on the 1st of November 2012. The estimation is computed on the ODC2020 dataset, leveraging SWOT+nadir observations. MUSTI field exhibits superior quality, characterized by enhanced resolution and visual details. A comparison between the DIP and MUSTI error maps reveals important error magnitudes in DIP's outputs, particularly in the highlighted regions. Notably, within areas denoted as 1 and 2, where significant SSH variations are present, the incorporation of temperature as a contextual variable

| Method | $\mu(norm)$ | $\mu(cm)$ | $\sigma_t(cm)$ | $\lambda_x(°)$ | $\lambda_t(days)$ | $\mu_u(cm/s)$ | $\mu_v(cm/s)$ |
|---|---|---|---|---|---|---|---|
| | SWOT+nadir | | | | | | |
| DIN | 0.937 | 3.65 | 2.01 | 1.22 | 9.8 | 13.3 | 13.0 |
| MUSTI$_{1BY1}$ | 0.942 | 3.38 | 1.59 | 1.22 | 9.3 | 12.9 | 12.8 |
| MUSTI | **0.954** | **2.64** | **1.21** | **0.62** | **3.4** | **10.5** | **11.1** |
| | nadir | | | | | | |
| DIN | 0.933 | 3.88 | 1.96 | 1.36 | 9.6 | 13.2 | 14.4 |
| MUSTI$_{1BY1}$ | 0.932 | 3.94 | 1.69 | 1.2 | 10.3 | 13.2 | 14.0 |
| MUSTI | **0.946** | **3.12** | **1.32** | **1.23** | **4.1** | **11.7** | **13.8** |

**Tab. 5.4.:** Comparison between DIN, MUSTI$_{1BY1}$, and MUSTI errors on ODC2020 for various metrics.

| Method | $\mu(norm)$ | $\mu(cm)$ | $\sigma_t(cm)$ | $\lambda_x$(km) | $\mu_u(cm/s)$ | $\mu_v(cm/s)$ |
|---|---|---|---|---|---|---|
| DIN | 0.89 | 6.29 | 2.03 | 128 | 20.1 | 21.6 |
| MUSTI$_{1BY1}$ | 0.883 | 6.71 | 2.38 | **112** | **19.3** | 19.9 |
| MUSTI | **0.894** | **6.08** | **1.77** | 113 | **19.3** | **19.4** |

**Tab. 5.5.:** Comparison between DIN, MUSTI$_{1BY1}$, and MUSTI errors on ODC2021 for various metrics.

aids in determining precisely the location of the spatial frontier. In regions 3 and 4, instances of local maxima are present in the SSH reference, which correspond to mesoscale anticyclonic eddies. These eddies exhibit discernible signatures in the temperature field, particularly for area 3. MUSTI can use these signatures to infer SSH accurately, thereby yielding improved eddies shapes and intensities.

### 5.3.4 Conclusion

In this work, we presented a method using SST information to enhance SSH interpolation in an unsupervised way: we compute the loss on observations only similar to DIN, but we start from a complete SST image, unlike DIN. This Multi-variate Unsupervised Spatio-Temporal Interpolation (MUSTI) shows promising performances once compared to DIN. On ODC2020, it decreases SSH RMSE by 1 cm and 0.76 cm when interpolating SWOT+nadir and nadir-only pseudo-observations, respectively. However, its improvement is smaller on ODC2021, with only 0.2 cm of RMSE gain.

However, MUSTI has limitations that restrict its operational use. Similar to DIN, MUSTI is trained to estimate one example or a small dataset accurately but needs to be refitted to be applied to unseen data. In an operational setting, this would necessitate significant computing resources, whereas standard neural networks should be able to generalize to unseen data without the need for retraining, making them much more efficient. To achieve this, we must incorporate SSH observations in inputs, whereas MUSTI currently only considers SST. With a larger dataset spanning multiple

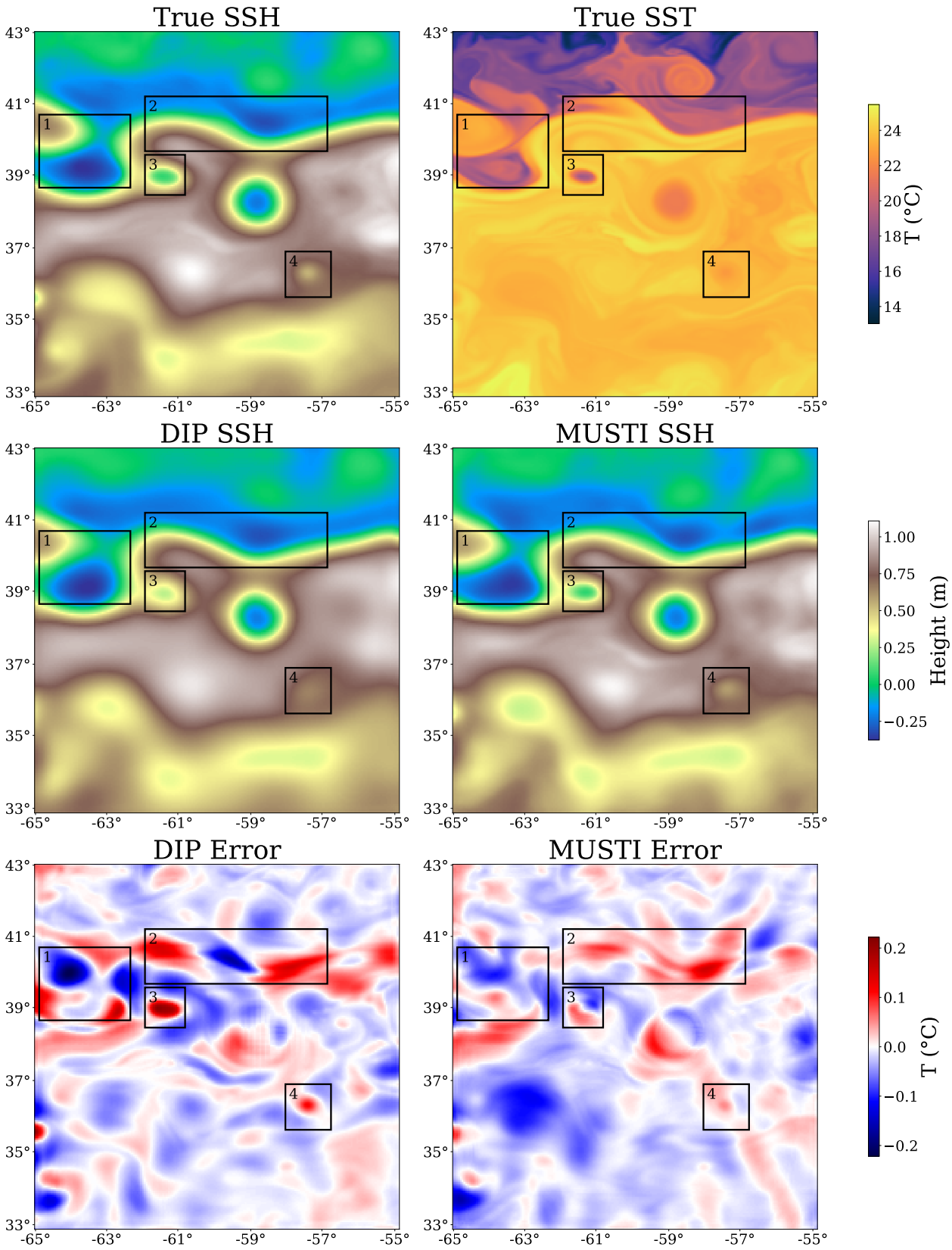


**Fig. 5.11.:** Images of an example of MUSTI and DIP interpolations of SWOT+nadir pseudo observations from ODC2020. The first line is the SSH and SST reference; the second line presents DIP and MUSTI estimations, and the last line presents their associated error maps.

years, we can train a neural network in an unsupervised setting and fuse SSH and SST information for interpolation.

## 5.4 Unsupervised learning of SSH interpolation on a new OSSE

In this section, we aim to train a neural network to perform SSH interpolation leveraging SST contextual data, and that can be applied to unseen data in a single inference. To train such a network, we require a dataset spanning several years to capture seasonal variability and the complex hidden relationship between the two variables. The impact of the dataset temporal length on neural network training is presented in Appendix **B.4**. The ODC2020 and ODC2021 datasets, although very useful for method inter-comparison, are not well suited to neural network training. As they comprise approximately one year of data, fitting a neural network on a portion of the dataset and testing it on another leads to incomplete seasonal variability and generalization issues, as shown in Appendix **B.5**. In 4DVarNet, this problem is tackled by training the neural network on wide areas (the entire North Atlantic), which provides more examples and better generalization to the test set [Fablet et al., 2021, Fablet et al., 2023]. Also, during its inference, 4DVarNet performs several gradient descent steps of its solver as in 4D-VAR, which helps to estimate new trajectories. In the following, we will tackle this issue by introducing a new training Observing System Simulation Experiment (OSSE), with 20 years of data: simulated reference and associated pseudo-observations. We will study unsupervised training methods, with or without SST, and compare it to their supervised equivalents.

### 5.4.1 Multi-variate satellite observation simulation

We present hereafter the OSSE that we designed to train neural networks. We first introduce the reference simulation upon why we simulate satellite observations, then describe how we produce SSH pseudo-observations and, finally, SST pseudo-observations. We only simulated nadir-pointing measurements because SWOT observations were not available at the time that this study was performed. Simulating only nadir-pointing satellite enables testing neural networks on real observations, which we will do in Section **5.5**.

### Base simulation: the Global Ocean Physics Reanalysis

We conduct our experiments on the Global Physical Reanalysis 12° (GLORYS12) [CMEMS, 2020]. It provides various physical data such as SSH, SST, and oceanic currents with a spatial resolution of 1/12° (around $8$ km). GLORYS12 is based on the NEMO 3.6 model [Madec et al., 2017], a global ocean circulation model, and assimilates satellite observations (SSH along-track observations and SST full domain observations) through a reduced-order Ensemble Kalman filter (see Section **3.2.2**). It is updated annually by the Copernicus European Marine Service, making it impossible to use in near real-time applications. Its long temporal extent, since 1993, is a significant advantage for creating a multi-year OSSE and learning seasonal variability. Additionally, its resolution of 1/12° is much higher than the effective resolution of altimetry products. For these reasons, it is a suitable reference dataset for modeling the multi-variate satellite observation OSSE.

The studied area is still the Gulf Stream, between 33° to 43° North and -65° to -55° East, which is the same region as in ODC2020 and ODC2021 and we select a temporal subset of this simulation from 2000-03-20 to 2019-12-29, for a total of 7194 days. Comparing the surface circulation of GLORYS12 with its geostrophic approximation, we find that an RMSE of 6.6 cm/s for $u$ and $6.1$ cm/s for $v$. Considering the high intensity and variations of the currents in the Gulf Stream (with 37.1 and 34.3 cm/s of standard deviation for $u$ and $v$, respectively), geostrophy seems to be an adequate estimation. Thus, we expect a significant synergy between SSH and SST, which a neural network can learn. For computational reasons, we resample the data to images of size $128 \times 128$ with a bilinear interpolation, corresponding to a resolution of 0.078° by pixel (approximately $8.7$ km). Doing so, the perceptive field of the network covers the entire 10° by 10° area.

### SSH simulated observations

As detailed in Section **2.2.2**, the nadir-pointing altimetry L3 product provides approximately a measurement per second along the ground tracks of the satellite. We retrieve the support of real-world satellite observations denoted $\mathbf{\Omega} = \{\Omega_i = (t_i, lat_i, lon_i)\,, i \in [0:N]\}$ from the Copernicus sea level product [CMEMS, 2021]. Using $\mathbf{\Omega}$ and the ground truth data $\mathbf{X}^{\text{ssh}}$ we produce SSH pseudo-observations $\mathbf{Y}^{\text{ssh}}$ by computing trilinear interpolation of $\mathbf{X}^{\text{ssh}}$ along each point of the support, as described in Equation **3.2**. We add an instrumental error $\varepsilon \sim \mathcal{N}(0, \sigma)$ with $\sigma = 1.9$ cm, which is the distribution used in ODC2020. The obtained SSH observation operator, $\mathcal{H}^{\text{ssh}}$ is slightly different than the masking operator used for DIP and MUSTI.

Here, the pseudo-observations are not on a fixed grid but at the exact spatiotemporal coordinates of their support $\boldsymbol{\Omega}$.

However, for the neural network input observations, we grid these data to a daily $128 \times 128$ image. We set the pixel value with no simulated satellite observation to zero, and we average the daily measurements of SSH inside each pixel to represent the mean of the daily data from the different satellites (if any). As GLORYS12 assimilates along-track SSH data, selecting satellite measurements at the same location as the assimilated data might introduce a bias in our observations. To overcome this issue, we desynchronize the real satellite ground tracks from the one we use to produce SSH observations by introducing a time delay (772 days) between the real L3 satellite observations and the simulation. It ensures that simulated along-track data is selected randomly rather than specifically where the model assimilates real-world observations. In Figure **5.12**, we plot an example of the SSH reference and their associated observations.

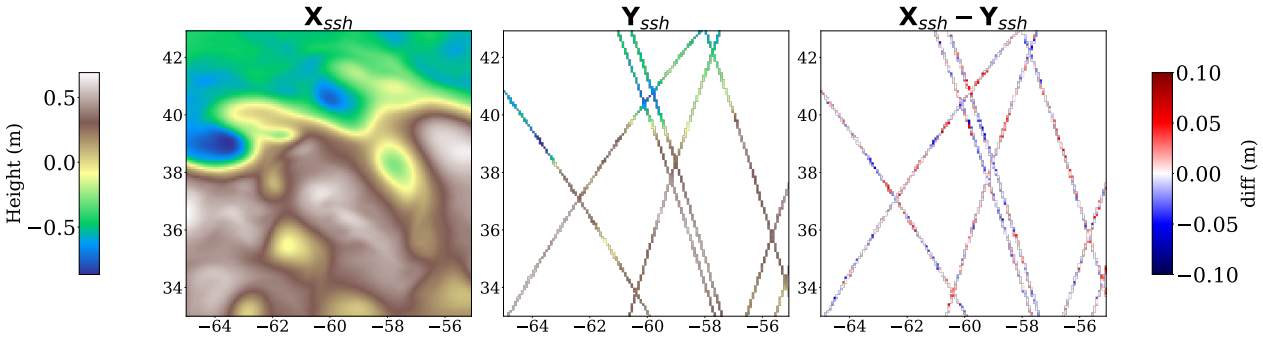


**Fig. 5.12.:** Images of the ground truth SSH from GLORYS12, the simulated along-track measurements, and the difference.

### SST simulated observations

SST remote sensing is based on direct infrared imaging, leading to wider measurement swaths but making the data sensitive to cloud cover (see Section **2.3**). To fill the gaps, the L3 products from several satellites are merged and interpolated to form the fully gridded image, using complementary microwave satellite sensors (which produce lower resolution data but are less sensitive to clouds), and *in situ* measurements [Donlon et al., 2012, Chin et al., 2017]. The product obtained from the interpolation of these data has various effective resolutions, depending on which data is available location, and high-resolution structures are artificially smoothed when the cloud cover ($C \in [0:1]$) is too thick.

We simulate the SST observation operator $\mathcal{H}^{\text{sst}}$ as follows:

$$\mathbf{Y}^{\text{sst}} = \mathcal{H}^{\text{sst}}\left(\mathbf{X}^{\text{sst}}, C\right) = (1 - C) \odot \left(\mathbf{X}^{\text{sst}} + \varepsilon\right) + C \odot \mathcal{G}_{\sigma_t, \sigma_x} \star \left(\mathbf{X}^{\text{sst}} + \varepsilon\right) \tag{5.8}$$

where $\odot$ is the element-wise product, $\star$ the convolution product, and $\varepsilon$ is a white Gaussian noise image of size $32 \times 32$ linearly upsampled to a $128 \times 128$ image. We also use a spatio-temporal Gaussian filter, $\mathcal{G}_{\sigma_{\text{t}}, \sigma_{\text{x}}}$ with $\sigma_t = 1.23$ days and $\sigma_x \approx 16$(km) to simulate the smoothing of the interpolation performed by satellite products. To compute a realistic cloud cover $C$, we use two years of data from an NRT L3 product [CMEMS, 2023c], which we periodically replicate to match the length of our dataset. We then linearly interpolate the cloud cover to our spatial resolution and perform an average filter with a kernel size approximately equal to $43$ (km). In doing so, the cloud cover is not a binary mask but a continuous value representing the ratio of observed pixels within an area. This step is essential, as applying a binary mask results in patches at the frontiers between cloud-free and cloudy regions. It is also realistic to simulate L4 SST products, as they are made through optimal interpolation of multiple sensors high-resolution infrared and low-resolution microwave. A comparison between binary and nonbinary cloud covers is presented in Appendix **B.2**. Our SST observations thus present a spatially and temporally correlated noise, with different resolutions depending on cloud cover. In the end, $\mathcal{H}^{\text{sst}}$ adds a noise with RMSE of $0.48$°C where the SST standard deviation of the ground truth is $4.96$°C, which we present in Figure **5.13**. This observation operator differs from real-world degradations but produces an image with an in-equal noise resolution similar to the errors present in the L4 SST products. Also, as SST presents strong annual variations that should be removed, we deseasonalize it. For each SST image, we subtract the mean image calculated for the corresponding day across the dataset. This is known to improve machine learning time-series prediction [Ahmed et al., 2010], and in our case, it produces better reconstructions as shown in Appendix **B.1**.

### 5.4.2 Proposed method

We chose to perform the interpolation on a temporal window of 21 days; the input is thus a tensor of 21 images of SSH, with or without SST images, and the output is the 21 corresponding days of SSH only. The neural network estimates the true state from observations, $\widehat{\mathbf{X}}^{\text{ssh}} = f_\theta\left(\mathbf{Y}\right)$, where $\mathbf{Y} = \mathbf{Y}^{\text{ssh}}$ for a SSH-only interpolation, and $\mathbf{Y} = \left(\mathbf{Y}^{\text{ssh}}, \mathbf{Y}^{\text{sst}}\right)$ if the network uses SST. The length of the time window is discussed in Section **5.4.3**, and training losses of the network in Section **5.4.2**.

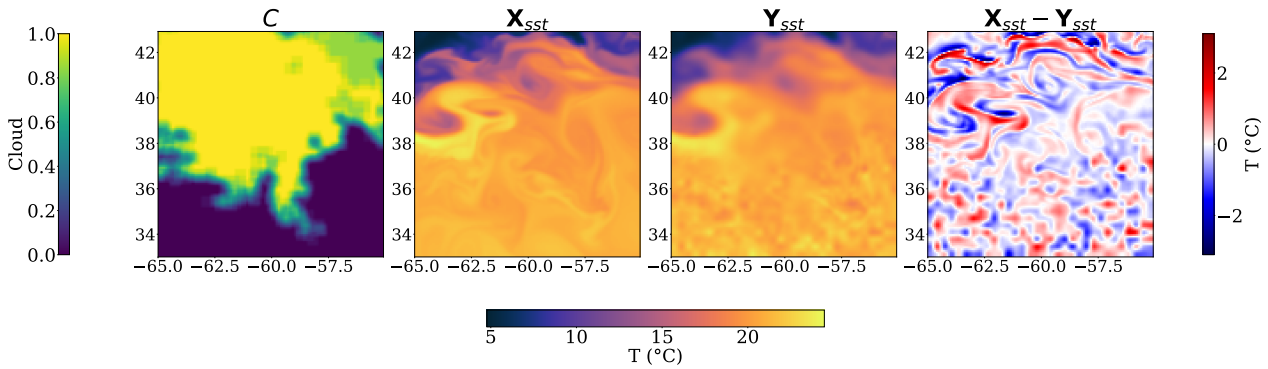


**Fig. 5.13.:** Images of our cloud cover, the ground truth SST from GLORYS12, the noised SST, and the difference.

### Attention-Based Auto Encoder

Convolutional neural networks, one of the most used deep learning methods in image tasks, learn convolution operations able to identify features over space and/or time. Over time, the machine learning community introduced various ways to organize these convolution operations, each one presenting distinct advantages (see Section **3.3.2**). Residual layers learn small modifications between their input and output, making neural networks easier to train [He et al., 2016]. Attention layers ponder their inputs by a factor between zero and one. This allows subsequent layers to focus on important features while neglecting irrelevant ones, which makes it well-suited to extracting information from contextual variables. An encoder-decoder architecture progressively compresses and decompresses the input data, identifying structures at different resolutions. In this study, we are interested in testing learning strategies using a fixed architecture. To this end, we propose an Attention-Based Encoder Decoder (ABED), presented in Figure **5.14**, to perform the interpolation over the time window. The overall structure of our neural network is inspired by [Che et al., 2022], who introduced a residual U-Net with attention layers for rain nowcasting. This neural network benefits from the layers described above, and we detail its key features in the following.

The encoder starts with batch normalization and a 3D convolution (in time and in the two spatial dimensions) followed by two downsampling blocks that divide spatial dimensions by 2 (see Figure 5.14). The decoder is composed of residual attention blocks followed by upsampling blocks.

Our attention block consists of two essential steps: a temporal attention and then a spatial attention. Our approach builds upon the Convolutional Block Attention Module (CBAM) principle introduced by [Woo et al., 2018] (see Section **3.3.2**),

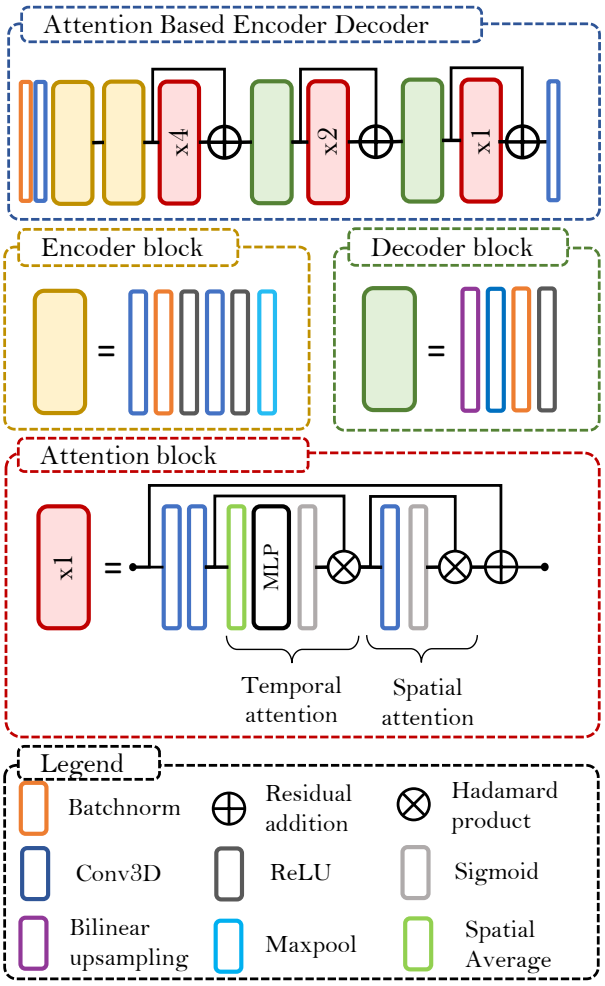


**Fig. 5.14.:** Architecture of the proposed Attention-Based Encoder Decoder (ABED) neural network. It is designed to take a time series of 21 images of SSH, with or without a time series of SST. The encoder divides the spatial dimensions of the images by 4 through 2 "down-block". Then, the decoder uses an attention block to highlight relevant information in the images and progressively upscales it.

which successively performs channel and spatial attention. We extend this idea by incorporating temporal information in the channel attention mechanism. To do so, we first compute the spatial average of each channel and instant, resulting in a tensor of size $C \times T$ where $C$ is the channel number and $T$ is the time series length. Subsequently, we apply two one-dimensional convolutional layers with a kernel of size 1, followed by a sigmoid activation function to estimate the attention weights. This corresponds to a 2-layer perceptron shared by every time step, which differs from the CBAM, as it includes the temporal information in the channel attention. These weights are then multiplied to each timestep of every channel, enabling the network to highlight salient features and suppress irrelevant information. After performing temporal attention, we proceed with spatial attention. This step involves

utilizing a 3-dimensional convolutional operation, where the kernel size's temporal length matches the time series's length. As a result, the entire time series is aggregated into a single 2D image, which serves as the basis for deriving spatial attention. A residual skip connection is then applied, and the described block is repeated 4, 2, and 1 times for the first, second, and last block, respectively. For further details about our implementation, we provide the PyTorch implementation of our network in `https://gitlab.lip6.fr/archambault/james2024`.

### Losses

We propose to compare two main strategies to train the neural network. Thanks to the OSSE previously described, we have access to the ground truth, which we can use to learn the interpolation in a classic supervised fashion. However, it is also possible to train directly on observations by applying the $\mathcal{H}^{\text{ssh}}$ on the generated map $\widehat{\mathbf{X}}^{\text{ssh}}$ before computing the loss (see Equations **5.9**, **5.10**, **5.11**). In Section **5.2** we performed the interpolation with SSH observations only, and, using the same principle, in Section **5.3** we showed that it was possible to estimate SSH images starting from SST only and constraining on SSH observations. Both these methods are fitted on one (or a small number) example and must be refitted to be applied to unseen data. Using a larger real-world satellite dataset, [Martin et al., 2023] trained a neural network directly from observations by constraining it on independent satellite observations that were not given in the input. However, the lack of ground truth reference makes it harder to compare the different reconstructions, especially regarding detected eddies and structures. We propose to train neural networks using the 3 following losses:

- The MSE using ground truth (supervised training):

$$\mathcal{L}_{\text{sup}}(\mathbf{X}^{\text{ssh}}, \widehat{\mathbf{X}}^{\text{ssh}}) = \text{MSE}\left(\mathbf{X}^{\text{ssh}}, \widehat{\mathbf{X}}^{\text{ssh}}\right) \tag{5.9}$$

- The MSE using only observations (unsupervised training):

$$\mathcal{L}_{\text{unsup}}(\mathbf{Y}^{\text{ssh}}, \widehat{\mathbf{X}}^{\text{ssh}}) = \text{MSE}\left(\mathbf{X}^{\text{ssh}}, \mathcal{H}^{\text{ssh}}(\widehat{\mathbf{X}}^{\text{ssh}})\right) \tag{5.10}$$

- The MSE using only observations and the regularization introduced by [Martin et al., 2023] (unsupervised training):

$$\mathcal{L}_{\text{unsup\_reg}}(\mathbf{Y}^{\text{ssh}}, \widehat{\mathbf{X}}^{\text{ssh}}) = \mathcal{L}_{\text{unsup}}(\mathbf{Y}^{\text{ssh}}, \widehat{\mathbf{X}}^{\text{ssh}}) + \lambda_1 \text{MSE}\left(\frac{\partial}{\partial s}\mathbf{Y}^{\text{ssh}}, \frac{\partial}{\partial s}\mathcal{H}^{\text{ssh}}(\widehat{\mathbf{X}}^{\text{ssh}})\right) + \lambda_2 \text{MSE}\left(\frac{\partial^2}{\partial^2 s}\mathbf{Y}^{\text{ssh}}, \frac{\partial^2}{\partial^2 s}\mathcal{H}^{\text{ssh}}(\widehat{\mathbf{X}}^{\text{ssh}})\right) \tag{5.11}$$

where $\frac{\partial}{\partial s}$ is the along-track derivation of the SSH approximated by its rate of change (see Appendix **B.6**). We take $\lambda_1 = \lambda_2 = 0.05$ the regularization coefficients, the same values used by [Martin et al., 2023].

Losses $\mathcal{L}_{\text{unsup}}$ and $\mathcal{L}_{\text{unsup\_reg}}$ apply the observation operator $\mathcal{H}^{\text{ssh}}$, before computing the MSE, which allows training in a framework where only observations are available. Thus, from an interpolation point of view, the inversion methods that use these losses are unsupervised as they can be trained without any ground truth image. However, if we constrain the network on the same observations that were given in input, an over-fitting of along tracks will occur as the neural network can simply copy its input on its output. This is shown in Appendix **B.3**, where we see that constraining the network on its input data leads to very bad reconstruction scores. To avoid this problem, we must find a way to select a subset of SSH observations that will be given in input and some that will only serve to control the reconstruction. In the following, we consider two ways to perform this subset:

- First, we can train the network on the observations of one satellite that were withdrawn from the input as described by [Martin et al., 2023]. Similarly, we remove the data of one satellite from the inputs, but we calculate the loss function on all satellite observations (the ones given and the ones left aside). In doing so, the network must generalize outside the along-track measurement that was given as input. We call *1-sat* this subsetting method.

- Second, we can select patches in the images where we remove the data from all the satellites. We use three patches by timestep, each of a size $32 \times 32$ pixels, placed randomly in the image at different locations between days. This method has two main differences from the previous one: first, as new masks are drawn for each training epoch, it leads to data augmentation, where each example is slightly different from the previous one. Also, with this method, there are entire areas with no SSH input data, whereas in the previous method, the measurements from another satellite might have been present. We will show in the results that this method outperforms the one introduced in [Martin et al., 2023]. We call *mask* this subsetting method.

We call $\mathbf{Y}_{\text{in}}^{\text{ssh}}$ the subset of observations passed to the neural network inputs. In Figure **5.15**, we present a graphical overview of the supervised training method (top), and of the unsupervised training (bottom).

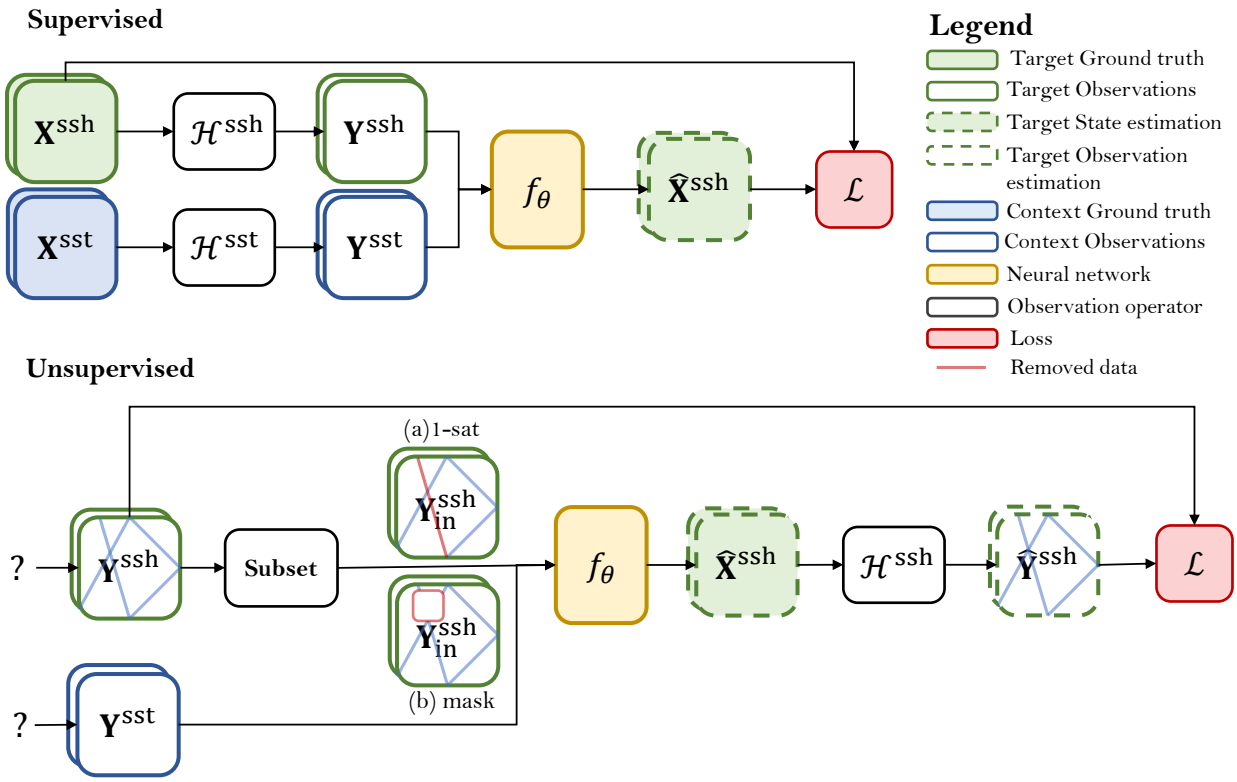


**Fig. 5.15.:** Graphical overview of the supervised and unsupervised interpolation method. The neural network input is a 21-day time series of SSH satellite observations, excluding data from a single satellite, and optionally includes SST measurements. The network estimates a time series of SSH field states, upon which the observation operator is subsequently applied in order to deduce $\widehat{\mathbf{Y}}^{\text{ssh}}$. Finally, the Mean Squared Error between the $\widehat{\mathbf{Y}}^{\text{ssh}}$ and $\mathbf{Y}^{\text{ssh}}$ is used to control the network.

### Training details

**Train, validation, test split.** We partitioned the OSSE dataset into three subsets: training, validation, and test data. We used the year 2017 exclusively to test our reconstructions (every analysis conducted in the following was performed on this data). We validate our methods on three distinct time intervals: (1) from 2002/07/14 to 2003/07/28, (2) from 2008/01/05 to 2009/01/18, and (3) from 2013/06/28 to 2014/07/13. The remaining data was used for training, but leaving a 15-day period to prevent data leakage.

**Normalization.** We normalize the artificial network's input and output by subtracting the mean and dividing by the standard deviation. The normalization parameters are computed only on the neural network inputs, SST, or along-track data. Specifically, we first perform this normalization for images related to SSH along-track measurements and subsequently replace any missing values with zeros. We normalize the neural network SSH outputs with the statistics computed on the input observations (so that the method remains applicable in an unsupervised setting). When training with the regularized loss of Equation **5.11**, we also normalize the data from the first and second SSH along-tracks derivative.

**Training hyperparameters.** We train every method using an ADAM optimizer [Kingma and Ba, 2014] with a learning rate starting at $5.10^{-5}$ and a decay of $0.99$. We perform an early stopping with a patience of 8 epochs. For the supervised training, the stopping criteria is the RMSE of the reconstruction on the fully gridded domain on the validation data, but in the unsupervised setting, we compute this RMSE on left-aside along-track measurements. Doing so, the stopping strategy is still compliant with a situation where no ground truth is accessible.

**Ensemble.** As neural network optimization is sensitive to its weight initialization, we train 3 networks for every setting. The so-called "Ensemble" estimation is the average SSH map of the 3 networks.

### 5.4.3 Results: comparison of the reconstructed maps

In Sections **5.4.3** and **5.4.4**, we compare the different training methods on our OSSE to highlight the drawbacks of unsupervised learning and the advantages of SST. In Section **5.4.3**, we first focus on SSH reconstruction and the associated geostrophic current estimation. Then, in Section **5.4.4**, we compare the mesoscales eddies of the different maps, to provide a physical interpretation of the reconstruction errors.

#### SSH reconstruction and quality of derived geostrophic currents

We compare the fields estimated by the networks trained using the three losses $\mathcal{L}_{\text{sup}}$, $\mathcal{L}_{\text{unsup}}$ and $\mathcal{L}_{\text{unsup\_reg}}$, with 3 different sets of input data: only SSH tracks, SSH and the noised SST from our OSSE (denoted nSST), and noise-free SST (denoted SST). The noise-free SST provides an upper-bound performance of the neural network in the case of a perfect physical link between SSH and SST. For the unsupervised loss functions $\mathcal{L}_{\text{unsup}}$ and $\mathcal{L}_{\text{unsup\_reg}}$, we test the two ways to remove some SSH

information from ABED inputs: the one where the data from one satellite is removed, hereby called *1-sat* subset, and the one where a random mask is applied, called *mask*. We give the RMSE of the SSH estimates fields on the test set in Table **5.6** and the RMSE on the velocity fields in Table **5.7**.

| Loss | subset | SSH | SSH+nSST | SSH+SST |
|---|---|---|---|---|
| $\mathcal{L}_{\text{sup}}$ | – | 4.16 \| 3.81 | 3.34 \| 3.03 | **2.97** \| **2.63** |
| $\mathcal{L}_{\text{unsup}}$ | *1-sat* | 4.56 \| 4.20 | 3.84 \| 3.49 | 3.56 \| 3.16 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *1-sat* | 4.33 \| 4.07 | 3.76 \| 3.52 | 3.48 \| 3.20 |
| $\mathcal{L}_{\text{unsup}}$ | *mask* | 3.89 \| 3.66 | 3.43 \| 3.19 | 3.18 \| 2.92 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *mask* | 3.83 \| 3.67 | 3.54 \| 3.35 | 3.22 \| 3.04 |

**Tab. 5.6.:** SSH reconstruction RMSE in centimeters (mean score on the left and ensemble score on the right) of 3 ABED networks. The interpolation is trained using the three different losses described in Section **5.4.2** with the following settings: SSH-only interpolation, SSH and noised SST, and SSH and noise-free SST. For the unsupervised inversions, we include the two types of subsetting: *1-sat* and *mask*. All metrics are given on the central image of a 21-day time window.

First, the ensemble reconstruction has a lower RMSE than the mean performance, which is usual in machine learning, as individual member errors are compensated by others. Comparing the ensemble scores, we observe that the supervised loss function outperforms the unsupervised framework in every data scenario. Specifically, in the SSH+SST scenario, the supervised loss decreases the ensemble RMSE of $\mathcal{L}_{\text{unsup}}$ by 17%, and 9% without SST, in the *1-sat* subset. Also, adding SST as an additional input to the network generally improves performance compared to using SSH alone. This improvement is observed across all three loss functions, as the error values decrease for SSH+nSST compared to SSH. For instance, the SSH-only ensemble RMSE is decreased by 31% and 20% for SST and nSST, respectively, with $\mathcal{L}_{\text{sup}}$.

Comparing the unsupervised inversions, we see that the regularization introduced by [Martin et al., 2023] slightly improves reconstruction in the *1-sat* subset but slightly degrades it in the *mask* subset. This last sub-setting method leads to a significant performance increase, especially when using only SSH, where it even outperforms the supervised training. This improvement is important compared to [Martin et al., 2023, Archambault et al., 2024b]. In the following, we mostly interpret the *1-sat* subsetting to be coherent with our articles [Archambault et al., 2024b, Archambault et al., 2024a].

We estimate the surface currents from the reconstructed SSH geostrophy, and we compare it to the surface circulation of the model. The errors on velocity in Table **5.7** follow the same trends as the RMSE on the SSH fields but with lesser differences

| Loss | subset | SSH | | SSH+nSST | | SSH+SST | |
|---|---|---|---|---|---|---|---|
| | $u$ | $v$ | $u$ | $v$ | $u$ | $v$ | |
| $\mathcal{L}_{\text{sup}}$ | – | 12.8 | 13.9 | 11.1 | 12.0 | **10.1** | **10.7** |
| $\mathcal{L}_{\text{unsup}}$ | *1-sat* | 13.4 | 15.5 | 12.0 | 14.1 | 11.1 | 13.1 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *1-sat* | 12.8 | 14.3 | 11.7 | 12.9 | 11.0 | 12.0 |
| $\mathcal{L}_{\text{unsup}}$ | *mask* | 12.4 | 15.0 | 11.3 | 13.8 | 10.6 | 12.8 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *mask* | 12.1 | 13.5 | 11.5 | 12.7 | 10.8 | 11.7 |

**Tab. 5.7.:** Eastward ($u$) and Northward ($v$) surface currents in cm/s. The currents were estimated by applying the geostrophic approximation (see Equation **2.3**) on the SSH ensemble estimation of the 3 ABED networks.

between methods. The RMSE is still far from the minimal error achievable through geostrophy, which is $6.57$ cm/s for $u$ and $6.14$ for $v$ on this data.

### Seasonal errors and temporal window size

In Figure **5.16**, we show the daily errors of the different methods on the test year. For more clarity, we only plot the unsupervised inversion using the *1-sat* subsetting. We notice a substantial temporal variability of the RMSE, with a notable increase in late summer. Specifically, in August and September, all methods perform worse than in Winter, which can be explained by the high kinetic energy of the ocean in Summer [Zhai et al., 2008, Kang et al., 2016].

An important challenge of ocean satellite products is to provide real-time estimations, as many applications cannot use products available with too much time delay. In an operational framework, products that are immediately available are called Near Real Time (NRT) whereas those that require a time delay before release are called Delayed Time (DT). While in Tables **5.6** and **5.7**, we presented the results obtained on the central image of the time window, we can also display their scores along the 21-day temporal window as in Figure **5.17**. The central image is a 10-day Delayed Time reconstruction as we need images of observations 10 days in the future. In Figure **5.17**, we verify that 21 days of data contain enough information to reconstruct the central image: for instance, 5 days from the border of the temporal window the reconstruction error is just 3% higher than the one at the center. This means that we can significantly reduce the delay (and therefore the training cost of our model) without causing severe drops in performance, which could be useful if applied in an operational framework. However, when it comes to producing NRT products (0 delay) this graph shows that we expect a significant loss of quality in the reconstruction which is usual [Amores et al., 2018, Stegner et al., 2021].

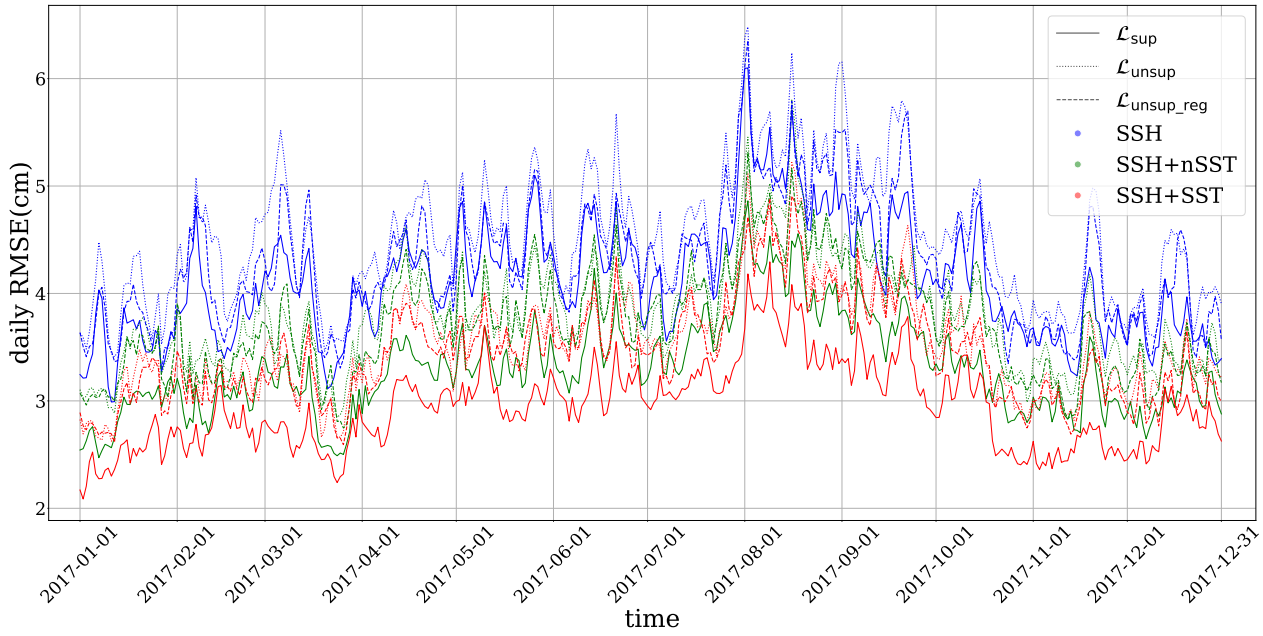


**Fig. 5.16.:** RMSE of the different reconstructions during the test year (2017). The unsupervised methods are using the *1-sat* subsetting.

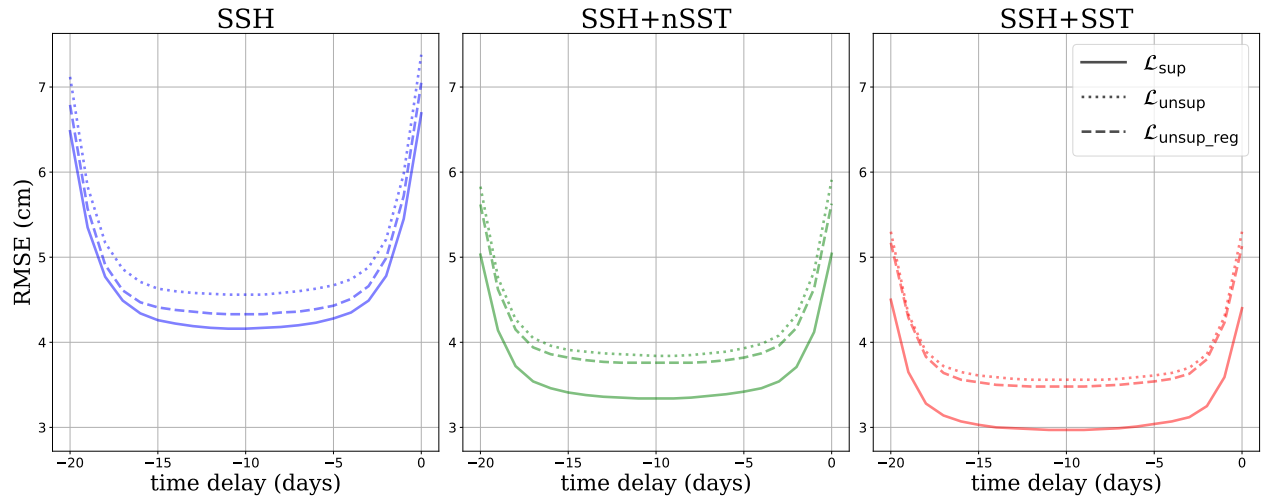


**Fig. 5.17.:** RMSE of the different reconstructions along the time window as a function of the time delay to present. The errors at a time delay of $-20$ correspond to an anti-causal scheme (knowing only future observations), whereas $timedelay = 0$ corresponds to a causal scheme (knowing no future observations). Knowing both past and future observations leads to the optimal reconstruction at $timedelay = -10$. The unsupervised methods are using the *1-sat* subsetting.

### 5.4.4 Results: mesoscale eddy analysis

#### Importance of mesoscale eddies

Mesoscale eddies play an important role in ocean circulation and dynamics, and their understanding leads to diverse applications in oceanography or navigation [Chelton et al., 2011b]. Previous studies underline how these structures transport heat, especially between latitudes 0° and 40° in the North Atlantic [Jayne and Marotzke, 2002], but also salinity [Amores et al., 2017], or plankton [Chelton et al., 2011a]. In practice, mesoscale eddies and structures are estimated through geostrophic currents derived from satellite altimetry. However, operational satellite products such as DUACS OI, have too coarse resolutions to resolve accurately mesoscale structures. Performing an OSSE to simulate the satellite's remote sensing, [Amores et al., 2018, Stegner et al., 2021] showed that DUACS-like optimal interpolation aggregates small eddies into larger ones (i.e. with a radius greater than 100 km). These interpolations also capture a small percentage of eddies in the model simulation (around 6% in the North Atlantic) and change the eddies' distribution and properties. This is why we are interested in finding to what extent our reconstruction methods can detect small eddies in the ground truth, and how well the detected eddies are resolved and their physical properties conserved.

#### Automatic eddy detection algorithm: AMEDA

We use the Angular Momentum for Eddy Detection and tracking Algorithm (AMEDA) introduced by [Vu et al., 2018] to perform the eddies detection. It is based on the LNAM, a metric first introduced by [Mkhinini et al., 2014], that we define hereafter:

$$\mathrm{LNAM}(P_i) = \frac{\sum_j \overrightarrow{P_iP_j} \times \overrightarrow{V_j}}{\sum_j \overrightarrow{P_iP_j} \cdot \overrightarrow{V_j} + \sum_j |\overrightarrow{P_iP_j}||\overrightarrow{V_j}|} = \frac{L_i}{S_i + BL_i} \tag{5.12}$$

where $P_i$ is the point of the grid where we compute the LNAM, $P_j$ is a neighbor point of the grid, $\overrightarrow{P_iP_j}$ is the position vector from $P_i$ to $P_j$, $\overrightarrow{V_j}$ is the velocity vector in $P_j$, $\times$ the cross product, and $\cdot$ the dot product. Thus, the unnormalized angular momentum $L_i$ is computed through a sum of cross products and is bounded by $BL_i$, so that if $P_i$ is the center of an axisymmetric cyclone (resp anticyclone), $\mathrm{LNAM}(P_i)$ will be equal to 1 (resp -1). Also, if the circulation field is hyperbolic and not an

ellipsoid, $S_i$ will reach large values, and $\mathrm{LNAM}(P_i)$ will be close to 0. All sum is computed on a local neighborhood of $P_i$, which is a hyperparameter of the method (typically a square centered on $P_i$). In our case, we used the default parameters where the square has a length of $2\Delta x$, with $\Delta x$ being the grid resolution ($\approx 9$ km).

AMEDA finds potential eddy centers by searching for the local extrema of the $\mathrm{LNAM}$ field, more precisely by taking the points $P_i$ where $|\mathrm{LNAM}(P_i)| > 0.7$. The characteristic contour of an eddy is then defined as the closed streamline of maximum velocity which does not include another eddy center. We perform the AMEDA algorithm on the geostrophic velocity field of our estimation and on the ground truth currents. We then look for the eddies that are both present in the ground truth and in our estimation. An eddy is said to be detected if the distance between its barycenter and the reference one is smaller than the average of the mean radius of the two characteristic contours. This definition allows “multiple" detection (i.e., colocalization with several eddies). Therefore, we exclude eddies that include more than one candidate in the ground truth. For further details about the AMEDA algorithm, we refer the reader to [Vu et al., 2018].

### Eddy detection performances

We present the detection scores of the different reconstruction methods, with three data scenarios and three losses. We present the two sunsetting methods for the unsupervised loss functions, *1-sat* and *mask*. We take the ensemble SSH estimation of the neural networks and perform the AMEDA algorithm on the velocity field derived through the geostrophic approximation.

In Table **5.8** we present the $F_1$ score, the recall, and the precision of the methods. The recall tells us the proportion of actual positive instances that were correctly identified by the detection (a recall of $1$ means that all ground truth eddies were detected). The precision gauges our trust in the detected eddies (a precision of $1$ means that all eddies in the simulation were also present in the ground truth). To aggregate recall and precision, we use the $F_1$ score, which is the harmonic mean of recall and precision. A value of $1$ means a perfect detection: all ground truth eddies were detected, and the estimation produced no false positives.

**Data comparison.** As expected, no matter which loss we consider, the noise-free temperature detection method outperforms the two other scenarios with higher $F_1$ scores. Even the noisy SST provides important information for eddy reconstruction, as the SSH-only method yields lower results than the two other scenarios. We also

| Loss | subset | SSH | | | SSH+nSST | | | SSH+SST | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $F_1$ | recall | precision | $F_1$ | recall | precision | $F_1$ | recall | precision |
| $\mathcal{L}_{\text{sup}}$ | – | 0.699 | 0.607 | 0.825 | 0.735 | 0.657 | 0.833 | **0.762** | 0.705 | 0.83 |
| $\mathcal{L}_{\text{unsup}}$ | *1-sat* | 0.692 | 0.634 | 0.76 | 0.715 | 0.665 | 0.772 | 0.731 | 0.694 | 0.771 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *1-sat* | 0.684 | 0.581 | 0.831 | 0.704 | 0.608 | 0.835 | 0.713 | 0.622 | 0.836 |
| $\mathcal{L}_{\text{unsup}}$ | *mask* | 0.715 | 0.669 | 0.768 | 0.725 | 0.697 | 0.756 | 0.735 | **0.714** | 0.758 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *mask* | 0.699 | 0.6 | 0.838 | 0.709 | 0.612 | 0.842 | 0.725 | 0.631 | **0.851** |

**Tab. 5.8.:** Scores of the AMEDA eddy detection performed on the Ensemble estimation of ABED interpolation. The considered scores are the precision, the recall, and the $F_1$ score.

see that for each loss, the precision scores are less impacted by the input data than the recall is. This means that the SSH-only scenario does not produce a lot more false detection than the SST methods but misses many more structures.

**Loss comparison.** On the other hand, the loss function used to perform the inversion substantially impacts precision and recall. The regularization of the unsupervised loss brings the detection precision to the level of the supervised method (even higher for the SSH-only and SSH+SST) but also reduces the recall of all methods compared to their unregularized version. In other words, the regularization prevents the neural network from generating false eddies and from retrieving some structures, which leads to lower $F_1$ scores. Similarly to SSH reconstruction metrics, using *mask* subsetting, we enhance the detection scores of the methods in every data scenario.

**Visual comparison.** We plot in Figure **5.18** the SSH maps and eddies detected by AMEDA, and in Figure **5.19** their associated relative vorticity (see Equation **4.4**). These figures illustrate an example of the conclusions established in Table **5.8**. the SSH-only reconstruction shows fewer eddies than the ones using SST and aggregates small eddies into larger ones (see highlighted eddies). We also see the effect of regularization, especially in the relative vorticity fields, which are much smoother than the ones in the supervised and regularized inversion. This smoothing effect results in a reduced number of detected eddies, as illustrated by the two highlighted eddies that are detected separately when SST is used without regularization.

### Physical properties of detected eddies

To further investigate the performance of the eddy detection methods, we analyze the detection outcomes based on the physical characteristics of the eddies. For instance, smaller eddies tend to have shorter lifespans, making them more challenging to detect due to their decreased likelihood of being observed by satellites. Conversely,

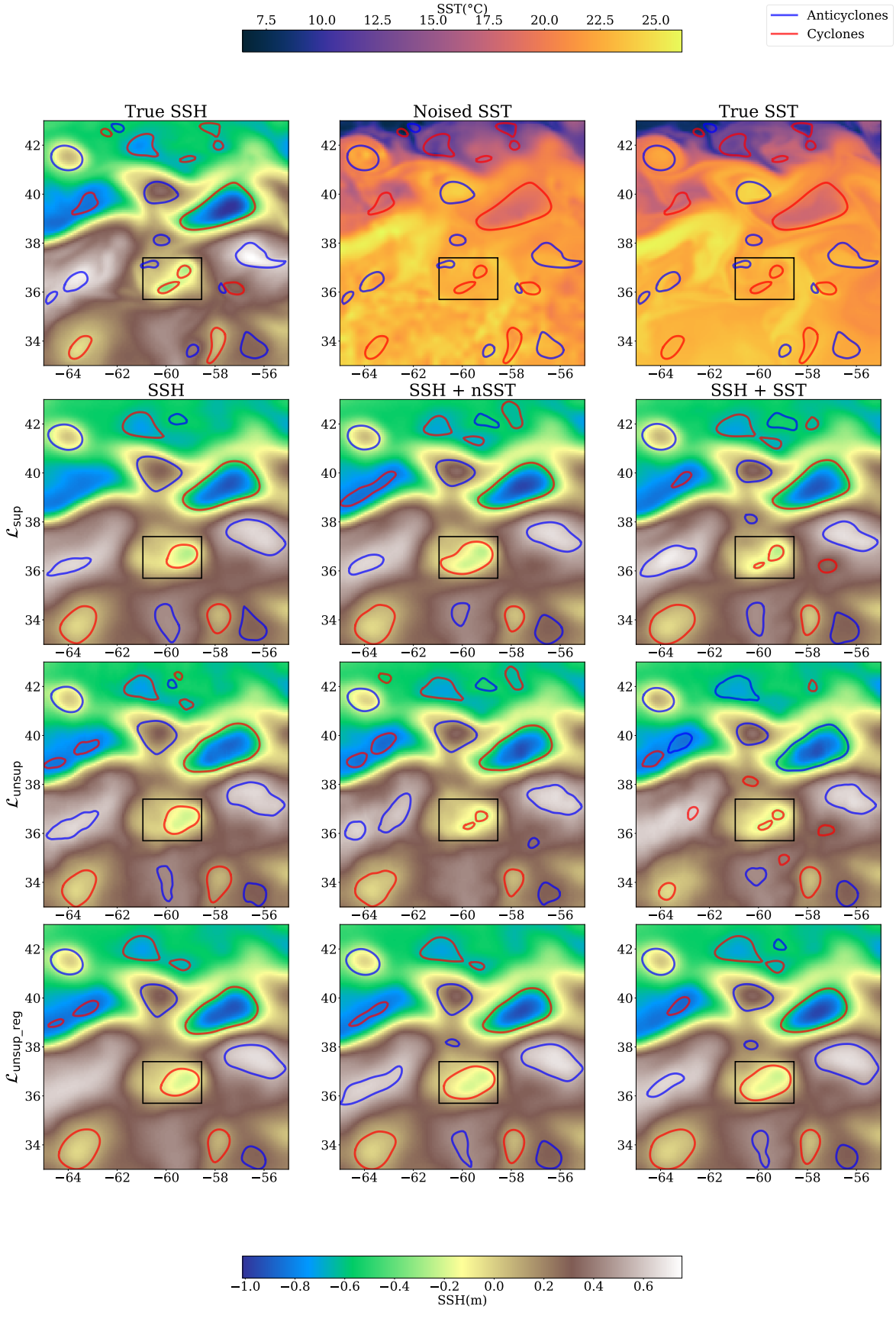


**Fig. 5.18.:** SSH maps and detected eddies on $1^{st}$ June 2017 from our OSSE. The first line presents the True SSH, the noised SST, and the True SST, on which we plot the eddies detected on the True SSH. The second, third, and last lines present, respectively, the inversion using $\mathcal{L}_{\text{sup}}$, $\mathcal{L}_{\text{unsup}}$, and $\mathcal{L}_{\text{unsup\_reg}}$. The first, second, and last columns present the maps using the SSH-only, SSH+nSST, and SSH+SST data, respectively. Each SSH map is the ensemble reconstruction of 3 networks with their associated eddies.

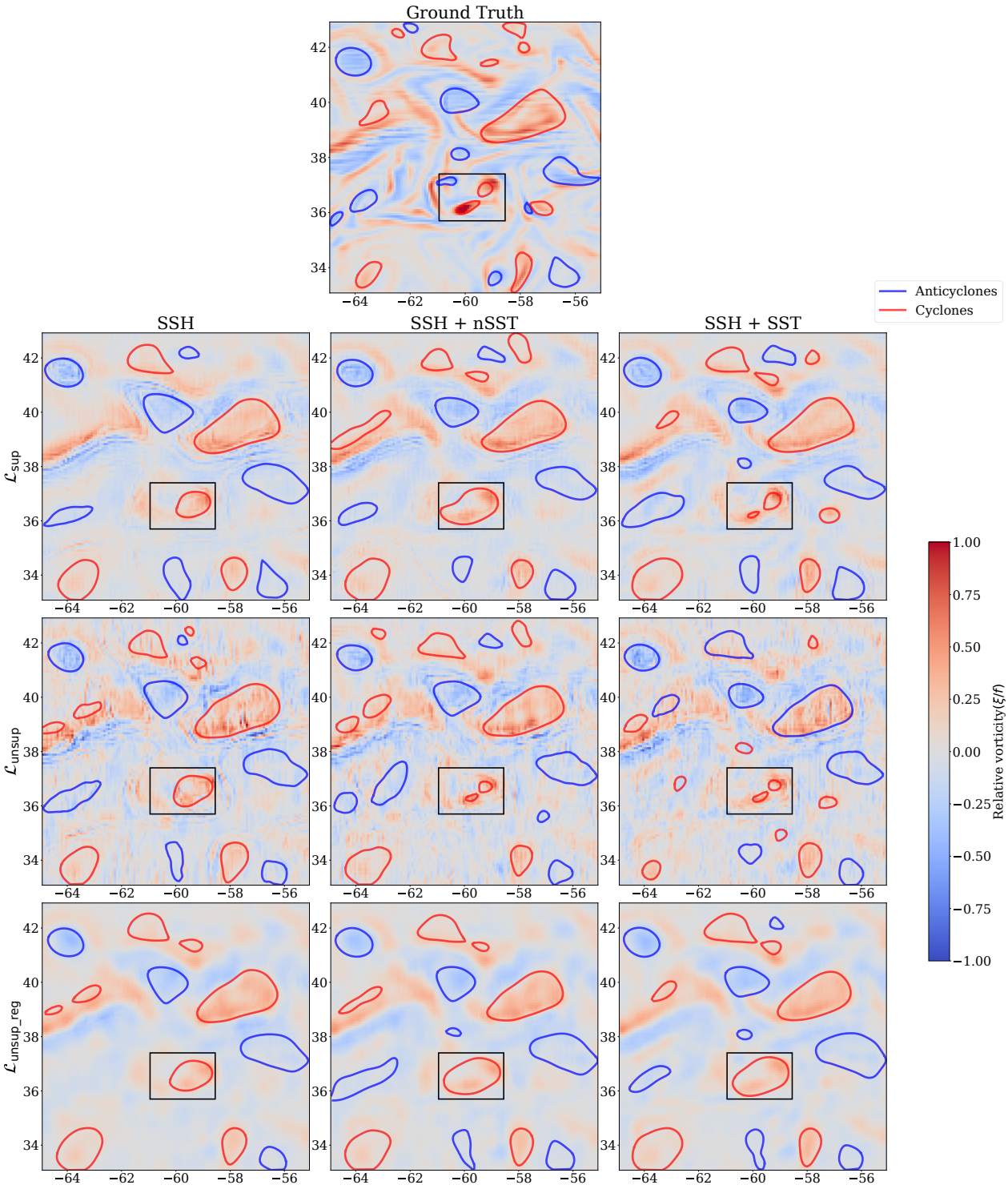


**Fig. 5.19.:** Relative vorticity (normalized by the Coriolis factor) and detected eddies on $1^{st}$ June 2017 from our OSSE. The first line presents the true relative vorticity. The second, third, and last lines present the neural networks trained with $\mathcal{L}_{\text{sup}}$, $\mathcal{L}_{\text{unsup}}$, and $\mathcal{L}_{\text{unsup\_reg}}$. The first, second, and last columns present the SSH-only, SSH+nSST, and SSH+SST interpolations. Each relative vorticity map is computed from the ensemble SSH estimation of the 3 networks.

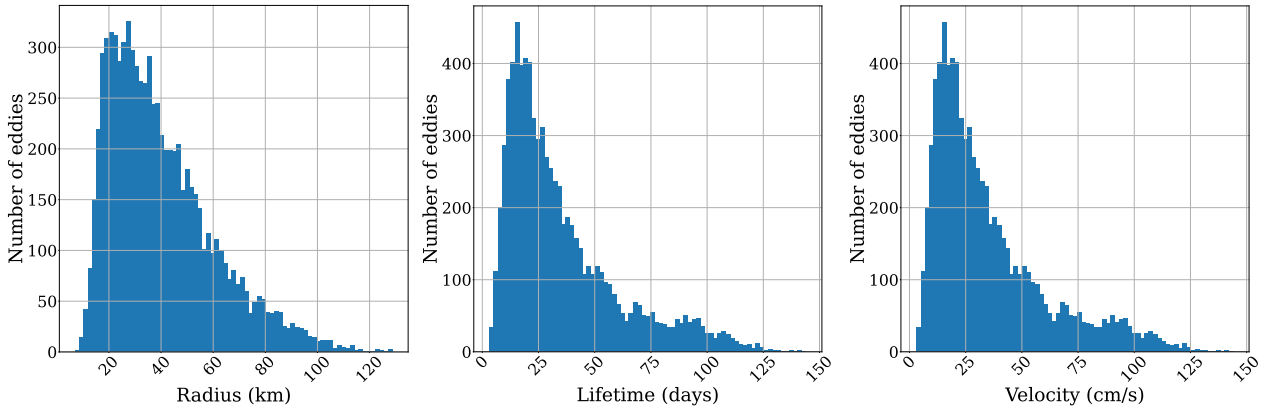


**Fig. 5.20.:** Histograms of the eddies detected by AMEDA on the SSH ground truth, by radius, lifetime, and maximum velocity.

high-speed eddies are derived from important sea surface height (SSH) variations, thus exhibiting a strong signature in the generated mapping. In Figure **5.20**, we show the histograms of the eddies detected by AMEDA on the SSH ground truth, as a function of maximum radius, lifetime, or maximum velocity along the final closed current line. Figure **5.21** shows the detection performances as a function of these parameters.

As anticipated, using SST and nSST data contributes to the detection of eddies, as indicated by the higher $F_1$ scores achieved in every loss scenario. However, small and short-lived eddies are less frequently detected, resulting in lower recall scores. Specifically, only 17% of the eddies with a radius below 15 km are successfully detected in the best scenario. Nonetheless, except for the unregularized loss function, the precision scores for the detected eddies remain high, even for small and short-lived ones. This observation confirms the previously observed phenomenon where the regularization employed in the inversion process prevents the network from generating false eddy detections but also stops it from capturing a significant portion of the actual eddies. This regularization behavior is expected, as forcing a smoothness constraint on the SSH gradient field leads to denying some of the small structures.

We also want to assess the model's accuracy to estimate the eddies' physical properties. To this end, we focus on the eddies that were successfully detected by all the methods (3534 eddies out of the 7908 eddies in the ground truth) and compare the physical parameters of the estimated eddies to their values in the corresponding true eddy. We compute the RMSE and bias of the following parameters: maximum radius and velocity of the characteristic contour of the eddies. Once again, Tables **5.9** and **5.10** show that SST helps to estimate eddies radius, and velocity. Nonetheless, there

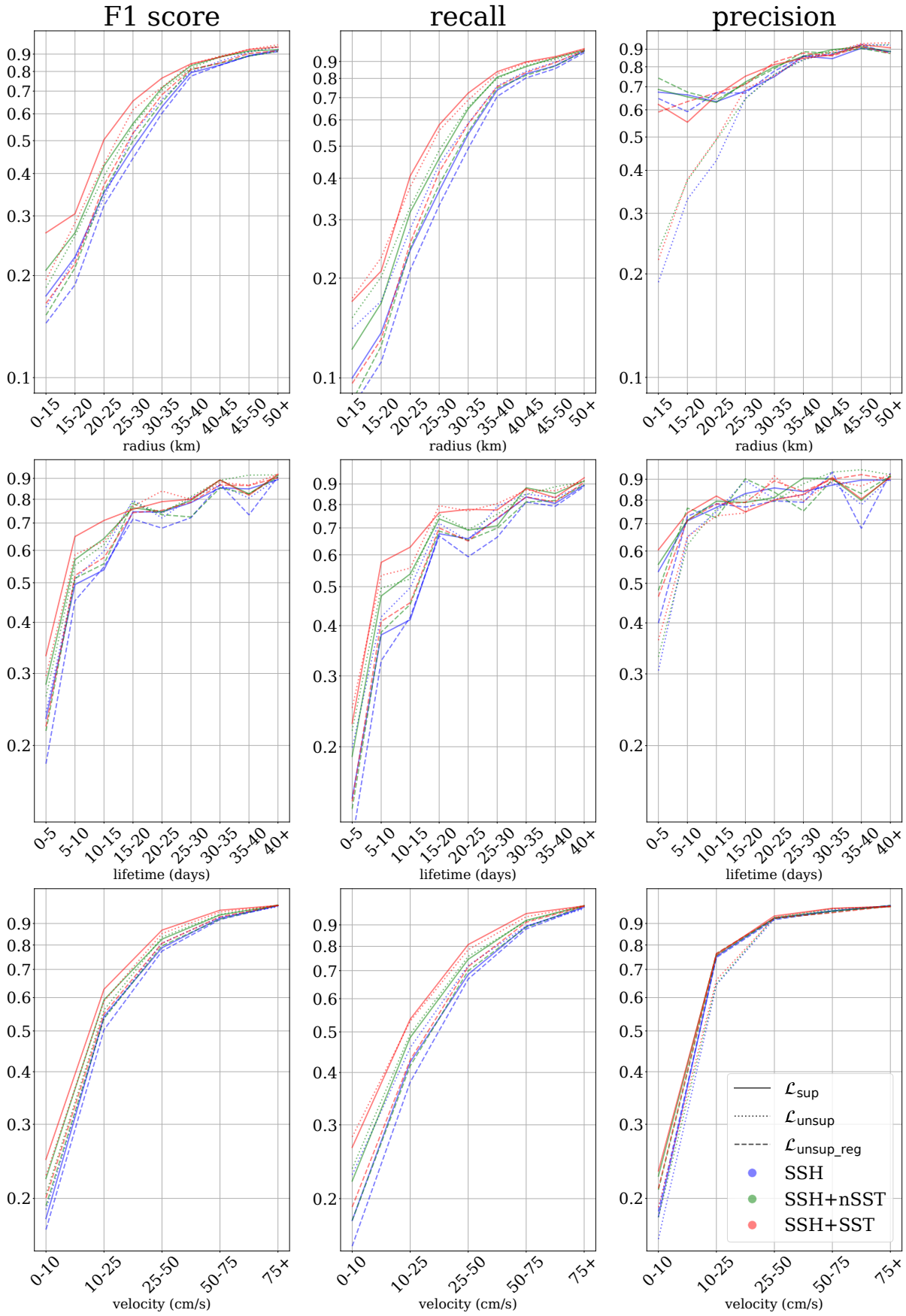


**Fig. 5.21.:** Detection scores of the different methods on eddies separated by radius (first row), lifetime (second row), and maximum velocity (last row). The considered scores are $F_1$ (first column), recall (second column), and precision (third column). The recall tells the proportion of actual positive instances that were correctly identified, the precision gauges the trust that we can put in the detected eddies, and the $F_1$ score aggregates these two values.

is a bias of radius and velocity: the size of the eddy is statistically overestimated compared to its ground truth, while its speed is systematically underestimated. This is particularly true for the regularized unsupervised loss because of its smoothness constraint, with a velocity bias accountable for half of the RMSE.

| Loss | subset | SSH | | SSH+nSST | | SSH+SST | |
|---|---|---|---|---|---|---|---|
| | | RMSE | bias | RMSE | bias | RMSE | bias |
| $\mathcal{L}_{\text{sup}}$ | – | 16.7 | 3.6 | 15.7 | 4.2 | **14.7** | 3.7 |
| $\mathcal{L}_{\text{unsup}}$ | *1-sat* | 16.6 | **0.9** | 16.3 | 1.3 | 15.5 | 1.3 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *1-sat* | 16.6 | 3.5 | 16.5 | 4.1 | 15.7 | 4.5 |
| $\mathcal{L}_{\text{unsup}}$ | *mask* | 16.1 | 1.0 | 15.9 | 1.0 | 15.1 | 1.3 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *mask* | 16.3 | 4.3 | 16.0 | 4.2 | 15.5 | 4.4 |

**Tab. 5.9.:** Eddies maximum radius RMSE and bias (km). The eddy detection is performed on geostrophic currents of the ensemble estimation and the bias is computed from the estimated radius minus ground truth radius.

| Loss | subset | SSH | | SSH+nSST | | SSH+SST | |
|---|---|---|---|---|---|---|---|
| | | RMSE | bias | RMSE | bias | RMSE | bias |
| $\mathcal{L}_{\text{sup}}$ | – | 14.3 | -5.3 | 12.4 | -3.1 | **11.7** | **-2.0** |
| $\mathcal{L}_{\text{unsup}}$ | *1-sat* | 14.4 | -5.7 | 13.4 | -3.3 | 12.5 | -2.9 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *1-sat* | 15.2 | -8.3 | 13.9 | -7.4 | 13.2 | -6.5 |
| $\mathcal{L}_{\text{unsup}}$ | *mask* | 13.5 | -3.2 | 13.2 | -2.2 | 13.0 | -1.2 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | *mask* | 13.8 | -7.0 | 13.6 | -6.7 | 13.0 | -6.1 |

**Tab. 5.10.:** Eddies maximum velocity RMSE and bias (cm/s).

### 5.4.5 Conclusion

In this work, we designed a new OSSE emulating 20 years of satellite observations of SSH and SST, available for the community[1], while the previously existing OSSE provided only one year of simulated SSH observations ODC2020. We were able to train an ABED using 3 different loss functions (2 of them learning the reconstruction without ground truth), on three different sets of data (SSH only, SSH and noised SST, SSH, and SST). Compared to our publication in [Archambault et al., 2024b], we tested a new way to retrieve some information from the input of the neural network to force its spatial generalization. This method, consisting of masking SSH inputs in squares randomly located on the images, shows enhanced performances compared to the one introduced by [Martin et al., 2023].

We show a systematic interpolation improvement thanks to the use of SST. Using temperature data (noisy or not), the unsupervised inversion outperforms even the

[1] `https://zenodo.org/records/10551897` [Archambault, 2024]

supervised SSH-only neural network (3.86 cm of RMSE for the unsupervised noisy SST against 4.18 cm for the supervised SSH-only method). This supports further the conclusions drawn earlier in this chapter and in Chapter **4**, stating that SST contextual information can be used to enhance SSH reconstruction.

Using AMEDA, an automatic eddy detection algorithm, we were able to identify cyclones and anticyclones in the ground truth and compare them with the eddies detected in the geostrophic approximation of the different mappings. This allows a deeper physical interpretation than the SSH reconstruction alone. We conclude that SST aids in capturing finer structures that might be overlooked by SSH-only methods and that SST-using methods better render the key physical properties of the detected eddies, such as size, speed, or center position. Furthermore, in unsupervised reconstruction, we show that the non-regularized and regularized inversions have close detection scores, but their errors are different. The regularized inversions exhibited lower recall scores, indicating that certain eddies were not detected due to the smoothing effect of the regularization process. However, they demonstrated higher precision scores, implying increased confidence in the successfully detected eddies.

However, we only applied our training methodologies to OSSE data, with no guarantee of generalization to real observations. The distribution of OSSE and real observations might not be identical, as the observing system is not perfectly modeled, and GLORYS might not perfectly model the relationship between SSH and SST. This is why, in the next section, we test these training methodologies on observations and experiment ways to benefit from supervised and unsupervised learning.

## 5.5 Transfer learning from OSSE to observations

This section focuses on applying the learning strategies presented in the last section to real observations. Because of the OSSE modeling errors, either from the observation operator or from the GLORYS simulation itself, we expect a significant domain gap between OSSE and true data. This domain gap reduces the performances of a neural network trained on our OSSE once applied to real observations by a standard inference. This is why we are interested in directly comparing unsupervised learning methods on observations to supervised learning on pseudo-observations.

The first way to reduce this domain gap is to improve the OSSE realism so that it correctly models the target interpolation problem. We explored this possibility by

proposing a realistic SST noise. However, the errors from the physical model will not be tackled in this way.

Another way to address this issue is to adapt a model trained on a simulation to real observations. In machine learning, this is called domain adaptation, which is a typical case of transfer learning. In our case, as the labels of the target domain are unknown, it is called "unsupervised domain adaptation" (see Section **3.3.1**). In the following, we present a method to reduce the domain gap problem. We first start by introducing the data, the experiment that we conduct, and the results that we obtain.

### 5.5.1 Experiments

#### Satellite observations

To constitute a dataset of real-world observations, we propose the L3 SSH product that we used to recover realistic satellite ground tracks [CMEMS, 2021]. These data are the inputs used in the DUACS optimal interpolation process and are available from the years 1993 to 2023. For the L4 SST product, we use the Multiscale Ultrahigh Resolution SST (MUR SST) [NASA/JPL, 2019]. MUR SST is produced through an optimal interpolation of infrared, microwave, and *in situ* measurements [Chin et al., 2017]. Its resolution is very high (0.01°), so we linearly interpolate the data to our resolution (0.078°). Its temporal coverage is from 2002/05/31 to the present. We select satellite observations from 2002/06/01 to 2022/02/09 for a total of 7194 days, which is the same number of timesteps that our OSSE. We also select the same geographical area between 33° to 43° North and -65° to -55° East. The two data are presented in Figure **5.22**.

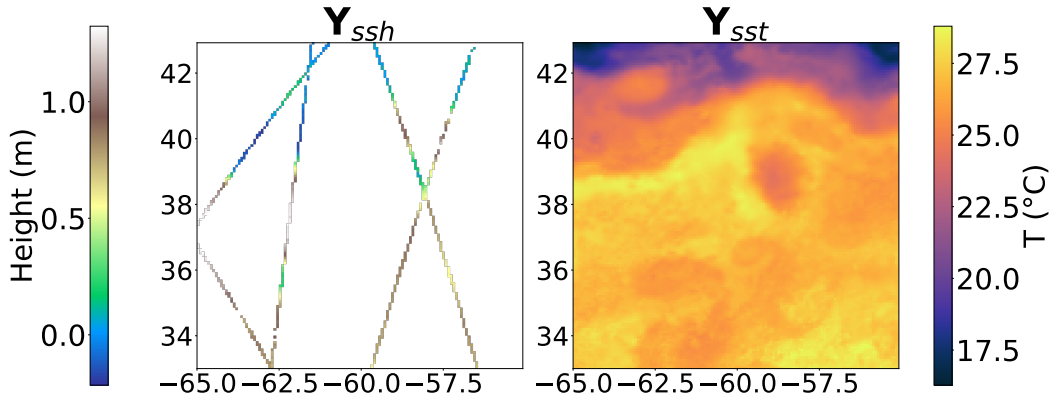


**Fig. 5.22.:** Images of satellite observations of the SSH and the SST, respectively.

#### Training methods

Using the OSSE described in Section **5.4.1** and the satellite data mentioned above, we can define three main training strategies.

- **Observation only**: Replicate the unsupervised training on real-world data with the loss function described in Equation **5.10**. In doing so, there is no domain gap problem, as we entirely rely on satellite data. We use the same training hyperparameters, and the dataset split is the same as the ones used in the OSSE study.
- **Simulation only**: Use the networks supervised on our OSSE on satellite data without domain adaptation. In this setting, the only way to reduce the domain gap is to improve the OSSE. With this setting, we can assess the realism of our OSSE, particularly the SST noise.
- **Pre-training on OSSE and fine-tuning on satellite data**: After the supervised pre-training on OSSE data, we fine-tune the neural network on satellite data for a few epochs using the unsupervised loss. The fine-tuning is done using a small learning rate of $1.10^{-5}$ and a decay of $0.9$. We use an early stopping with a patience of 8 epochs, and we save the best model on the validation set. In this mindset, the neural network is able to learn the physical relationship between variables on our OSSE, as it is supervised to do so. Then, the unsupervised fine-tuning adjusts the neural network weights to satellite data, adapting it to the gap between the source and target domain. Fine-tuning attention layers is particularly efficient, as they can easily increase or decrease the importance of features [Touvron et al., 2022]. In ViTs, it is also possible to fine-tune only attention layers and leave the rest of the weights unchanged, but this idea will not be tested in our work.

### 5.5.2 Results

We test on ODC2021 the learning strategies for ABED neural networks. In all the settings that require unsupervised training (observation-only learning or during the fine-tuning phase), we include the two sub-setting methods *1-sat* and *mask* defined in Section **5.4.2**. The temporal split between train, validation, and testing is the same for the OSSE and the satellite observations dataset. It ensures that the networks pre-trained on one day will be fine-tuned on the same ones. The 2017 year is left aside from the training data of the two datasets so that our model can be fairly

evaluated on the ODC2021. During the test phase, the along-track observations of the Cryosat2 satellite are not used in input so that they can be used as independent data for evaluation as described in Section **5.1.2**. Simultaneously, we evaluate three distinct sets of inputs: SSH-only, SSH and noised SST, and SSH and the ground truth SST (a configuration only possible while training on simulation).

In the following and in Section **5.6.1**, we evaluate the methods on a smaller area than the one used to produce de dataset, between 34° to 42° North and -65° to -55° West. This is because, in the ODC2021, some state-of-the-art methods estimated the SSH fields on this reduced area. To fairly compare all the methods in Section 5.6.1, we also adopt this area in our evaluation. In Table **5.11**, we present the SSH RMSE of the interpolations on the independent Cryosat2.

| Learning method | subset | SSH | SSH+nSST | SSH+SST |
|---|---|---|---|---|
| Observation-only | *1-sat* | 7.07 \| 6.75 | 6.63 \| 6.27 | – |
| Observation-only | *mask* | 6.30 \| 6.10 | 6.04 \| 5.85 | – |
| Simulation-only | – | 6.63 \| 6.35 | 6.28 \| 6.06 | 6.89 \| 6.68 |
| Pre-training & Fine-tuning | *1-sat* | 6.49 \| 6.28 | 6.02 \| 5.82 | 6.04 \| 5.84 |
| Pre-training & Fine-tuning | *mask* | 6.40 \| 6.22 | **5.90** \| **5.72** | 5.93 \| 5.75 |

**Tab. 5.11.:** Along-track SSH RMSE in centimeters (mean score on the left and ensemble score on the right) of 3 ABED networks, computed on one year of data provided by the Ocean Data Challenge 2021. The training strategies include observation-only training (with satellite SSH and SSH+nSST), simulation-only training (SSH, SSH+nSST, SSH+SST), and fine-tuned networks (SSH, SSH+nSST, SSH+SST). For the Fine-tuned networks, when a network is pre-trained with noise-free SST, it is still fine-tuned with noisy satellite SST. In the "Observation" and "Pre-training & Fine-tuning" settings, we include the two subseting methods: *1-sat* and *mask*.

**Training methodology comparison.** The results indicate that the pre-training and fine-tuning approach for ABED systematically yields lower reconstruction errors than simulation- and observation-only learning. An exception is observed with SSH observations-only learning using a random mask subset, which unexpectedly outperforms its fine-tuned counterpart. The improved performance from simulation-only to fine-tuned networks suggests that our training strategy effectively facilitates domain adaptation. Additionally, using a random masking subset significantly enhances the results, mirroring its impact on the simulation data. This is particularly true for networks trained using only observations where random masking leads to the highest improvement.

**Impact of the SST.** As anticipated, incorporating SST data significantly enhances performance compared to networks relying solely on SSH. This can be seen across all experiments except when training on simulation data with noise-free SST, which

leads to the lowest performance. It shows that without a realistic SST noise in the OSSE, the domain gap is more important, and it becomes preferable to learn from SSH only or use only observations. However, after fine-tuning, networks trained on either noisy SST (nSST) or noise-free SST achieve similar performance. This is because fine-tuning adjusts the network weights, adapting it to the new domain. Therefore, a key takeaway from this experiment is that, with an effective fine-tuning strategy, simulating realistic noise in contextual variables is less necessary since fine-tuning itself accomplishes domain adaptation.

## 5.6 Conclusion

Throughout this chapter, we studied SSH Delayed Time interpolation methods based on neural networks. As our objective was to get toward an operational training methodology, we studied various manners to train neural networks, going from unlearned methods such as DIP and MUSTI to unsupervised learning and from simulation training to unsupervised domain adaptation.

In the following, we first present a state-of-the-art comparison on ODC2020 and ODC2021, including the methods described in Section **5.1.2** and the ones that we produced during this thesis. Then, we summarize and conclude this study and give some perspectives.

### 5.6.1 State-of-the-art comparison

#### Benchmark

In the following we give a full state-of-the-art comparison between the SSH interpolations methods. In Tables **5.12** and **5.12**, we compare the methods on ODC2020, using SWOT and nadir-pointing altimetry, and nadir-pointing only, respectively. In Table **5.14**, we compare the methods on the ODC2021. For each benchmark, we include all the methods described in Section **5.1.2** that provided an SSH field in this particular setting. The ConvLSTM introduced by [Martin et al., 2023], was not trained on ODC2020 OSSE, and, to the best of our knowledge, 4DVARNET-SSH-SST was not computed on ODC2021. Regarding our methods, we incorporate DIP and MUSTI in all benchmarks. For ABED estimations, we select the best training methods for each setting. In the ODC2020 nadir-only scenario, we apply ABED supervised on our OSSE directly to the challenge data. Since it is impossible to deseasonalize

the SST with just one year of data, we retrain ABEDs without SST deseasonalization or SST noise and apply them to ODC2020. For ODC2021, we use the best training methods: observations only for ABED-SSH and a combination of simulation and observations for ABED-SSH-SST, each with random mask SSH subsetting.

**Neural networks scores.** First, in all tables, we see a predominance of neural network-based methods among the best performances. This is particularly the case on ODC2020, where all the neural networks exhibit superior performances than OI or data assimilation techniques. In the ODC2021 dataset, the performance gap narrows, possibly because OIs are tuned on real observations and the reference data contains noise, leading to an overestimation of all errors. Notably, DIP estimation remains highly effective compared to other OIs methods, particularly in scenarios where only nadir-pointing data is used. One hypothesis for DIP's slightly lower performance compared to OIs when SWOT data is available is the imbalance between the data coverage of SWOT and nadir data. The significant weighting of SWOT data in the DIP loss function may bias the estimation by overemphasizing SWOT swath data. To address this issue, different weights could be assigned to the loss of SWOT and nadir data, encouraging the network to estimate nadir measurements accurately. In any case, the reported experiments fully support the use of neural networks for the SSH interpolations problem.

**SST improvement.** The true advancement brought by neural networks lies in their ability to fuse SSH and SST information, leading to a major improvement across all benchmarks. On ODC2021, the enhanced scores of ABED-SSH-SST and CONVLTSM-SSH-SST compared to their SSH-only versions emphasize the improvements brought by the SST. On the ODC2020 nadir+SWOT dataset, 4DVARNET-SSH-SST demonstrates remarkable performance in terms of SSH RMSE and current estimation. However, its effective resolution in time and space is lower than that of its SSH-only version. This unexpected result may be attributed to a limitation in the metric. The effective resolution is defined as the largest wavelength at which the reconstruction error is half of the signal, meaning that if a specific frequency is not well resolved, the overall metric suffers. On ODC2020 nadir-pointing-only, ABED-SSH-SST also drastically improves the reconstruction compared to SST-agnostic methods, both in terms of SSH RMSE, current estimation, and effective resolution. Integrating SST in SSH reconstructions seems to be a key feature of interpolation methods, thereby unlocking the full potential of satellite altimetry.

| Method | Type | SST | SUP | $\mu_{nor}$ | $\mu$ | $\sigma_t$ | $\lambda_x$ | $\lambda_t$ | $\mu_u$ | $\mu_v$ |
|---|---|---|---|---|---|---|---|---|---|---|
| DUACS | OI | ✗ | ✗ | 0.922 | 4.56 | 2.85 | 1.22 | 11.3 | 16.0 | 15.0 |
| DYMOST | OI | ✗ | ✗ | 0.926 | 4.32 | 2.48 | 1.19 | 10.3 | 15.7 | 15.3 |
| MIOST | OI | ✗ | ✗ | 0.938 | 3.6 | 2.1 | 1.18 | 10.3 | 13.7 | 13.1 |
| **DIP** | OI &NN | ✗ | ✗ | 0.937 | 3.65 | 2.01 | 1.22 | 9.8 | 13.3 | 13.0 |
| BFN-QG | DA | ✗ | ✗ | 0.926 | 4.28 | 2.56 | 1.02 | 10.4 | 13.7 | 13.6 |
| 4DVarNet | NN | ✗ | ✓ | 0.959 | 2.37 | 1.3 | 0.62 | 4.3 | 10.4 | 10.5 |
| 4DVarNet | NN | ✓ | ✓ | 0.966 | 1.95 | 0.83 | 0.73 | 36.0 | 9.4 | 9.2 |
| **MUSTI** | NN | ✓ | ✗ | 0.954 | 2.64 | 1.21 | 0.62 | 3.4 | 10.5 | 11.1 |

**Tab. 5.12.:** Comparison of the state-of-the-art reconstruction methods on the Ocean Data Challenge 2020 with SWOT and nadir-pointing data. SST stands for whether or not the reconstruction methods are using SST, and SUP stands for whether or not the methods are supervised.

| Method | Type | SST | SUP | $\mu_{nor}$ | $\mu$ | $\sigma_t$ | $\lambda_x$ | $\lambda_t$ | $\mu_u$ | $\mu_v$ |
|---|---|---|---|---|---|---|---|---|---|---|
| DUACS | OI | ✗ | ✗ | 0.916 | 4.89 | 3.02 | 1.42 | 12.1 | 16.6 | 16.2 |
| DYMOST | OI | ✗ | ✗ | 0.911 | 5.18 | 3.05 | 1.35 | 11.9 | 16.7 | 16.8 |
| MIOST | OI | ✗ | ✗ | 0.927 | 4.21 | 2.50 | 1.34 | 10.3 | 14.7 | 14.5 |
| **DIP** | OI &NN | ✗ | ✗ | 0.933 | 3.88 | 1.96 | 1.36 | 9.6 | 13.2 | 14.4 |
| BFN-QG | DA | ✗ | ✗ | 0.919 | 4.70 | 2.73 | 1.23 | 10.6 | 14.8 | 15.3 |
| 4DVarNet | NN | ✗ | ✓ | 0.944 | 3.26 | 1.73 | 0.84 | 8.0 | 12.6 | 12.7 |
| **ABED** | NN | ✗ | ✓ | 0.936 | 3.70 | 2.01 | 1.25 | 8.8 | 13.1 | 14.6 |
| **MUSTI** | NN | ✓ | ✗ | 0.946 | 3.12 | 1.32 | 1.23 | 4.1 | 11.7 | 13.8 |
| **ABED** | NN | ✓ | ✓ | 0.950 | 2.88 | 1.24 | 0.95 | 4.5 | 11.3 | 11.3 |

**Tab. 5.13.:** Comparison of the state-of-the-art reconstruction methods on the Ocean Data Challenge 2020 with nadir-pointing data only. SST stands for whether or not the reconstruction methods are using SST, and SUP stands for whether or not the methods are supervised.

| Method | Type | SST | Learning | $\mu_{nor}$ | $\mu$ | $\sigma_t$ | $\lambda_x$ | $\mu_u$ | $\mu_v$ |
|---|---|---|---|---|---|---|---|---|---|
| DUACS | OI | ✗ | ✗ | 0.88 | 7.66 | 2.66 | 149 | 21.9 | 21.2 |
| DYMOST | OI | ✗ | ✗ | 0.895 | 6.75 | 2.0 | 131 | 21.5 | 20.2 |
| MIOST | OI | ✗ | ✗ | 0.893 | 6.8 | 2.35 | 139 | 21.2 | 20.9 |
| **DIP** | OI &NN | ✗ | ✗ | 0.897 | 6.53 | 2.3 | 131 | 21.0 | 21.6 |
| BFN-QG | DA | ✗ | ✗ | 0.883 | 7.46 | 2.59 | 119 | 21.4 | 20.8 |
| 4DVarNet | NN | ✗ | simulation | 0.898 | 6.49 | 1.86 | 107 | 18.9 | 18.7 |
| **MUSTI** | NN | ✓ | observation | 0.902 | 6.26 | 1.96 | 114 | 19.8 | 18.8 |
| ConvLSTM | NN | ✗ | observation | 0.893 | 6.82 | 1.86 | 114 | 19.9 | 18.8 |
| ConvLSTM | NN | ✓ | observation | 0.9 | 6.29 | 1.6 | 108 | 19.1 | 18.3 |
| **ABED** | NN | ✗ | observation | 0.904 | 6.1 | 1.74 | 111 | 19.6 | 19.8 |
| **ABED** | NN | ✓ | both | 0.909 | 5.72 | 1.58 | 105 | 18.7 | 18.4 |

**Tab. 5.14.:** Comparison of the state-of-the-art reconstruction methods on the Ocean Data Challenge 2021. SST stands for whether or not the reconstruction methods are using SST.

#### Visual comparison

In the following, we discuss a visual comparison of all the benchmarked methods on the ODC2020 nadir-only and ODC2020. Figures **5.23** and **5.24** display the reconstructed SSH and relative vorticity on 2012/11/01, respectively. Areas where the different estimations differ the most are highlighted. In Area 3, there is a very strong signature in the temperature, which the MUSTI and ABED-SSH-SST methods effectively use to estimate a small eddy. Conversely, in Area 4, where the temperature signature of the eddy is less pronounced, the methods SST-using show less improvement.

In Figures **5.25** and **5.26**, we present an example of the SSH and associated relative vorticity on the ODC2021 dataset (2017/01/19). Because of the absence of fully gridded ground truth data in the real-world setup, the interpretation of the results is difficult. Visually, we see smaller and more precise eddies in the methods using SST than in their SSH-only counterparts. To further illustrate the impact of SST on the reconstruction, we plot the alongtrack profile of the interpolations by DUACS, ConvLSTM, and ABED (both with and without SST) against targeted independent data in Figure **5.27**.

We focus on an area where SST improves the results, as SST-agnostic methods clearly overestimate the SSH. By plotting the satellite trajectory on the SST image, we notice that this area corresponds to a small temperature drop. This example highlights the benefit of using temperature to constrain the inpainting process, as this high-resolution information is missing from the input SSH observations. In both simulations and observations, OI methods, particularly DUACS and DYMOST, produce much smoother SSH fields and relative vorticities, explaining their lower effective resolutions. On the other hand, BFN-QG displays significantly more detail, likely due to the Quasi-Geostrophic model. However, most of these details are absent in the simulation ground truth, suggesting that the QG model might be too simplistic for the NATL60 simulation.

### 5.6.2 Summary and perspectives

#### Summary

SSH spatio-temporal interpolation is a challenging inverse problem, leading to important applications in navigation, oceanography, and weather forecasting, among others. In the past years, this problem was tackled by OI schemes, using solely SSH

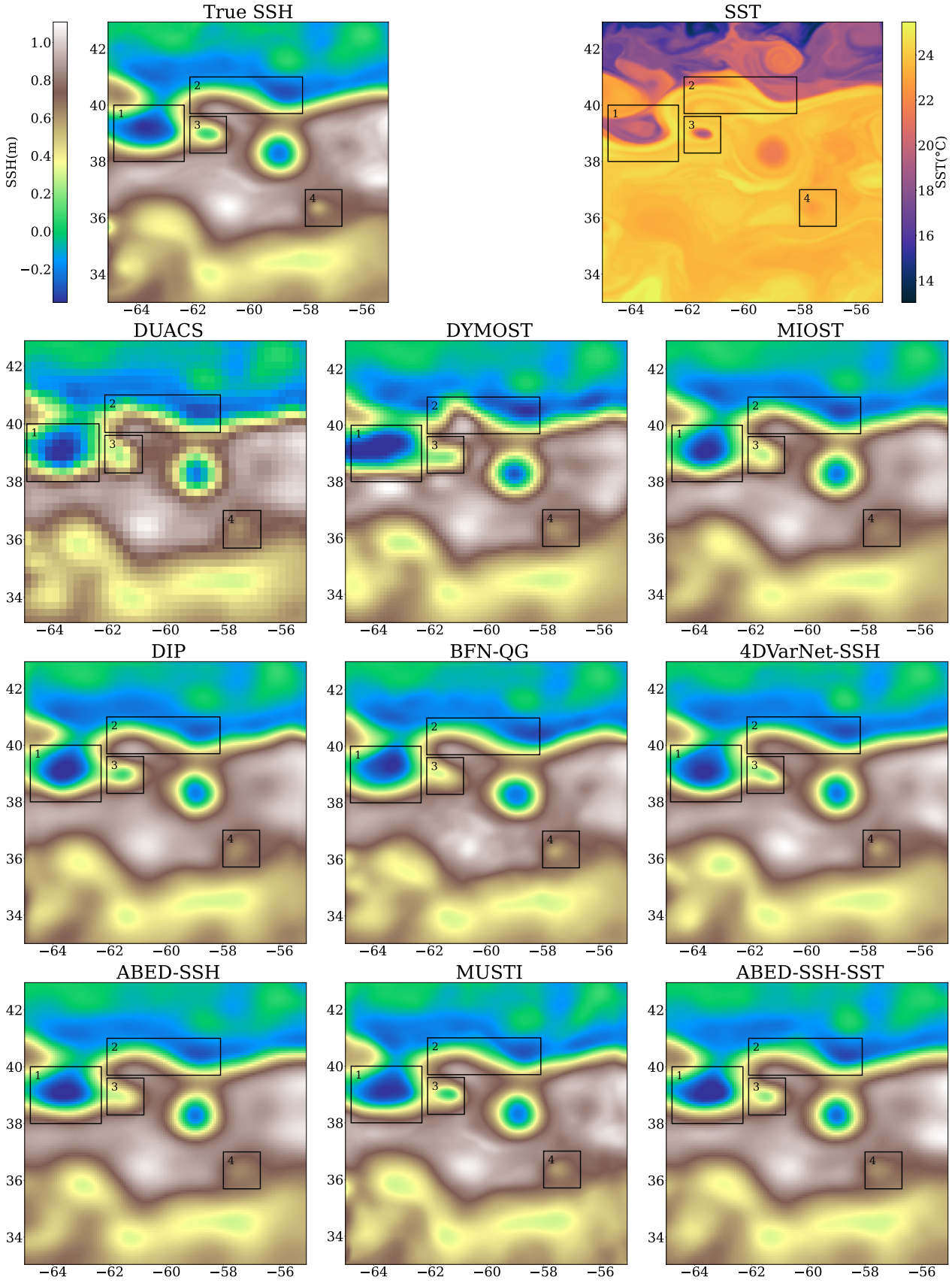


**Fig. 5.23.:** An example of the interpolated SSH on the ODC2020 using nadir-pointing altimeters pseudo observations, on the 2012/11/01. We plot in the first line the ground truth reference of SSH and SST and highlight 4 areas of interest.

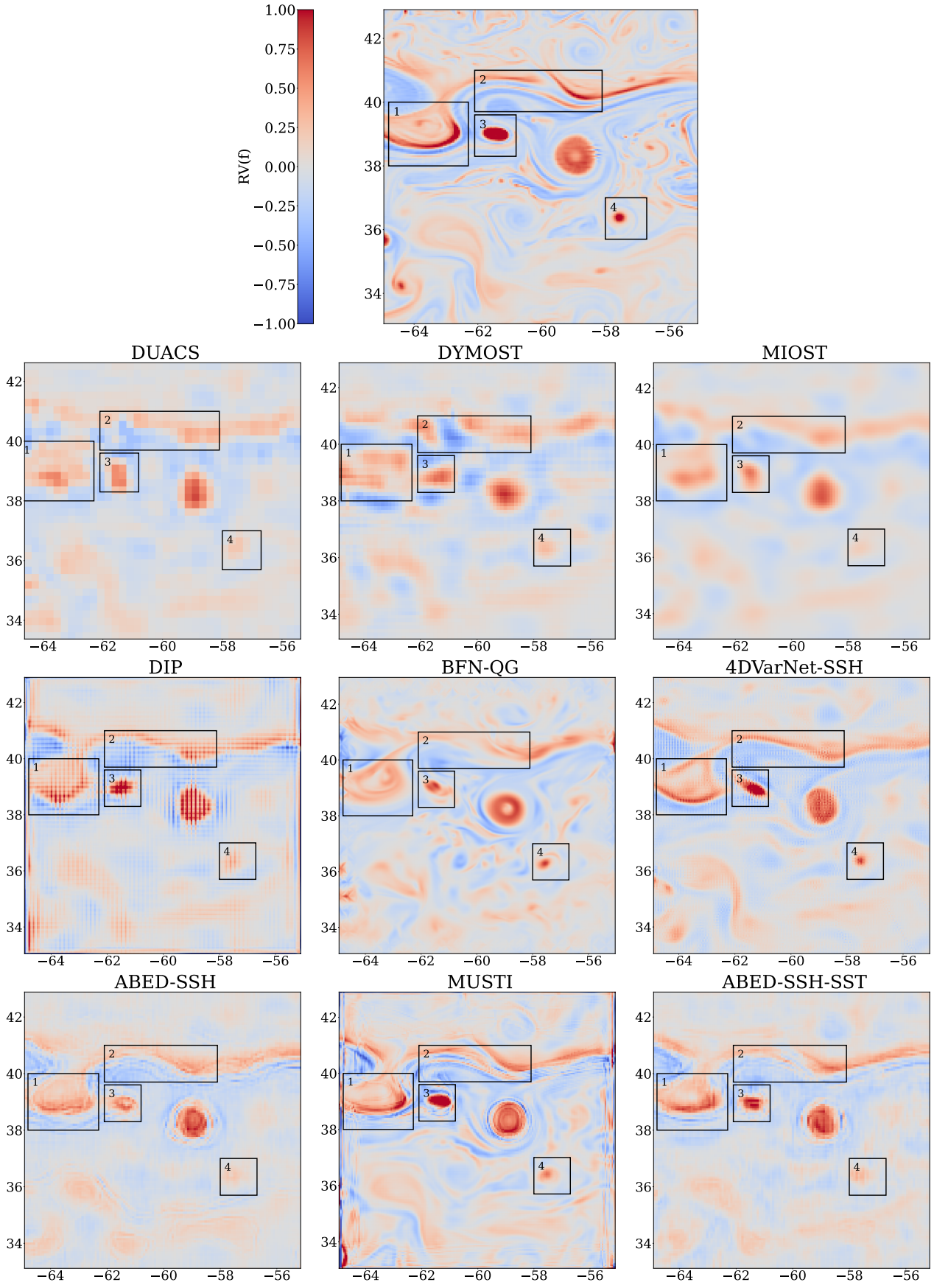


**Fig. 5.24.:** Relative vorticity corresponding to the geostrophic currents of the maps presented in Figure **5.23**.

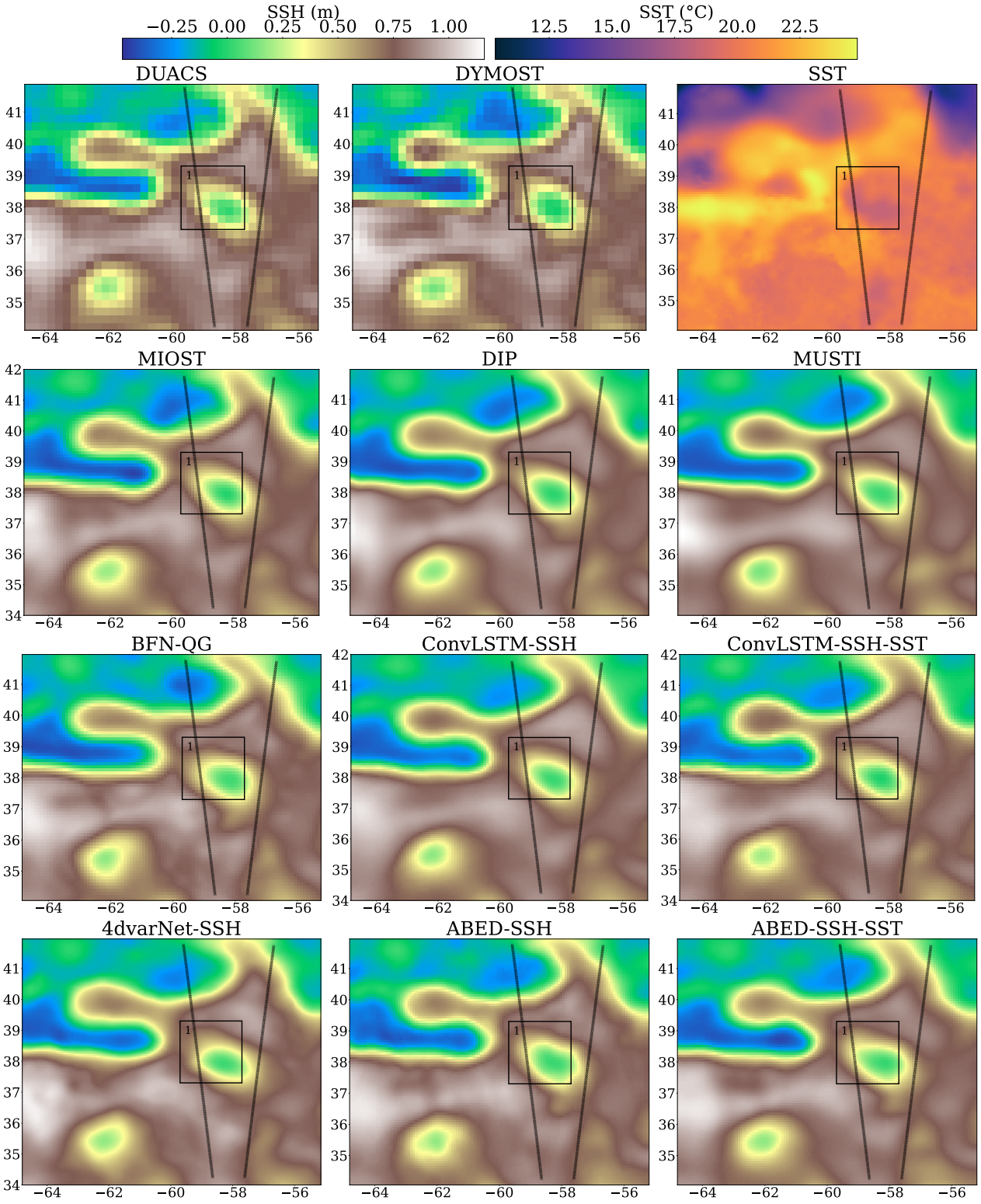


**Fig. 5.25.:** An example of the interpolated SSH on the ODC2021 using nadir-pointing observations, on the 2017/01/19. We plot in the black line the independent satellite data used for inter-comparison of the methods. We highlight an area of interest where the path of the validation satellite passes over a small local minimum of temperature. The along-track profile of some interpolations methods is presented in Figure **5.27**.

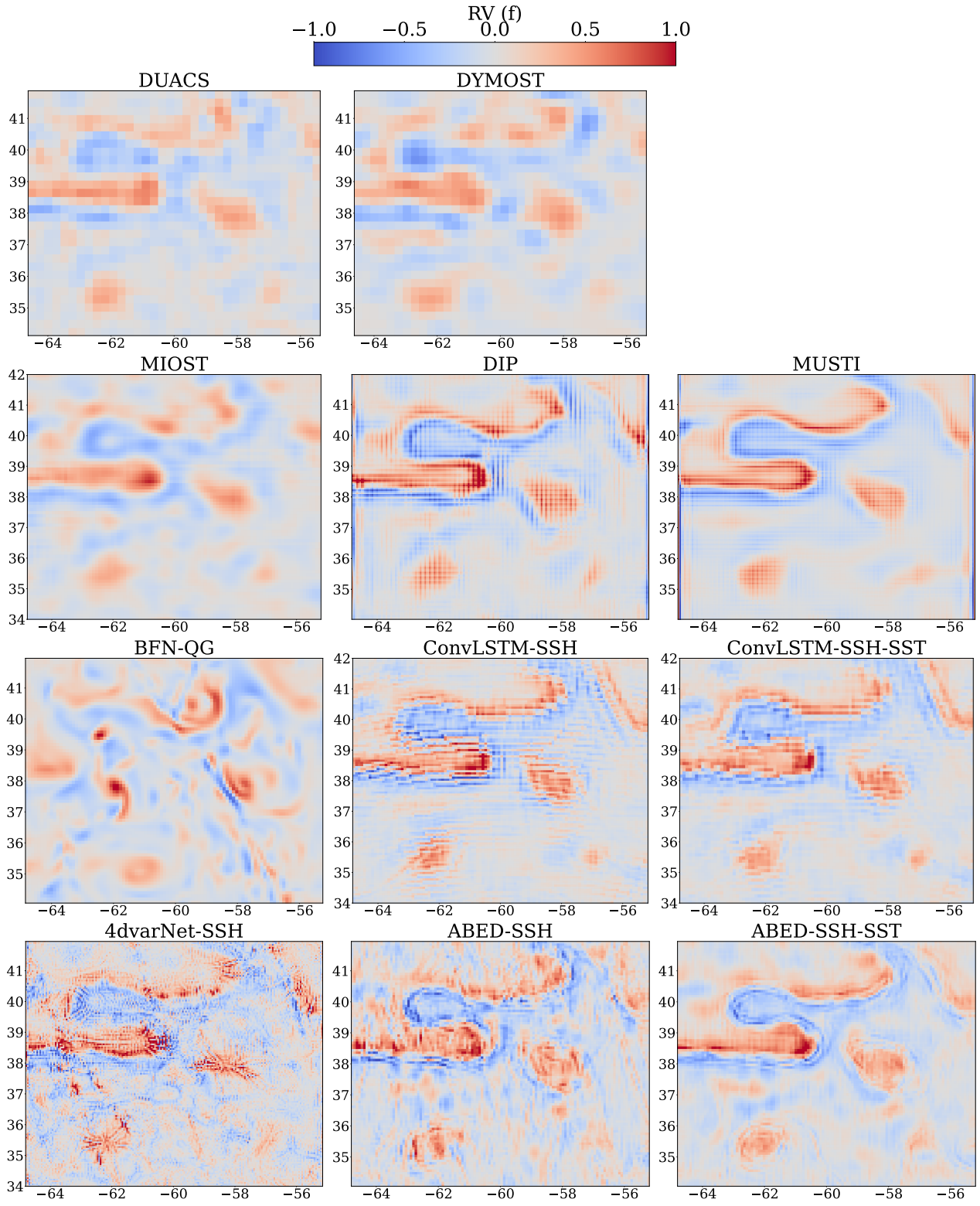


**Fig. 5.26.:** Relative vorticity corresponding to the geostrophic currents of the maps presented in Figure **5.25**.

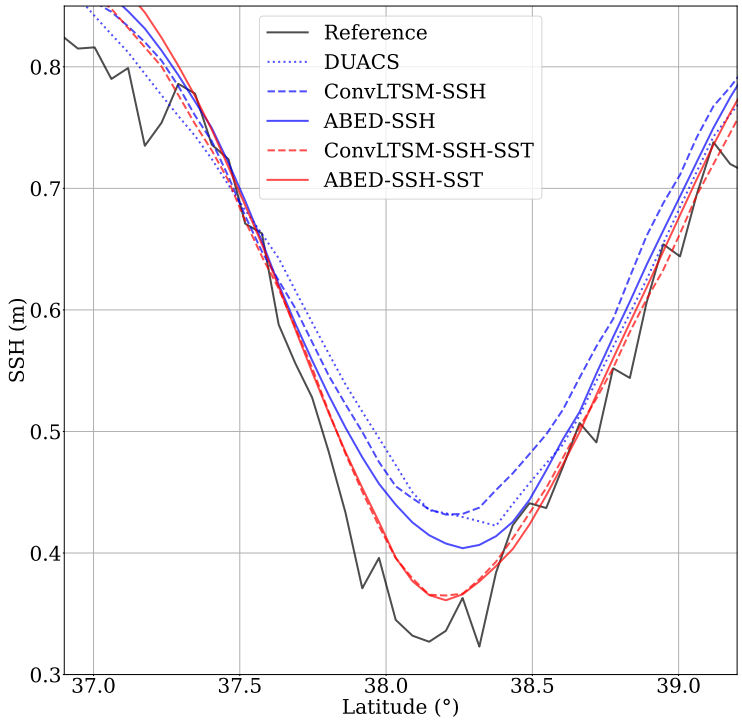


**Fig. 5.27.:** Along track profile of the SSH of the different methods.

data and fine-tuned covariance matrices leveraging expert knowledge. However, these methods present some limitations as they are confined to uni-variate interpolation, whereas including contextual information from better-resolved physical variables seems to be a key to augmenting SSH field quality. On the other hand, deep learning techniques show remarkable flexibility and potential in fusing information from multiple sensors. With the increasing amount of available observations, these data-driven approaches are promising.

In this chapter, we presented a comprehensive exploration of methodologies for producing a high-quality SSH field using SSH observations and SST contextual variables within a deep learning framework.We began by investigating the inversion within a Deep Image Prior strategy, demonstrating the potential to leverage neural networks' inductive bias for interpolation on single observation examples without reference fields. This initial exploration can be seen as a neural network covariance OI, and highlighted the feasibility of statistics-free interpolation using CNNs' intrinsic properties. Building upon this foundation, we adapted the DIP approach to incorporate SST contextual information and multiple observation examples. We show that it enhances the reconstruction on OSSE and OSE. Despite these advancements, refitting to new examples via gradient descent posed significant computational challenges, limiting operational applicability. Recognizing the need for a more efficient

solution, we introduced a new OSSE tailored for DL training. This OSSE enabled us to compare supervised and unsupervised methods, revealing that unsupervised methods lagged behind their supervised counterparts in performance. To bridge this performance gap, we developed a method to transfer physical knowledge from simulations to real observations. By pre-training on an OSSE and fine-tuning on an OSE, we achieved substantial improvements in reconstruction accuracy for a network using SST. This hybrid training approach effectively combined the strengths of both supervised and unsupervised methods, enhancing the overall performance and operational viability of our DL framework. In conclusion, our contributions have laid a robust foundation for operational SSH field reconstruction using DL methods. The advancements in unsupervised and supervised training, combined with the innovative pre-training and fine-tuning approach, provide a promising pathway toward efficient and accurate oceanographic data assimilation. Future work will continue to refine these methods, aiming for broader application and integration into operational oceanography systems.

### Perspectives

In this study, we focused on deep learning interpolations of SSH using a central image within a 21-day time window. While effective, this approach introduces a latency of approximately 10 days that can be reduced to 5 days with an acceptable accuracy. As Near Real-Time (NRT) estimations and short-term forecasts are crucial for operational oceanography, the latency of our method is an important goal. In the next chapter, we present preliminary work aimed at addressing this issue. We explore training neural networks to predict SSH fields in the future based on current observations. This shift towards forecasting represents a critical step in making our DL framework more operationally viable and responsive to the needs of marine applications.

Additionally, there are many unresolved questions that we did not study in this chapter, such as how to adapt this methodology to a global estimation instead of a single area or to use different target and input data. We detail these possible future works and improvements in Chapter **7**.

# 6 SSH Near real-time mapping and forecast

## 6.1 Introduction

In the previous chapter, we discussed the interpolation of SSH satellite observations in a Delayed Time (DT) context. In many operational applications, Near Real-Time (NRT) estimations and forecasts are crucial for the navigation and route optimization of maritime vessels. Since SSH is a reliable proxy for surface currents (at least away from the Equator), providing nowcasts and forecasts of SSH fields is a key feature to improve navigation.

Currently, forecasting systems rely on explicit physical models known as Ocean General Circulation Models (OGCM) with assimilation of observations [Tonani et al., 2015]. A OGCM is a partial differential equations system modeling the physics that rules the ocean's dynamics. First, the OGCM estimates the current ocean state leveraging data assimilation. Then, the dynamical model is applied to the reconstructed state to provide forecast estimations. This is extremely computationally intensive to run, and because of their dependence on OGCM initial conditions, their performance to predict future measurement is limited.

In contrast, data-driven methods are gaining popularity in forecasting complex physical systems. Unlike classical systems, they are not based on physical partial differential equations but learn the dynamics on a dataset. *Graphcast*, a neural network introduced in [Lam et al., 2023] and trained on weather data, showed that it was possible to considerably reduce computing costs and produce precise predictions. In the oceanography community, attempts have been made to replicate *Graphcast* conclusions, such as the recent work from [Wang et al., 2024]. In this study, the authors showed that a ViT can forecast GLORYS data and outperforms explicit global forecast. However, some weakness of this study comes from the fact that the authors use reanalyzed data as inputs of their neural network, which are not available in a NRT setting. Also, as in *graphcast*, the neural network requires the entire state of the physical system, meaning it cannot work without prior assimilation and solely performs the forecast step. On the other hand, some studies show that

data-driven forecasts taking incomplete ocean states as input are possible. [Zheng et al., 2020] showed the feasibility of forecasting SST from satellite observations only near the Equator, and more recently, [Kugusheva et al., 2024] showed that a precise nowcast of the surface currents was possible, even outperforming GLORYS forecast system. This opens the way for forecasting specific ocean fields without running computationally intensive ocean models, at least in the inference step.

In this chapter, we propose an unpublished preliminary study focused on forecasting SSH fields using satellite data. We aim to get even closer to an operational SSH product, available NRT, and applicable to real observations. Building on the conclusions from the previous chapter, we extend our methodology to estimate future SSH fields. Specifically, we compare the performance of neural networks that conduct simultaneous interpolation and forecasting with those that perform these tasks successively.

## 6.2 Proposed method

### 6.2.1 From delayed-time to forecast

Let $\mathbf{Y}_{\mathrm{t}}$ be the satellite observations of SSH and SST at day $t$. In a delayed time product interpolation, the neural network takes as inputs the observations on a time window between $t = 0$ to $t = T$ and estimates the corresponding states on the same time period, following Equation **6.1**:

$$i_{\theta_i}(\mathbf{Y}_{0:\mathrm{T}}) = \widehat{\mathbf{X}}_{0:\mathrm{T}} \tag{6.1}$$

where $i_{\theta_i}$ denotes the interpolation network. The forecast setting is different: the network still inputs observations between $t = 0$ to $t = T$ but estimates the time step between $t = T$ to $t = T + \tau$. In this case, we have the following equation:

$$f_{\theta_f}(\mathbf{Y}_{0:\mathrm{T}}) = \widehat{\mathbf{X}}_{\mathrm{T}:\mathrm{T}+\tau} \tag{6.2}$$

where $f_{\theta_f}$ is the neural network performing the forecast.

### 6.2.2 Forecast methods

#### Simultaneous or successive interpolation and forecast

Several forecasting strategies can be established, given the forecast and interpolation networks. The first one is to perform the forecast directly from observations of SSH without any prior interpolation. In this case, only one network is involved, as in Equation **6.2**. We call this straightforward strategy, *simultaneous forecast and interpolation* (*Sim*), as the two tasks are by the same network without in one step.

The second forecast method uses $i_{\theta_i}$ to interpolate the SSH observations before performing the forecast using the obtained interpolated maps. It corresponds to the following equation:

$$f_{\theta_f} \circ i_{\theta_i}(\mathbf{Y}_{0:\mathrm{T}}) = \widehat{\mathbf{X}}_{\mathrm{T}:\mathrm{T}+\tau} \tag{6.3}$$

This method, which we call *sequential interpolation and forecast (Seq)*, allows training the two networks separately as the two tasks are disjoint.

#### Training methods

**Pre-training.** The interpolation network $i_{\theta_i}$ can be trained using the same procedure that the one used in the last chapter by pre-training and fine-tuning the OSSE and real observations. The forecast network can be pre-trained similarly on the OSSE. For the simultaneous interpolation and forecast, $f_{\theta_f}$ is trained using SSH along-track observations as input and estimates future SSH fields. For sequential interpolation and forecast, $f_{\theta_f}$ is trained using ground truth SSH as input. This is less realistic, as the SSH field cannot be retrieved in such high resolution on real data, but the fine-tuning will adapt this learning to interpolated data limitations.

**Fine-tuning.** The interpolation networks are fine-tuned following the same strategy as in the last chapter. We study two methods for fine-tuning the forecast networks. First, we can apply the unsupervised loss function defined in Equation **5.10**. The only difference is that the observations considered will be in the future compared to the one given in the input of the neural network, except for one timestep, at $t = T$. This means that for this timestep only, we must remove some SSH measurements from the inputs as described in Section **5.4.2**. Here, even if the random masking subsetting improved the results in Chapter **5**, we use the *1-sat* subsetting. As it only concerns one timestep, the performance difference should be lesser.
Second, we can use a standard MSE loss function, where the target is the estimation from another pre-trained and fine-tuned interpolation network. This other network is

trained to provide delayed time interpolated maps, as we only require its estimation in the training phase, not during inference. In doing so, the target is a more accurate estimation than NRT products. In Figure **6.1**, we summarize the training methods described above. We call *Sim* and *Seq* Simultaneous and Sequential interpolation and forecast, respectively, and $\mathcal{L}_{\text{psd\_lab}}$ the loss which uses the pseudo-labels from a DT interpolation network.

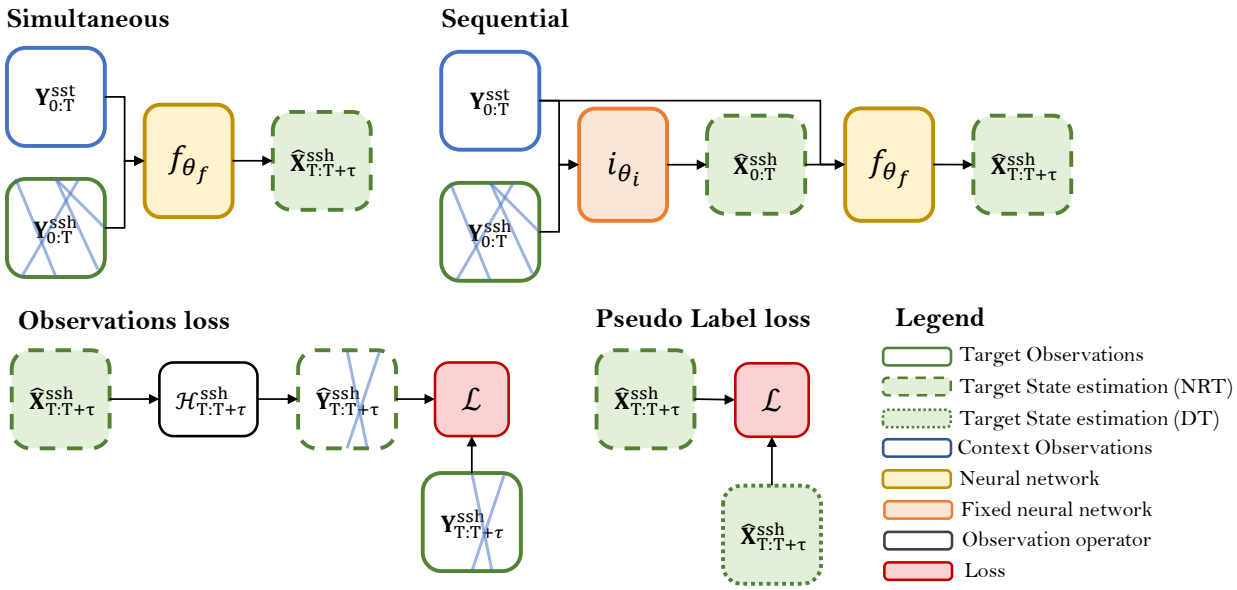


**Fig. 6.1.:** Computational graph of the interpolation and forecast method discussed above. The first line represents the two forecast methods, *Simultaneous* and *Sequential*. In the *Sequential*, the interpolation network is trained beforehand following the methodology of the previous chapter, meaning that its heights $\theta_i$ are fixed in the described setting. The second line represents the two control strategies, the first being the MSE on future along-track observations and the last the MSE on pseudo-labels. These pseudo labels are estimated in delayed time by another neural network, which is not represented here.

**Subsetting SSH inputs**. We now outline the process for removing SSH measurements from the neural network inputs to enhance spatial generalization. During fine-tuning and inference, the forecast neural network receives SSH measurements from all satellites, except for the last day ($t = T$), where data from one satellite is omitted. Both the NRT and DT interpolations are trained using complete satellite data, except on the target days, where data from one satellite is excluded. However, during inference, the NRT interpolation utilizes all available data except for the last day, forcing the forecasting network to produce an accurate nowcast. The pseudo-labels from the DT networks are generated using all the data available at each timestep. We summarize all the inputs SSH details in Figure **6.2**.

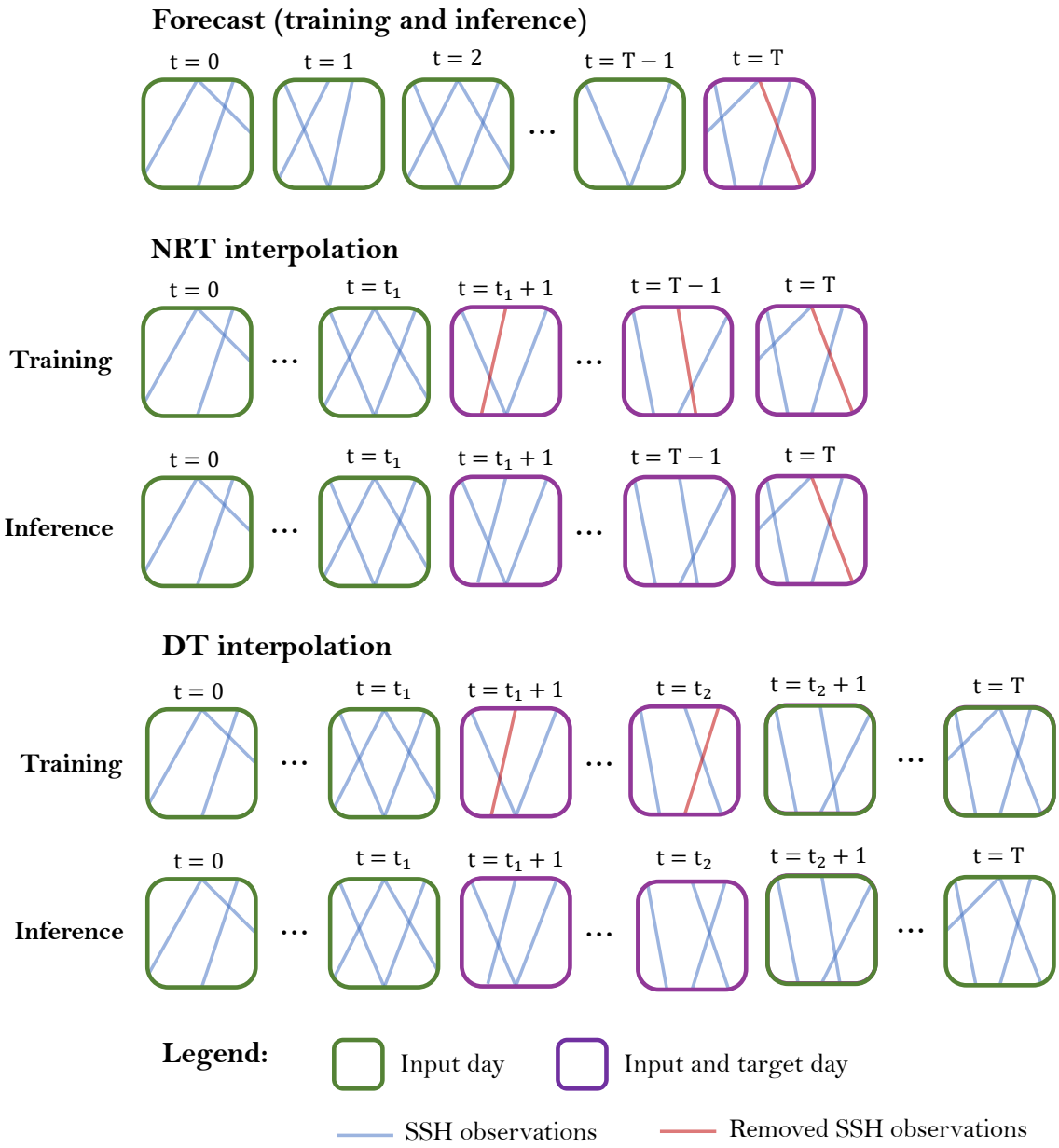


**Fig. 6.2.:** Summary of the SSH input for the forecast of the NRT and DT interpolations.

#### ABED modifications

We propose to use the same ABED as in the last chapter, with minor modifications. First, the temporal length of the input and output images is now different, as there are $T+1$ days in input and $\tau$ days to estimate. We modify ABED to correspond to the input/output sizes by changing the last 3D convolution layer of the network by 2D convolution, where each filter estimates a single timestep. We chose to perform a forecast of $4$ days from $11$ days of data, meaning an output shape of $5$ images (nowcast +forecast) and an input shape of $11$. For sequential interpolation and forecast, the interpolation networks use $21$ days off data in input, but unlike in the last chapter, they estimate $11$ timesteps. We summarize the input and output shapes of the neural networks in Table **6.1** and present a visual overview of input and output .

| Network | Forecast | NRT interpolation | DT interpolation (for pseudo labels) |
|---|---|---|---|
| Input size | 11 | 21 | 21 |
| Output size | 5 | 11 | 11 |
| Input timesteps | [0:10] | [0:20] | [0:20] |
| Target timesteps | [10:14] | [10:20] | [5:15] |

**Tab. 6.1.:** Summary of the input-output shapes and timesteps of the different neural networks.

## 6.3 Results

### 6.3.1 Reconstruction comparison

In the following, we present the results of the forecast with ABED networks evaluated on the Cryosat-2 satellite along-track data during the year 2017, unused during training. We chose the best training strategy from the last chapter, meaning a pre-training on our OSSE and fine-tuning on observations using a deseasonalized noisy SST. We test the two forecast strategies: simultaneous, denoted Sim, and Sequential, denoted Seq. We test the two control methods for each strategy: the MSE on observations and pseudo-labels. We train 3 neural networks and evaluate their ensemble reconstruction by averaging their estimations. We present a graph of the obtained RMSE as a function of the forecasting time in Figure **6.3**.

We compare our estimations to DUACS estimations provided by the ODC2021. This comparison has some limitations. Firstly, this version of DUACS is a delayed-time

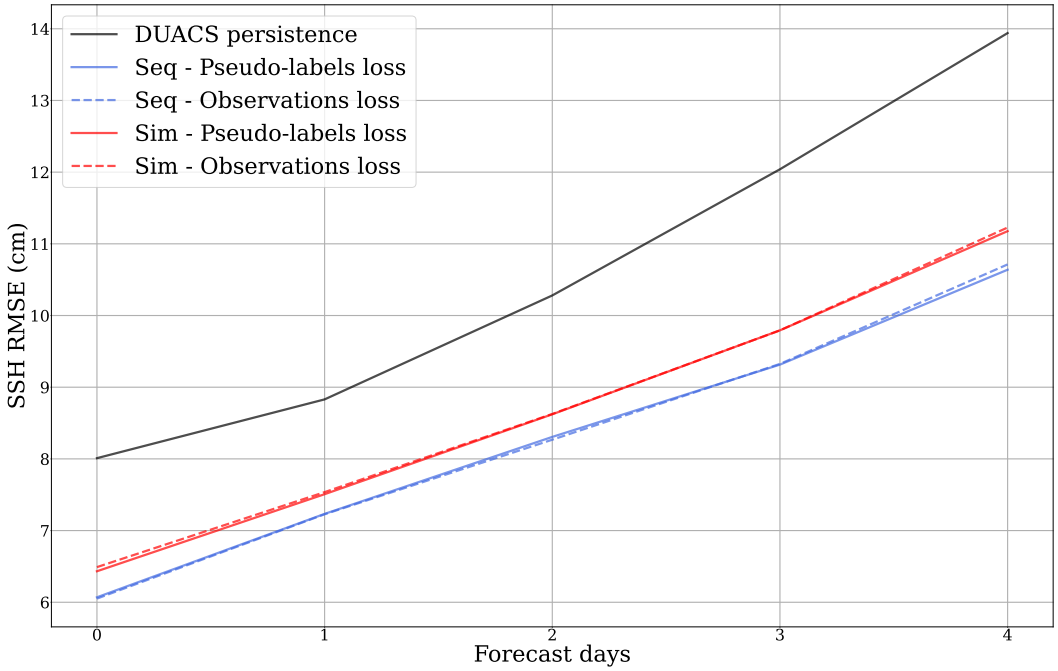


**Fig. 6.3.:** RMSE of the NRT (day=0) and forecast estimations. We compare to the DUACS persistence.

product and utilizes future SSH observations to create its maps. However, it does not incorporate Cryosat-2 data, whereas our forecast does, at least for the past few days. Additionally, since there is no forecast version of DUACS, we use its persistence for comparison, meaning that we copy the last time step to produce future estimations. Although this is not a fair comparison, it can still offer a useful perspective.

Our results show that all our NRT and forecast interpolations clearly outperform the DT DUACS and its persistence. As expected, the RMSE increases with the forecasting time for all methods. The sequential interpolation and forecast approach offers improved performance compared to the simultaneous method, with an improvement of approximately 0.5 cm in RMSE. Surprisingly, there is almost no difference in scores between the two loss functions, even though we expected the pseudo-label loss to perform better due to its use of more information, leading to better control. However, it shows that given an efficient DT interpolation, it is possible to train forecast in a standard supervised fashion, where we replace the ground truth with pseudo labels.

To evaluate the effectiveness of our forecasts, we compare the difference between the RMSE of the forecasted fields and the persistence of the NRT field. Figure **6.4** illustrates this difference for our four forecasting methods. All methods significantly outperform the reconstructions when compared to their persistence, demonstrating

that the neural network has learned at least a partial evolution of the ocean state, emphasizing the value of forecasting. Additionally, methods trained with pseudo-label loss exhibit a smaller but still notable increase in RMSE due to persistence. This is likely because methods trained with observation loss tend to generate erroneous small structures in the SSH NRT fields, leading to higher persistence errors.

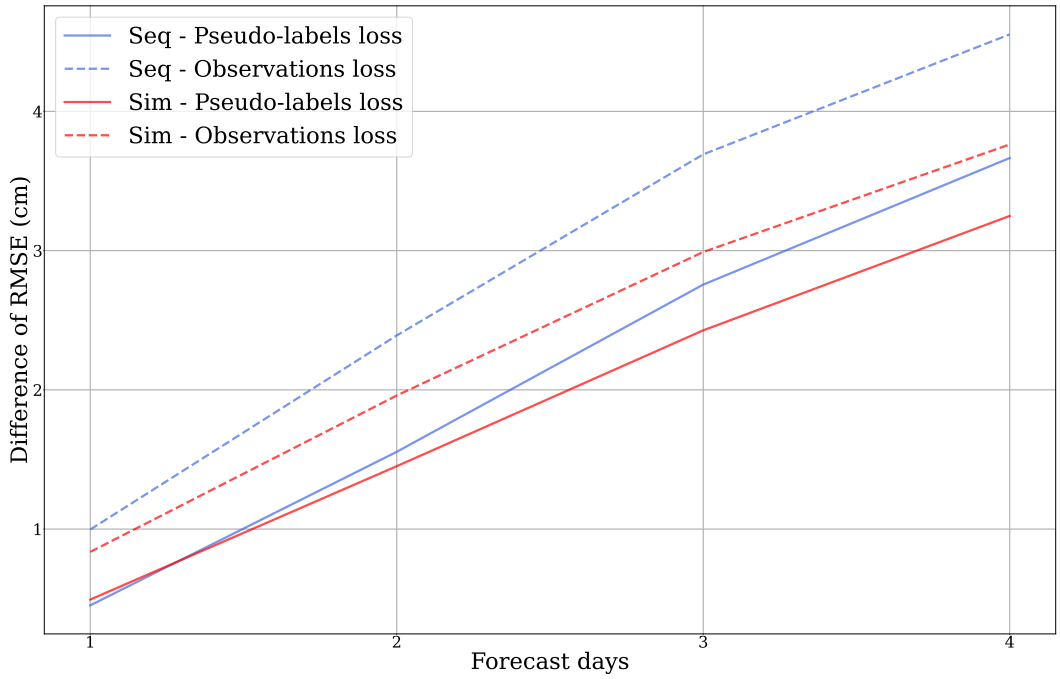


**Fig. 6.4.:** Difference of RMSE scores between the persistence and the forecast of the methods. A positive score means that the forecast is better than the persistence of the NRT field.

### 6.3.2 Visual comparison

In Figure **6.5**, we present an example of the NRT and four-day forecast from January 15, 2017. We selected the Sequential forecast with pseudo-label loss and plotted the SSH fields, their associated relative vorticity, and the differences from the DT pseudo-label fields. It is important to note that some structures might be missed by both the pseudo-labels and the forecast fields, and therefore, will not appear in the difference map. Nonetheless, they still provide a good viewpoint to look at the shapes of the errors.

The SSH fields and their associated relative vorticities show that longer forecasts result in smoother fields. This indicates that small spatial structures are harder to predict due to their rapidly changing locations. The difference maps reveal

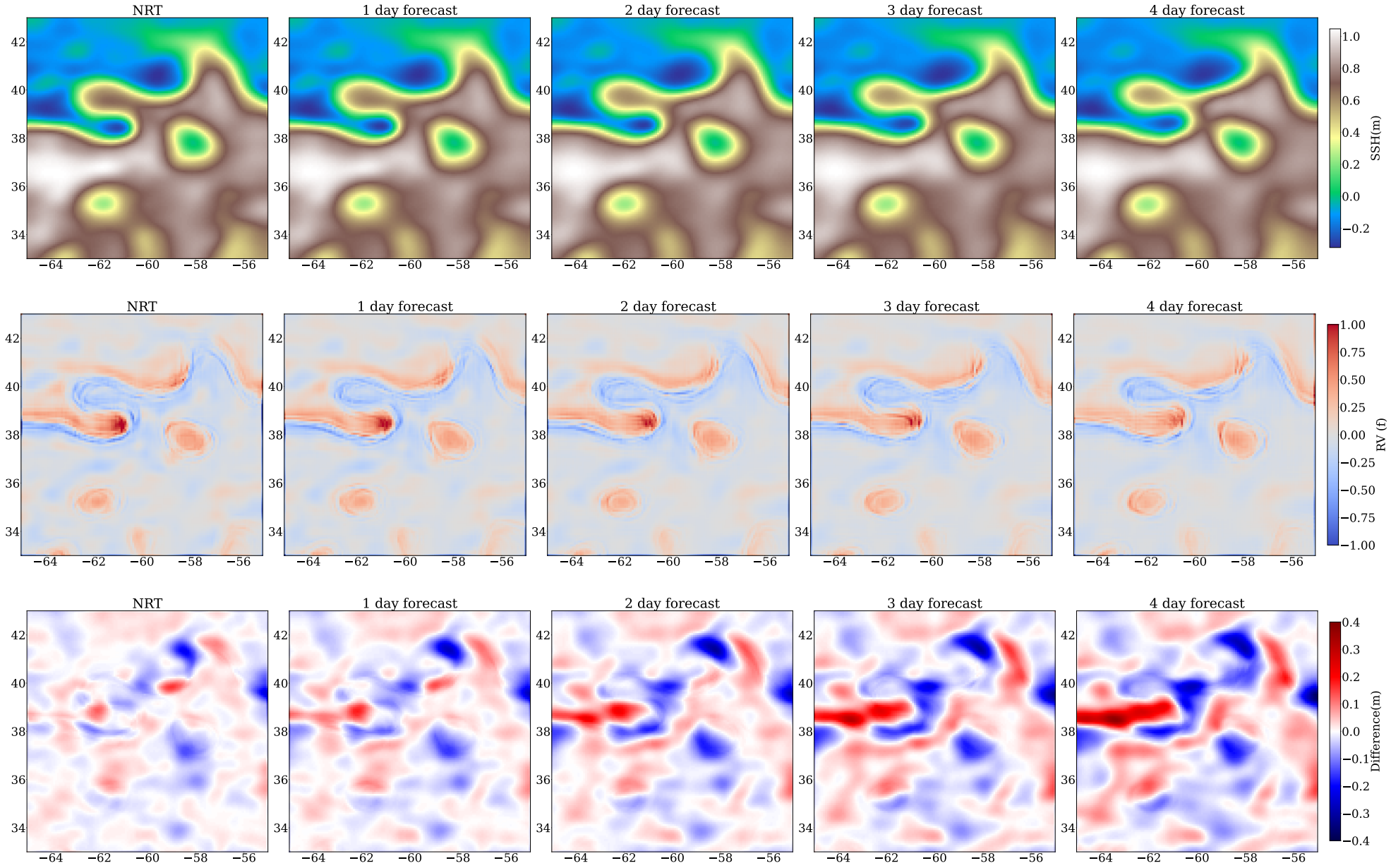


**Fig. 6.5.:** SSH maps, relative vorticities, and difference with DT pseudo labels of the sequential forecast using pseudo-labels loss.

progressively larger errors with increasing forecast length, especially in regions with intense SSH gradients along the north-south frontier of the Gulf Stream.

## 6.4 Conclusion

We adapted the methodology from the previous chapter to NRT and forecast estimations of SSH fields. It involved supervised pre-training and unsupervised fine-tuning of ABED, with the loss constrained by future data relative to the inputs. We tested two forecast types: sequential, where two networks separately perform the interpolation and forecast tasks, and simultaneous, where both tasks are performed by a single network. During fine-tuning, we tested two loss functions: the MSE on observations and the MSE on pseudo-labels generated by a third DT neural network. Our results show that the sequential approach yields better reconstruction than the simultaneous one, although both loss functions perform identically. This preliminary work can be improved in several ways, including exploring different neural architectures for the forecast and interpolation networks since the two tasks are distinct. For instance, incorporating skip and residual connections in the forecast network, which uses complete SSH fields as inputs, could be beneficial. Additionally, it is surprising that the loss on pseudo-labels does not outperform the loss on observations alone. Further experiments are needed to understand this phenomenon.

# 7 Conclusion

## 7.1 Summary

The satellite remote sensing era has produced unprecedented multivariate observations of the oceans. Among them, the Sea Surface Height (SSH) is an important one as it is a proxy for surface currents, impacting heat and nutrient transports as well as navigation. Currently, the SSH measurements are sparse and must be interpolated to produce a complete field, which is done through Optimal Interpolation (OI) in operational oceanography. However, OI-based systems have low spatial effective resolutions, and providing a high-quality field is still an open challenge.

Throughout this thesis, we tackled the SSH reconstruction problem using Deep Learning (DL) models: statistical methods able to learn complex relationships on a dataset. We demonstrate that SSH quality can be improved by this mean, especially when deep neural networks use information from other physical variables linked to the SSH. In particular, we used the Sea Surface Temperature (SST) data as a contextual variable in SSH reconstruction, as it is physically related to SSH and measured with higher spatiotemporal coverage and resolution.

### 7.1.1 Downscaling

First, we examined SSH reconstruction in a downscaling setting. Downscaling, also known as Super Resolution (SR) in imaging, aims to produce a high-resolution field from a low-resolution input. This approach applies to situations where an existing low-resolution interpolated product is available, which we seek to enhance. We trained deep learning models to perform SR on data from a numerical simulation, demonstrating that incorporating SST improves reconstruction across all considered metrics. We compared two neural architectures: fully convolutional CNNs and a residual subpixel convolution variant. The latter network is easier and faster to train, achieving better performance in terms of SSH RMSE. However, subpixel convolutions introduce checkerboard artifacts, resulting in a noisy SSH spatial gradient and poor geostrophic current estimation. To address this issue, we employed an additional residual block to estimate currents, effectively mitigating the problem.

### 7.1.2 Interpolation

The operational application of downscaling is challenging due to the need for accurately modeling the low-resolution input. To overcome this, we shifted focus from downscaling to interpolation, using sparse along-track SSH observations and simplifying the observation operator. Our primary goal was to develop a DL strategy capable of learning directly from observations, allowing it to be applied to real data without adaptation. We began by studying a Deep Image Prior (DIP) interpolation approach, where a neural network is fitted to a single time window of observations. Here, the neural network architecture acts as a regularization, similar to covariance matrices in OI schemes. Using a sliding window technique, we demonstrated that this method, which combines OI and neural networks, outperforms traditional OI on all tested benchmarks. Building on this, we developed a Multi-variate Unsupervised Spatio-Temporal Interpolation (MUSTI), extending the DIP principle to incorporate SST contextual information, resulting in significant interpolation improvements across all benchmarks. Despite these advancements, both methods require refitting to new observations, creating a computational burden. To overcome this issue, we trained a neural network to perform interpolation using both SSH and SST, rather than overfitting to SSH observations alone. We introduced a new OSSE dataset, containing twenty years of pseudo-observations and their ground truth, and developed training methodologies using only observations, demonstrating their effectiveness. Finally, we applied transfer learning to real satellite data, showing that methods trained on both simulation and real observations outperformed those using either dataset separately.

### 7.1.3 Forecast

Finally, we adapted our methodology to estimate nowcast and forecast fields to get toward an interpolated SSH field that is available in real-time. We compared two forecast strategies: a sequential approach, where the interpolation is performed by a separate neural network beforehand, and simultaneous forecast and interpolation, where the two problems are solved by the same neural network. We show that this sequential approach leads to better results when applied to real data. Additionally, we tested two loss functions: MSE on observations and MSE on pseudo-labels generated by a third Delayed Time (DT) network, both of which produced similar results.

The methods developed in this thesis can also be applied to downscale, interpolate, and forecast other geophysical fields. Other doctoral students are presently

evaluating their usefulness for Chlorophyll-A and Phytoplankton Functional type concentration interpolation and Lightning Strikes forecasting.

## 7.2 Perspectives

### 7.2.1 Global product

One of the main blind spots in all our experiments is that we trained a neural network on a single area: the Gulf Stream between latitudes $33°$ to $43°$ and longitudes $-65°$ to $-55°$. We did not study the behavior and performances of our network in other areas, and its generalization is not guaranteed. The physics ruling the surface dynamics depends on multiple region-specific factors. For instance, latitude impacts the surface circulation as it modifies the geostrophic approximation [Stewart, 2006]. Specifically, near the Equator, the geostrophy approximation does not apply, and physics are thus driven by other phenomena. The shape of the ocean floor also impacts on the deep circulation, as the currents are guided around the ground relief. As the surface currents approximately follow the deep ocean currents, the bathymetry, i.e. the depth of the ocean, provides information on the surface circulation [Gille et al., 2015]. All these area-specific conditions resolve in complex differences that a global DL model must learn to be effective. In particular, passing latitude, longitude, and bathymetry as inputs should help the SSH reconstruction. Positional encoding, used in sequence to sequence translation [Vaswani et al., 2017], could be an efficient way to inform the neural network of area-specific variables and thus allow feature-specific learning.

Another question regarding global estimation is whether to use a global model or several local ones. An idea could be to separate the areas of similar physics, close latitudes, and coastal regions, for instance, and train one model by domain. Another strategy could be to train a global model and perform several fine-tunings in local areas. In machine learning, this is called a foundation model, which is trained on a very wide dataset that can serve as a pre-trained model that can be more easily adapted to other tasks [Kolides et al., 2023].

### 7.2.2 Including more data

#### Contextual information

We have demonstrated the benefit of using multi-physical information, specifically SST, to enhance SSH reconstruction by implementing a flexible neural network framework. Integrating data from diverse physical sources exhibits promising outcomes, and the diversity of the contextual information employed is necessary to reach the true potential of DL data fusion.

In this thesis, we used complete SST fields, either coming from a physical simulation with or without noise addition or from optimally interpolated satellite observations. However, using an incomplete L3 SST product is also possible by total masking the temperature at the location of clouds on simulations. In doing so, the SST image has a more consistent effective resolution, as no artificial smoothing will occur with OI. However, as L4 SST products also include information from low-resolution microwave sensors and *in situ* measurements [Martin, 2014], another idea would be to simultaneously use L3 and L4 SST products, letting the model decide which is more relevant for each location.

Other physical measurements might improve the reconstruction, such as chlorophyll-A fields. In the ocean, chlorophyll-A is produced by phytoplankton that is advected by currents [Kahru et al., 2012]. Chlorophyll-A concentration in the water is measured through direct ocean color imaging, meaning that, like SST, it is retrieved with a wide coverage and high resolution. Because of these reasons, Chlorophyll-A should be an adequate additional contextual information in SSH estimation.

#### SWOT observations

With the launch of the Surface Water and Ocean Topography (SWOT) satellite in December 2022, a new type of altimetric is available. This satellite offers high-resolution SSH measurements over a wide swath, with the potential of significantly enhancing the resolution of altimetry products [Gaultier et al., 2016] and deep learning estimations [Fablet et al., 2021, Archambault et al., 2023]. Due to its comprehensive measurements, it is possible to compute the geostrophic approximation directly from SWOT observations, which can be helpful for current estimation, at least in high latitudes.

However, incorporating SWOT observations into an operational deep-learning framework presents several challenges. If we aim to pre-train a model on simulations, we

must include SWOT-like pseudo-observations in the OSSE. This involves sampling the simulated ground truth at the locations and resolutions corresponding to the real-world SWOT satellite. Given that SWOT's resolution can be very high (up to 250 meters for the most precise products), ensuring that the ground truth resolution aligns with SWOT's effective resolution is essential. Additionally, SWOT's Level 3 (L3) products are currently available for only a few months. Consequently, we need to develop methods to fine-tune neural networks using these shorter data periods. As more SWOT data becomes available, all these limitations should be overcome.

#### Ocean Curents Estimation

As explained in Section **2.4.2**, SSH is linked to ocean surface currents, which are crucial in many applications. However, even if geostrophy is a valid approximation under certain conditions (far from the Equator, no tide, no wind-driven currents...) [Stewart, 2006], estimating the ageostrophic component of the circulation can be challenging, yet useful. The current estimation problem is more ill-posed than the SSH interpolation, as we have fewer direct observations of currents. Drifters measurements [SEANOE, 2024] are sparser than SSH along-track data, which limits their use to train neural networks. [Kugusheva et al., 2024] introduced HIRES-CURRENTS-NET, a neural network estimating circulation from SSH and SST, trained on a OSSE, which achieves impressive results. Due to the differences between the SSH and current reconstruction problems, it is possible that we must develop new training strategies, relying more on OSSE realism than on fine-tuning, following [Kugusheva et al., 2024]. Specifically, we believe that splitting SSH and current estimations tasks could improve the performance. In doing so, we can apply the pre-training and fine-tuning strategy presented in this thesis to SSH reconstruction and develop a new neural network using the SSH reconstructed field and SST to estimate currents. Therefore, retrieval and forecasting of sea currents is a natural direction for prolonging the research presented in this manuscript, especially the transfer learning components of our approach.

### 7.2.3 Exploring state-of-the-art architectures and training method

Methodology-wise, several improvements can be brought to improve the deep learning method. Firstly, in Chapters **5** and **6**, we focused on encoder-decoder architectures, embedding attention layers. While we believe that it is most important to identify the optimal data, preprocessing, and training procedure for our problem, the

network architecture is still an important component. In the optic of pushing further reconstruction performances, using more advanced neural network architectures is a crucial feature. The never-ending improvement of neural architecture design leaves us with difficult choices. Many architectures have been proposed for SSH interpolation, such as 4DVarNet, a combination of ConvLSTM and Unet [Fablet et al., 2021], or other ConvLSTM-based encoder-decoder [Martin et al., 2023]. However, the Vision Transformer (ViT) (see Section **3.3.2**) seems particularly adequate for the task of interpolation and forecast. ViT can handle sparse and noisy satellite fields as their self-attention mechanism allows them to focus on the relevant inputs only. They are also well suited to model time series of observations through spatio-temporal positional encoding. Several studies show the advantages of such architectures; [Wang et al., 2024] and [Chen et al., 2023] used a ViT to forecast ocean and weather data, respectively, showing increased performances compared to physics-based models.

Additionally, training methodologies could be further refined. In Chapter **4**, a super-resolution network was introduced and supervised using simulation data. This neural network can be employed as the conditional generator within a Generative Adversarial Network (GAN) framework (see Section **3.3.1**). Within this setup, the network is trained to generate images resembling the SSH fields of the simulation by attempting to deceive another network, called the discriminator. This enhancement should add detail to the super-resolved images; however, ensuring the realism of these details remains a challenging task.

The interpolation techniques discussed in Chapter **5** could be significantly enhanced through the application of generative deep learning models. Specifically, diffusion models, as detailed in Section **3.3.1**, have demonstrated impressive performance on natural images. They can face significant issues, notably overcoming the smoothness and lack of visual details that characterize most natural image training datasets [Lugmayr et al., 2022, Kawar et al., 2022]. However, their application to oceanographic data remains a challenging task. A key question is how to adapt these methods to scenarios where only incomplete satellite observations are available. We are interested in experimenting the training methodology employed in Chapter **5**, involving pre-training on simulation and fine-tuning on observations, for a DDPM. Despite these challenges, generative modeling holds great promise for improving interpolation results. These models can enhance reconstruction quality and provide insights into the target distribution. As SSH reconstruction is an ill-posed inverse problem, the solution to the problem is not unique, and being able to generate different solutions is a valuable asset. By generating multiple samples from the generative framework, we can produce different realizations of the same SSH map

from identical input data. This collection of samples not only offers a measure of uncertainty but also provides a more comprehensive representation of the SSH field.

# Losses and metrics

# A

We define hereafter some of the loss and metric functions that we use in this thesis.

## A.1 Regression

Let $\mathbf{x} \in \mathbb{R}^N$ be a reference and $\widehat{\mathbf{x}} \in \mathbb{R}^N$ an estimation. In the following we suppose that $\mathbf{x} = (x_0, x_1, ...x_{n-1})$ and $\widehat{\mathbf{x}} = (\widehat{x}_0, \widehat{x}_1, ...\widehat{x}_{n-1})$ are vectors but the definitions can apply to images, time series of images... without loss of generality.

The Mean Squared Error (MSE) loss is defined by:

$$\mathrm{MSE}(\mathbf{x}, \widehat{\mathbf{x}}) = \frac{1}{n} \sum_{i=0}^{n-1} (x_i - \widehat{x}_i)^2 \tag{A.1}$$

The Root Mean Squared Error (RMSE) metric is the square root of the MSE:

$$\mathrm{RMSE}(\mathbf{x}, \widehat{\mathbf{x}}) = \sqrt{\mathrm{MSE}(\mathbf{x}, \widehat{\mathbf{x}})} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (x_i - \widehat{x}_i)^2} \tag{A.2}$$

And the Root Mean Square (RMS) of $\mathbf{x}$ is defined by:

$$\mathrm{RMS}(\mathbf{x}) = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} x_i^2} \tag{A.3}$$

## A.2 Detection

Hereafter, we consider a detection setting, taking the example of the eddy detection experiment presented in Section **5.4.4**. Let us consider two sets of eddies, $\mathrm{E}$ and $\widehat{\mathrm{E}}$, which correspond to the eddies of the ground truth and the one of the estimation, with cardinals $n_1$ and $n_2$, respectively. We define the true positives as the elements

present in the two sets, i.e., the eddies that are accurately retrieved in the estimated map. The number of true positives is defined by:

$$\mathrm{TP} = \mathrm{card}(\mathrm{E} \cap \widehat{\mathrm{E}}) \tag{A.4}$$

The false positives correspond to elements that are in $\widehat{\mathrm{E}}$ but not in $\mathrm{E}$, i.e., the eddies that are present in the estimation but not in the ground truth. Their number is given by:

$$\mathrm{FP} = n_2 - \mathrm{TP} \tag{A.5}$$

On the contrary, the false negatives are the elements present in $\mathrm{E}$ but not in $\widehat{\mathrm{E}}$, i.e., the eddies of the ground truth missing from the estimation. Their number is given by:

$$\mathrm{FN} = n_1 - \mathrm{TP} \tag{A.6}$$

We can define several detection metrics used in this thesis from these values. First, the recall tells us the proportion of actual positive instances that were correctly identified by the detection (a recall of $1$ means that all ground truth eddies were detected). It is defined by:

$$\mathrm{recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = \frac{\mathrm{TP}}{n_1} \tag{A.7}$$

The precision gauges our trust in the detected eddies (a precision of $1$ means that all eddies in the simulation were also present in the ground truth).

$$\mathrm{precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = \frac{\mathrm{TP}}{n_2} \tag{A.8}$$

The $\mathrm{F}_1$ score is the harmonic mean of recall and precision, which allows aggregating the precision and the recall metrics. A value of $1$ means a perfect detection: all ground truth eddies were detected, and the estimation produced no false positives.

$$\mathrm{F}_1 = 2\frac{\mathrm{recall} \times \mathrm{precision}}{\mathrm{recall} + \mathrm{precision}} \tag{A.9}$$

# B Interpolation supplementary materials

## B.1 Impact of the SST deseasonalization on reconstruction

In the results presented in Section **5.4.3**, we deseasonalized the SST data in the inputs of the neural networks. In Table **B.1**, we show the RMSE of the neural network using "native" SST and the ones using deseasonalized SST. We see that this preprocessing operation decreases the RMSE in every scenario.

| Loss | SSH+SST | SSH+SST (deseasonalized) |
|---|---|---|
| $\mathcal{L}_{\text{sup}}$ | 3.19 \| 2.88 | **2.97** \| **2.63** |
| $\mathcal{L}_{\text{unsup}}$ | 3.50 \| 3.09 | 3.56 \| 3.16 |
| $\mathcal{L}_{\text{unsup\_reg}}$ | 3.52 \| 3.26 | 3.48 \| 3.20 |

**Tab. B.1.:** SSH reconstruction RMSE in centimeters (mean score on the left and ensemble score on the right) of 3 ABED networks. The interpolation is trained using the three different losses described in Section **5.4.2** with the following settings: SSH + noise-free SST and SSH + deseasonalized noised-free SST.

## B.2 Cloud cover to simulate SST blurring

In Section **5.4.1** we described a blurring noise to emulate errors of the OI process when clouds are present. It involves a cloud cover $C$ retrieved from real SST products from which we construct our own. In Figure **B.1** we plot the cloud cover at different stages of the process. Image A shows the NRT L3 product [CMEMS, 2023c] cloud mask, image B this same image but bi-linearly interpolated to the resolution of the OSSE, image C shows image B thresholded, and image D shows this image after an average filter with a kernel size approximately equal to $43$ (km). In Figure **B.2**, we show the blurring noises obtained using cloud cover from images C and D. When using a binary mask as a cloud cover, patches appear in the SST images due to the

quick transition between blurred and unblurred areas. In this thesis we decided to use smooth cloud covers, with values between 0 and 1, to remove these artifacts.

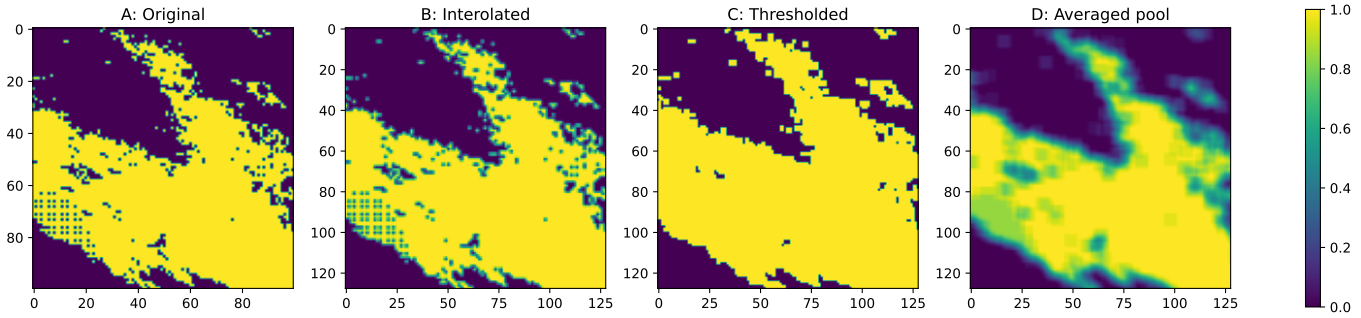


**Fig. B.1.:** Cloud cover $C$ at different stage of the process.

## B.3 Subsetting ablation study

To train the neural network using only observations, we introduced in Section **5.4.2** several ways to subset input and output observation of SSH, so that the network is controlled on data that it cannot access in its input. In the following, we present an ablation study in the situation where we do not perform this subsetting. We see in Table **B.2** that if no data is withdrawn from the inputs, it leads to bad reconstructions, as the network is able to copy its input on its output without performing an interpolation task.

| Loss | subset | SSH | SSH+nSST | SSH+SST |
|---|---|---|---|---|
| $\mathcal{L}_{\text{unsup}}$ | 1-sat | 4.56 \| 4.20 | 3.84 \| 3.49 | 3.56 \| 3.16 |
| $\mathcal{L}_{\text{unsup}}$ | mask | 3.89 \| 3.66 | 3.43 \| 3.19 | 3.18 \| 2.92 |
| $\mathcal{L}_{\text{unsup}}$ | none | 14.24 \| 13.65 | 12.08\| 11.59 | 13.23 \| 12.74 |

**Tab. B.2.:** SSH reconstruction RMSE in centimeters (mean score on the left and ensemble score on the right) of 3 ABED networks. We compare the unsupervised loss, with 3 subsetting modes: *1-sat*, where we retrieve the observations from one satellite from the inputs, *mask* where we apply random mask to the input, and *none* where all the data in inputs are used for control.

## B.4 Impact of the OSSE temporal length on training

The OSSE dataset that we introduce in Section **5.4** is composed of 7194 days, which leads to 5504 training days once the partition between train, validation, and test

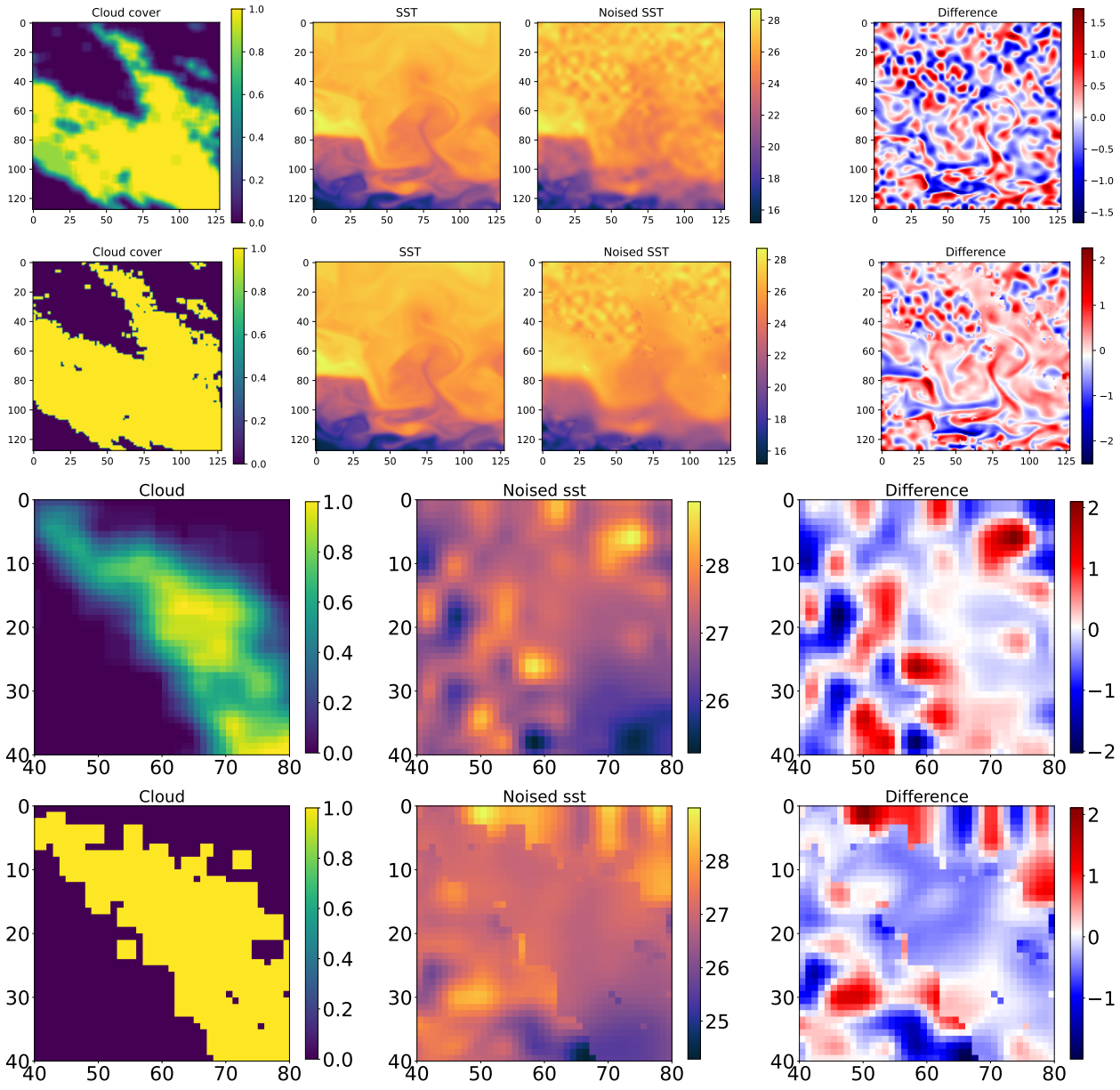


**Fig. B.2.:** Comparison of the SST noise with different cloud cover. The first and second raw represent the noise SST obtained using not binary and binary cloud maps, respectively. The last two rows are zoomed on particular areas to highlight the differences.

sets is made. To evaluate the interest in using more data to constrain the neural network, we train ABED network in the optimal configuration (supervised and using noise-free SST). We compare the scenario where all the samples are seen during training with those where only half, a quarter, or a single year of the dataset is used. The validation and test sets remain unchanged, while the training subset is the first consecutive days from the initial training set. Table **B.3** presents the RMSE of the reconstructions on the test year of our OSSE. The scores of the networks trained with different dataset sizes clearly show better reconstruction performance when the size increases.

| Size | Full | 1/2 | 1/4 | 1 year |
|---|---|---|---|---|
| Number of training samples | 5504 | 2752 | 1376 | 365 |
| RMSE (cm) | 2.97 | 3.97 | 4.77 | 7.76 |

**Tab. B.3.:** Mean RMSE score (in cm) of 3 ABED networks trained on our OSSE in a supervised manner using SSH and noise-free SST. We compare the situation where the full, half, a quarter, or one year of the dataset is used.

## B.5 Our OSSE against ODC2020 for neural network training

In Section **5.1.2** we describe previously existing OSSE, the ODC2020 dataset. Here, we compare the generalization to real satellite data of models trained on our OSSE with models trained on the ODC2020. As this last dataset provides one year of data, it can also be used to fit neural networks, but as shown in Appendix **B.4**, training on a longer dataset drastically improves reconstructions. As the existing OSSE does not provide SST data, it is possible to use NATL60 SST, but the lack of realistic noise leads to a domain gap with real data. To this day, if SSH-only neural networks have been successfully transferred to real SSH data, this is not the case for SST-aware ones. We compare ABED trained in a supervised way on our OSSE (SSH-only or using noisy SST), and on the ODC2020 (SSH-only or with NATL60 SST output). To train ABED on NATL60 data, we regrid the input and target data to our resolution, and use the data split of the challenge [CLS/MEOM, 2020]; validation of the training between 2012/10/22 and 2012/12/02, and fitting on the remaining days. We use the same hyperparameters as for the training on our OSSE. Once networks are trained on the simulation, we perform inferences on real data, excluding the tracks from the independent satellite. In Table **B.4**, we present the mean and ensemble scores of the models on the ODC2021.

| Method | Training OSSE data | $\mu(cm)$ | $\sigma_t(cm)$ | $\lambda_x(km)$ |
|---|---|---|---|---|
| ABED-SSH | ODC2020 | 8.90 \| 8.50 | 3.18 \| 3.10 | 148 \| 143 |
| ABED-SSH-SST | ODC2020 | 10.11 \| 9.73 | 3.38 \| 3.30 | 142 \| 137 |
| ABED-SSH | Ours | 6.63 \| 6.35 | 2.02 \| 1.90 | 122 \| 119 |
| ABED-SSH-SST | Ours | 6.28 \| 6.06 | 1.77 \| 1.73 | 115 \| 113 |

**Tab. B.4.:** Comparison of ABED networks trained on our OSSE to the ones trained on the Ocean Data Challenge 2020. All the metrics are computed on independent real data of the Ocean Data Challenge 2021. The left scores are the mean performances on three networks and the right ones are the ensemble scores.

As expected, ABED performs significantly better when trained on our OSSE. Specifically, ABED-SSH-SST trained on the ODC2020 leads to higher errors than its SSH-only version, which shows the domain gap between NATL60 and satellite SST. We conclude that the length of our OSSE and the addition of SST realistic noise enhanced the reconstructions of the real-world SSH.

## B.6 Along-track spatial derivatives

In Section **5.4.2** we present a regularization method introduced by [Martin et al., 2023], using the along-track derivative of SSH. In the following we describe the procedure to approximate these derivatives. We calculate the SSH's first and second spatial derivatives along the satellite ground track as described in Equation **B.1** and **B.2**. Given $\mathbf{Y}^{\text{ssh}}$, the list of SSH measurements from one satellite (sorted in time), we approximate the derivative by the rate of change of the SSH:

$$\frac{\partial}{\partial s}\mathbf{Y}^{\text{ssh}}_{\text{i}} \simeq \frac{\mathbf{Y}^{\text{ssh}}_{\text{i+1}} - \mathbf{Y}^{\text{ssh}}_{\text{i}}}{\triangle s_i} \tag{B.1}$$

$$\frac{\partial^2}{\partial s^2}\mathbf{Y}^{\text{ssh}}_{\text{i}} \simeq \frac{\frac{\partial}{\partial s}\mathbf{Y}^{\text{ssh}}_{\text{i+1}} - \frac{\partial}{\partial s}\mathbf{Y}^{\text{ssh}}_{\text{i}}}{\triangle s'_i} \tag{B.2}$$

# Bibliography

[Ahmed et al., 2010] Ahmed, N. K., Atiya, A. F., Gayar, N. E., and El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29:594–621.

[Ajayi et al., 2020] Ajayi, A., Sommer, J. L., Chassignet, E., Molines, J. M., Xu, X., Albert, A., and Cosme, E. (2020). Spatial and temporal variability of the north atlantic eddy field from two kilometric-resolution ocean models. *Journal of Geophysical Research: Oceans*, 125:e2019JC015827.

[Albawi et al., 2017] Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*, 2018-January:1–6.

[Amores et al., 2018] Amores, A., Jordà, G., Arsouze, T., and Le Sommer, J. (2018). Up to what extent can we characterize ocean eddies using present-day gridded altimetric products? *Journal of Geophysical Research: Oceans*, 123:7220–7236.

[Amores et al., 2017] Amores, A., Melnichenko, O., and Maximenko, N. (2017). Coherent mesoscale eddies in the North Atlantic subtropical gyre: 3-D structure and transport with application to the salinity maximum. *Journal of Geophysical Research: Oceans*, 122:23–41.

[Archambault, 2024] Archambault, T. (2024). 20 years OSSE and OSE of SSH and SST data in the Gulf Stream area [dataset]. `https://doi.org/10.5281/zenodo.10551897`.

[Archambault et al., 2022] Archambault, T., Charantonis, A., Béréziat, D., and Thiria, S. (2022). Sea surface height super-resolution using high-resolution sea surface temperature with a subpixel convolutional residual network. In *Environmental Data Science*.

[Archambault et al., 2023] Archambault, T., Filoche, A., Charantonis, A., and Béréziat, D. (2023). Multimodal unsupervised spatio-temporal interpolation of satellite ocean altimetry maps. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, Lisboa, Portugal.

[Archambault et al., 2024a] Archambault, T., Filoche, A., Charantonis, A., and Béréziat, D. (2024a). Pre-training and fine-tuning attention based encoder decoder improves sea surface height multi-variate inpainting. In *VISAPP*, Rome, Italy.

[Archambault et al., 2024b] Archambault, T., Filoche, A., Charantonis, A., Béréziat, D., and Thiria, S. (2024b). Learning sea surface height interpolation from multi-variate simulated satellite observations. *Journal of Advances of Modelling Earth Systems*, 16(6).

[Ardhuin et al., 2020] Ardhuin, F., Ubelmann, C., Dibarboure, G., Gaultier, L., Ponte, A., Ballarotta, M., and Faugère, Y. (2020). Reconstructing ocean surface current combining altimetry and future spaceborne doppler data. Earth and Space Science Open Archive.

[Arhan and Verdiére, 1985] Arhan, M. and Verdiére, A. C. D. (1985). Dynamics of eddy motions in the eastern north atlantic. *Journal of Physical Oceanography*, 15(2):153 – 170.

[Asch et al., 2016] Asch, M., Bocquet, M., and Nodet, M. (2016). *Data assimilation: methods, algorithms, and applications*. Society for Industrial and Applied Mathematics, Fundamentals of Algorithms.

[Auroux and Blum, 2008] Auroux, D. and Blum, J. (2008). A nudging-based data assimilation method: the back and forth nudging (BFN) algorithm. *Nonlin. Processes Geophys*, 15:305–319.

[Baldi, 2012] Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. In Guyon, I., Dror, G., Lemaire, V., Taylor, G., and Silver, D., editors, *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, volume 27 of *Proceedings of Machine Learning Research*, pages 37–49, Bellevue, Washington, USA. PMLR.

[Ballarotta et al., 2019] Ballarotta, M., Ubelmann, C., Pujol, M.-I., Taburet, G., Fournier, F., Legeais, J.-F., Faugère, Y., Delepoulle, A., Chelton, D., Dibarboure, G., and Picot, N. (2019). On the resolutions of ocean altimetry maps. *Ocean Science*, 15:1091–1109.

[Ballarotta et al., 2020] Ballarotta, M., Ubelmann, C., Rogé, M., Fournier, F., Faugère, Y., Dibarboure, G., Morrow, R., and Picot, N. (2020). Dynamic mapping of along-track ocean altimetry: Performance from real observations. *Journal of Atmospheric and Oceanic Technology*, 37:1593–1601.

[Barale et al., 2010] Barale, V., Gower, J. F., and Alberotanza, L. (2010). Oceanography from space: Revisited. *Oceanography from Space: Revisited*, pages 1–375.

[Battaglia et al., 2018] Battaglia, P., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Çaglar Gülçehre, Song, H. F., Ballard, A. J., Gilmer, J., Dahl, G. E., Vaswani, A., Allen, K. R., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. *arXiv.org*.

[Bengio et al., 1994] Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

[Bertino et al., 2003] Bertino, L., Evensen, G., and Wackernagel, H. (2003). Sequential data assimilation techniques in oceanography. *International Statistical Review*, 71(2):223–241.

[Bottou, 2010] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. *Proceedings of COMPSTAT 2010 - 19th International Conference on Computational Statistics, Keynote, Invited and Contributed Papers*, pages 177–186.

[Bretherton et al., 1976] Bretherton, F., Davis, R., and Fandry, C. (1976). A technique for objective analysis and design of oceanographic experiments applied to MODE-73. *Deep-Sea Research and Oceanographic Abstracts*, 23:559–582.

[Carrassi et al., 2018] Carrassi, A., Bocquet, M., Bertino, L., and Evensen, G. (2018). Data assimilation in the geosciences: An overview of methods, issues, and perspectives. *Wiley Interdisciplinary Reviews: Climate Change*, 9:e535.

[Che et al., 2022] Che, H., Niu, D., Zang, Z., Cao, Y., and Chen, X. (2022). ED-DRAP: Encoder–decoder deep residual attention prediction network for radar echoes. *IEEE Geoscience and Remote Sensing Letters*, 19.

[Chelton et al., 2011a] Chelton, D. B., Gaube, P., Schlax, M. G., Early, J. J., and Samelson, R. M. (2011a). The influence of nonlinear mesoscale eddies on near-surface oceanic chlorophyll. *Science*, 334:328–332.

[Chelton et al., 2001] Chelton, D. B., Ries, J. C., Haines, B. J., Fu, L.-L., and Callahan, P. S. (2001). Chapter 1 satellite altimetry. In *Satellite Altimetry and Earth Sciences*, volume 69 of *International Geophysics*, pages 1–ii. Academic Press.

[Chelton et al., 2011b] Chelton, D. B., Schlax, M. G., and Samelson, R. M. (2011b). Global observations of nonlinear mesoscale eddies. *Progress in Oceanography*, 91:167–216.

[Chen et al., 2023] Chen, L., Zhong, X., Zhang, F., Cheng, Y., Xu, Y., Qi, Y., and Li, H. (2023). Fuxi: a cascade machine learning forecasting system for 15-day global weather forecast. *Climate and Atmospheric Science 2023 6:1*, 6:1–11.

[Chiba et al., 2019] Chiba, S., Roquet, F., Jochum, M., Fox-Kemper, B., Adcroft, A., Böning, C. W., Chassignet, E. P., Curchitser, E., Danabasoglu, G., Eden, C., England, M. H., Gerdes, R., Greatbatch, R. J., Griffies, S. M., Hallberg, R. W., Hanert, E., Heimbach, P., Hewitt, H. T., Hill, C. N., Komuro, Y., Legg, S., Sommer, J. L., Masina, S., Marsland, S. J., Penny, S. G., Qiao, F., Ringler, T. D., Treguier, A. M., Tsujino, H., Uotila, P., and Yeager, S. G. (2019). Challenges and prospects in ocean circulation models. *Frontiers in Marine Science | www.frontiersin.org*, 6:65.

[Chin et al., 2017] Chin, T. M., Vazquez-Cuervo, J., and Armstrong, E. M. (2017). A multi-scale high-resolution analysis of global sea surface temperature. *Remote Sensing of Environment*, 200:154–169.

[Ciani et al., 2024] Ciani, D., Asdar, S., and Nardelli, B. B. (2024). Improved surface currents from altimeter-derived and sea surface temperature observations: Application to the north atlantic ocean. *Remote Sensing 2024, Vol. 16, Page 640*, 16:640.

[Ciani et al., 2020] Ciani, D., Rio, M.-H., Bruno Nardelli, B., Etienne, H., and Santoleri, R. (2020). Improving the altimeter-derived surface currents using sea surface temperature (SST) data: A sensitivity study to SST products. *Remote Sensing*, 12:1601.

[CLS/MEOM, 2020] CLS/MEOM (2020). SWOT data challenge NATL60 [dataset]. `https://doi.org/10.24400/527896/a01-2020.002`.

[CLS/MEOM, 2021] CLS/MEOM (2021). Data challenge OSE - 2021A_SSH_MAPPING_OSE [dataset]. `https://doi.org/10.24400/527896/a01-2021.005`.

[CMEMS, 2020] CMEMS (2020). Global ocean physics reanalysis [dataset]. `https://doi.org/10.48670/moi-00021`.

[CMEMS, 2021] CMEMS (2021). Global ocean along-track L3 sea surface heights reprocessed (1993-ongoing) tailored for data assimilation [dataset]. https://doi.org/10.48670/MOI-00146.

[CMEMS, 2023a] CMEMS (2023a). Global ocean gridded L4 sea surface heights and derived variables reprocessed 1993 ongoing. https://doi.org/10.48670/moi-00148.

[CMEMS, 2023b] CMEMS (2023b). Global ocean ostia sea surface temperature and sea ice analysis [dataset]. https://doi.org/10.48670/moi-00168.

[CMEMS, 2023c] CMEMS (2023c). Global oceans sea surface temperature multi-sensor L3 observations [dataset]. https://doi.org/10.48670/MOI-00164.

[Cohen and Shashua, 2017] Cohen, N. and Shashua, A. (2017). Inductive bias of deep convolutional networks through pooling geometry. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*.

[Cong et al., 2022] Cong, Y., Khanna, S., Meng, C., Liu, P., Rozi, E., He, Y., Burke, M., Lobell, D. B., and Ermon, S. (2022). Satmae: Pre-training transformers for temporal and multi-spectral satellite imagery.

[Cortes and Mohri, 2011] Cortes, C. and Mohri, M. (2011). Domain adaptation in regression. *International Conference on Algorithmic Learning Theory*, 6925 LNAI:308–323.

[Dimet and Talagrand, 1986] Dimet, F.-X. L. and Talagrand, O. (1986). Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Tellus A: Dynamic Meteorology and Oceanography*, 38(2):97–110.

[Dong et al., 2014] Dong, C., Loy, C., He, K., and Tang, X. (2014). Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision (ECCV)*, pages 184–199.

[Donlon et al., 2012] Donlon, C. J., Martin, M., Stark, J., Roberts-Jones, J., Fiedler, E., and Wimmer, W. (2012). The operational sea surface temperature and sea ice analysis (OSTIA) system. *Advanced Along Track Scanning Radiometer (AATSR), Special Issue of Remote Sensing of Environment*, 116:140–158.

[Dosovitskiy et al., 2021] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*.

[Dussin and Barnier, 2014] Dussin, R. and Barnier, B. (2014). The making of the DRAKKAR FORCING SET DFS5.

[Emery et al., 1989] Emery, W. J., Brown, J., and Nowak, Z. P. (1989). AVHRR image navigation-summary and review. *Photogrammetric engineering and remote sensing*, 4:1175–1183.

[Fablet et al., 2021] Fablet, R., Amar, M., Febvre, Q., Beauchamp, M., and Chapron, B. (2021). End-to-end physics-informed representation learning for satellite ocean remote sensing data: Applications to satellite altimetry and sea surface currents. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 5:295–302.

[Fablet et al., 2023] Fablet, R., Febvre, Q., and Chapron, B. (2023). Multimodal 4DVarNets for the reconstruction of sea surface dynamics from SST-SSH synergies. *IEEE Transactions on Geoscience and Remote Sensing*, 61.

[Farahani et al., 2021] Farahani, A., Voghoei, S., Rasheed, K., and Arabnia, H. R. (2021). A brief review of domain adaptation. pages 877–894.

[Fefferman et al., 2016] Fefferman, C., Mitter, S., and Narayanan, H. (2016). Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29:983–1049.

[Feichtenhofer et al., 2022] Feichtenhofer, C., Li, Y., He, K., et al. (2022). Masked autoencoders as spatiotemporal learners. *Advances in neural information processing systems*, 35:35946–35958.

[Filoche et al., 2022] Filoche, A., Archambault, T., Charantonis, A., and Béréziat, D. (2022). Statistics-free interpolation of ocean observations with deep spatio-temporal prior. In *ECML/PKDD Workshop on Machine Learning for Earth Observation and Prediction (MACLEAN)*.

[Fjørtoft et al., 2010] Fjørtoft, R., Gaudin, J.-M., Pourthie, N., Lion, C., Mallet, A., Souyris, J.-C., Ruiz, C., Koudogbo, F., Duro, J., Ordoqui, P., and Arnaud, A. (2010). Karin - the ka-band radar interferometer on SWOT: Measurement principle, processing and data specificities. In *2010 IEEE International Geoscience and Remote Sensing Symposium*, pages 4823–4826.

[Fjørtoft et al., 2014] Fjørtoft, R., Gaudin, J.-M., Pourthié, N., Lalaurie, J.-C., Mallet, A., Nouvel, J.-F., Martinot-Lagarde, J., Oriot, H., Borderies, P., Ruiz, C., and Daniel, S. (2014). Karin on SWOT: Characteristics of near-nadir ka-band

interferometric sar imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 52(4):2172–2185.

[Gaultier et al., 2016] Gaultier, L., Ubelmann, C., and Fu, L. (2016). The challenge of using future SWOT data for oceanic field reconstruction. *Journal of Atmospheric and Oceanic Technology*, 33:119–126.

[Gers et al., 2000] Gers, F. A., Schmidhuber, J., and Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *undefined*, 12:2451–2471.

[Gille et al., 2015] Gille, S. T., Metzger, E. J., and Tokmakian, R. (2015). Seafloor topography and ocean circulation. *Oceanography*, 17:47–54.

[Glorot and Bengio, 2010] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[Goodfellow et al., 2014] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27.

[Greff et al., 2017] Greff, K., Srivastava, R. K., Koutnik, J., Steunebrink, B. R., and Schmidhuber, J. (2017). Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28:2222–2232.

[Guo et al., 2021] Guo, M.-H., Xu, T.-X., Liu, J.-J., Liu, Z.-N., Jiang, P.-T., Mu, T.-J., Zhang, S.-H., Martin, R. R., Cheng, M.-M., and Hu, S.-M. (2021). Attention mechanisms in computer vision: A survey. *Computational Visual Media*, 8:331–368.

[Hadamard, 1924] Hadamard, J. (1924). Lectures on cauchy's problem in linear partial differential equations. by j. hadamard. pp. viii+316. 15s.net. 1923. *The Mathematical Gazette*, 12(171):173–174.

[Han et al., 2018] Han, Y., Yoo, J., Kim, H. H., Shin, H. J., Sung, K., and Ye, J. C. (2018). Deep learning with domain adaptation for accelerated projection-reconstruction mr. *Magnetic Resonance in Medicine*, 80:1189–1205.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[Hecht-Nielsen, 1992] Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. *Neural Networks for Perception*, pages 65–93.

[Henderson, 1975] Henderson, A. C. R. (1975). Best linear unbiased estimation and prediction under a selection model. *Biometrics*, 31:423.

[Hinton and Dean, 2015] Hinton, G. and Dean, J. (2015). Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

[Ho et al., 2020] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9:1735–1780.

[Isern-Fontanet et al., 2006] Isern-Fontanet, J., Chapron, B., Lapeyre, G., and Klein, P. (2006). Potential use of microwave sea surface temperatures for the estimation of ocean currents. *Geophys. Res. Lett*, 33:24608.

[Jam et al., 2021] Jam, J., Kendrick, C., Walker, K., Drouard, V., Hsu, J., and Yap, M. (2021). A comprehensive review of past and present image inpainting methods. *Computer Vision and Image Understanding*, 203.

[Jayne and Marotzke, 2002] Jayne, S. and Marotzke, J. (2002). The oceanic eddy heat transport. *Journal of Physical Oceanography*, 32:3328–3345.

[Kahru et al., 2012] Kahru, M., Di Lorenzo, E., Manzano-Sarabia, M., and Mitchell, B. G. (2012). Spatial and temporal statistics of sea surface temperature and chlorophyll fronts in the California Current. *Journal of Plankton Research*, 34(9):749–760.

[Kang et al., 2016] Kang, D., Curchitser, E. N., and Rosati, A. (2016). Seasonal variability of the Gulf Stream kinetic energy. *Journal of Physical Oceanography*, 46(4):1189 – 1207.

[Kawar et al., 2022] Kawar, B., Elad, M., Ermon, S., and Song, J. (2022). Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 35:23593–23606.

[Khan et al., 2022] Khan, S., Naseer, M., Khan, S., Naseer, M., City, M., Dhabi, A., Zamir, S. W., Shah, M., Hayat, M., Zamir, S. W., Khan, F. S., and Shah, M. (2022). Transformers in vision: A survey; transformers in vision: A survey. *ACM Computing Surveys*, 54.

[Kilpatrick et al., 2001] Kilpatrick, K. A., Podestá, G. P., and Evans, R. (2001). Overview of the NOAA/NASA advanced very high resolution radiometer pathfinder algorithm for sea surface temperature and associated matchup database. *Journal of Geophysical Research: Oceans*, 106:9179–9197.

[Kim et al., 2016] Kim, J., Lee, J., and Lee, K. (2016). Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

[Kolides et al., 2023] Kolides, A., Nawaz, A., Rathor, A., Beeman, D., Hashmi, M., Fatima, S., Berdik, D., Al-Ayyoub, M., and Jararweh, Y. (2023). Artificial intelligence foundation and pre-trained models: Fundamentals, applications, opportunities, and social impacts. *Simulation Modelling Practice and Theory*, 126:102754.

[Kugusheva et al., 2024] Kugusheva, A., Bull, H., Moschos, E., Ioannou, A., Le Vu, B., and Stegner, A. (2024). Ocean satellite data fusion for high-resolution surface current maps. *Remote Sensing*, 16(7).

[Lam et al., 2023] Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., Merose, A., Hoyer, S., Holland, G., Vinyals, O., Stott, J., Pritzel, A., Mohamed, S., and Battaglia, P. (2023). Learning skillful medium-range global weather forecasting. *Science (New York, N.Y.)*, 382:1416–1422.

[Lapeyre, 2017] Lapeyre, G. (2017). Surface quasi-geostrophy. *Fluids*.

[Le Guillou et al., 2023] Le Guillou, F., Gaultier, L., Ballarotta, M., Metref, S., Ubelmann, C., Cosme, E., and Rio, M.-H. (2023). Regional mapping of energetic short mesoscale ocean dynamics from altimetry: performances from real observations. *Ocean Science*, 19(5):1517–1527.

[Le Guillou et al., 2020] Le Guillou, F., Metref, S., Cosme, E., Ubelmann, C., Ballarotta, M., Verron, J., and Le Sommer, J. (2020). Mapping altimetry in the forthcoming SWOT era by back-and-forth nudging a one-layer quasi-geostrophic model. Earth and Space Science Open Archive.

[Le Provost, 2001] Le Provost, C. (2001). Chapter 6 ocean tides. In Fu, L.-L. and Cazenave, A., editors, *Satellite Altimetry and Earth Sciences*, volume 69 of *International Geophysics*, pages 267–303. Academic Press.

[Lecun et al., 2015] Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444.

[Ledig et al., 2017] Ledig, C., Theis, L., Huszár, H., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., and Wang, Z. (2017). Photo-realistic single image super-resolution using a generative adversarial network. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4681–4690.

[Lellouche et al., 2018] Lellouche, J.-M., Greiner, E., Le Galloudec, O., Regnier, C., Benkiran, M., Testut, C.-E., Bourdalle-Badie, R., Drevillon, M., Garric, G., and Drillet, Y. (2018). Mercator ocean global high-resolution monitoring and forecasting system. *New Frontiers in Operational Oceanography*, pages 563–592.

[Lim et al., 2017] Lim, B., Son, S., Kim, H., Nah, S., and Lee, K. (2017). Enhanced deep residual networks for single image super-resolution. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1132–1140.

[Lugmayr et al., 2022] Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., and Gool, L. V. (2022). Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2022-June, pages 11451–11461. IEEE Computer Society.

[Madec et al., 2017] Madec, G., Bourdallé-Badie, R., Bouttier, P.-A., Bricaud, C., Bruciaferri, D., Calvert, D., Chanut, J., Clementi, E., Coward, A., Delrosso, D., et al. (2017). NEMO ocean engine. Scientific Notes of Climate Modelling Center.

[Mahesh, 2018] Mahesh, B. (2018). Machine learning algorithms-a review. *International Journal of Science and Research*.

[Manzella et al., 2022] Manzella, G. M., De Strobel, F., Pinardi, N., and Emery, W. (2022). Chapter one - a narrative of historical, methodological, and technological observations in marine science. In Manzella, G. and Novellino, A., editors, *Ocean Science Data*, pages 3–64. Elsevier.

[Martin, 2014] Martin, S. (2014). *An Introduction to Ocean Remote Sensing*. Cambridge University Press, 2 edition.

[Martin et al., 2023] Martin, S. A., Manucharyan, G. E., and Klein, P. (2023). Synthesizing sea surface temperature and satellite altimetry observations using deep

learning improves the accuracy and resolution of gridded sea surface height anomalies. *Journal of Advances in Modeling Earth Systems*, 15(5).

[Mirza and Osindero, 2014] Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv.org*.

[Mkhinini et al., 2014] Mkhinini, N., Coimbra, A. L. S., Stegner, A., Arsouze, T., Taupier-Letage, I., and Béranger, K. (2014). Long-lived mesoscale eddies in the eastern Mediterranean Sea: Analysis of 20 years of AVISO geostrophic velocities. *Journal of Geophysical Research: Oceans*, 119:8603–8626.

[Mohammed and Kora, 2023] Mohammed, A. and Kora, R. (2023). A comprehensive review on ensemble deep learning: Opportunities and challenges. *Journal of King Saud University - Computer and Information Sciences*, 35(2):757–774.

[Nardelli et al., 2022] Nardelli, B., Cavaliere, D., Charles, E., and Ciani, D. (2022). Super-resolving ocean dynamics from space with computer vision algorithms. *Remote Sensing*, 14:1159.

[NASA/JPL, 2019] NASA/JPL (2019). GHRSST level 4 MUR 0.25 deg global foundation sea surface temperature analysis (v4.2) [dataset]. `https://doi.org/10.5067/GHGMR-4FJ04`.

[NASA/JPL and CNES, 2024] NASA/JPL and CNES (2024). The SWOT_L3_LR_SSH product, derived from the L2 SWOT KaRIn low rate ocean data products [Dataset]. `https://doi.org/10.24400/527896/A01-2023.018`.

[Odena et al., 2016] Odena, A., Dumoulin, V., and Olah, C. (2016). Deconvolution and checkerboard artifacts. *Distill*, 1.

[Ongie et al., 2020] Ongie, O., Jalal, A., Metzler, C., Baraniuk, R., Dimakis, A., and Willett, R. (2020). Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1:39–56.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

[Patel and Oberai, 2021] Patel, D. V. and Oberai, A. A. (2021). GAN-based priors for quantifying uncertainty in supervised learning. *SIAM/ASA Journal on Uncertainty Quantification*, 9(3):1314–1343.

[Patel et al., 2015] Patel, V. M., Gopalan, R., Li, R., and Chellappa, R. (2015). Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32:53–69.

[Patwardhan et al., 2023] Patwardhan, N., Marrone, S., and Sansone, C. (2023). Transformers in the real world: A survey on NLP applications. *Information 2023, Vol. 14, Page 242*, 14:242.

[Peral and Esteban-Fernandez, 2018] Peral, E. and Esteban-Fernandez, D. (2018). SWOT mission performance and error budget. *International Geoscience and Remote Sensing Symposium (IGARSS)*, 2018-July:8625–8628.

[Piterbarg, 2009] Piterbarg, L. I. (2009). A simple method for computing velocities from tracer observations and a model output. *Applied Mathematical Modelling*, 33:3693–3704.

[Popescu et al., 2009] Popescu, M.-C., Balas, V. E., Perescu-Popescu, L., and Mastorakis, N. (2009). Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8(7):579–588.

[Pujol et al., 2016] Pujol, M.-I., Faugère, Y., Taburet, G., Dupuy, S., Pelloquin, C., Ablain, M., and Picot, N. (2016). DUACS DT2014: the new multi-mission altimeter data set reprocessed over 20 years. *Ocean Sci*, 12:1067–1090.

[Pujol et al., 2018] Pujol, M.-I., Schaeffer, P., Faugère, Y., Raynal, M., Dibarboure, G., and Picot, N. (2018). Gauging the improvement of recent mean sea surface models: A new approach for identifying and quantifying their errors. *Journal of Geophysical Research: Oceans*, 123(8):5889–5911.

[Puntanen and Styan, 1989] Puntanen, S. and Styan, G. P. H. (1989). The equality of the ordinary least squares estimator and the best linear unbiased estimator. 43:153–161.

[Qin et al., 2021] Qin, Z., Zeng, Q., Zong, Y., and Xu, F. (2021). Image inpainting based on deep learning: A review. *Displays*, 69:102028.

[Radford et al., 2016] Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*.

[Rahaman et al., 2019] Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 5301–5310. PMLR.

[Raven and Falkowski, 1999] Raven, J. A. and Falkowski, P. G. (1999). Oceanic sinks for atmospheric co2. *Plant, Cell & Environment*, 22(6):741–755.

[Rio and Santoleri, 2018] Rio, M. H. and Santoleri, R. (2018). Improved global surface currents from the merging of altimetry and sea surface temperature data. *Remote Sensing of Environment*, 216:770–785.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICAI*, pages 234–24.

[Rosen et al., 2000] Rosen, P., Hensley, S., Joughin, I., Li, F., Madsen, S., Rodriguez, E., and Goldstein, R. (2000). Synthetic aperture radar interferometry. *Proceedings of the IEEE*, 88(3):333–382.

[Ruthotto and Haber, 2021] Ruthotto, L. and Haber, E. (2021). An introduction to deep generative modeling. *GAMM-Mitteilungen*, 44:e202100008.

[Schuckmann et al., 2020] Schuckmann, K. V., Cheng, L., Palmer, M. D., Hansen, J., Tassone, C., Aich, V., Adusumilli, S., Beltrami, H., Boyer, T., Cuesta-Valero, F. J., Desbruyères, D., Domingues, C., Garciá-Garciá, A., Gentine, P., Gilson, J., Gorfer, M., Haimberger, L., Ishii, M., Johnson, G. C., Killick, R., King, B. A., Kirchengast, G., Kolodziejczyk, N., Lyman, J., Marzeion, B., Mayer, M., Monier, M., Monselesan, D. P., Purkey, S., Roemmich, D., Schweiger, A., Seneviratne, S. I., Shepherd, A., Slater, D. A., Steiner, A. K., Straneo, F., Timmermans, M. L., and Wijffels, S. E. (2020). Heat stored in the earth system: Where does the energy go? *Earth System Science Data*, 12:2013–2041.

[SEANOE, 2024] SEANOE (2024). Copernicus marine in situ - global ocean-delayed mode in situ observations of surface (drifters, HFR) and sub-surface (vessel-mounted ADCPs) water velocity. `https://doi.org/10.17882/86236`.

[Shi et al., 2017a] Shi, W., Caballero, J., Huszar, F., Totz, J., Aitken, A., Bishop, R., Rueckert, R., and Wang, Z. (2017a). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2016-Decem, pages 1874–1883.

[Shi et al., 2017b] Shi, W., Caballero, J., Theis, L., Huszar, F., Aitken, A., Tejani, A., Totz, J., Ledig, C., and Wang, Z. (2017b). Is the deconvolution layer the same as a convolutional layer? A note on Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network.

[Shi et al., 2015] Shi, X., Chen, Z., Wang, H., Yeung, D. Y., Wong, W. K., and Woo, W. C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems*, 2015-January:802–810.

[Singhal and Majumdar, 2020] Singhal, V. and Majumdar, A. (2020). A domain adaptation approach to solve inverse problems in imaging via coupled deep dictionary learning. *Pattern Recognition*, 100:107163.

[Stammer and Cazenave, 2017] Stammer, D. and Cazenave, A. (2017). *Satellite altimetry over oceans and land surfaces*. CRC Press.

[Stegner et al., 2021] Stegner, A., Le Vu, B., Dumas, F., Ghannami, M., Nicolle, A., Durand, C., and Faugere, Y. (2021). Cyclone-anticyclone asymmetry of eddy detection on gridded altimetry product in the mediterranean sea. *Journal of Geophysical Research: Oceans*, 126.

[Stewart, 2006] Stewart, R. H. (2006). *Introduction To Physical Oceanography*.

[Taburet et al., 2019] Taburet, G., Sanchez-Roman, A., Ballarotta, M., Pujol, M.-I., Legeais, J.-F., Fournier, F., Faugere, Y., and Dibarboure, G. (2019). DUACS DT2018: 25 years of reprocessed sea level altimetry products. *Ocean Sci*, 15:1207–1224.

[Tamir et al., 2019] Tamir, J. I., Yu, S. X., and Lustig, M. (2019). Unsupervised deep basis pursuit: Learning inverse problems without ground-truth data. *arXiv*.

[Thiria et al., 2023] Thiria, S., Sorror, C., Archambault, T., Charantonis, A., Béréziat, D., Mejia, C., Molines, J.-M., and Crepon, M. (2023). Downscaling of ocean fields by fusion of heterogeneous observations using deep learning algorithms. *Ocean Modeling*, 182.

[Thoppil et al., 2021] Thoppil, P. G., Frolov, S., Rowley, C. D., Reynolds, C. A., Jacobs, G. A., Metzger, E. J., Hogan, P. J., Barton, N., Wallcraft, A. J., Smedstad, O. M., and Shriver, J. F. (2021). Ensemble forecasting greatly expands the prediction horizon for ocean mesoscale variability. *Communications Earth & Environment 2021 2:1*, 2:1–9.

[Tonani et al., 2015] Tonani, M., Balmaseda, M. A., Bertino, L., Blockley, E. W., Brassington, G. B., Davidson, F. J. M., Drillet, Y., Hogan, P. J., Kuragano, T., Lee, T., Mehra, A., Paranathara, F., Tanajura, C. A. S., and Wang, H. (2015). Status and future of global and regional ocean prediction systems. *Journal of Operational Oceanography*, 8:s201 – s220.

[Touvron et al., 2022] Touvron, H., Cord, M., El-Nouby, A., Verbeek, J., and Jégou, H. (2022). Three things everyone should know about vision transformers. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 13684 LNCS:497–515.

[Tran et al., 2018] Tran, D., Wang, H., Torresani, L., Ray, J., Lecun, Y., and Paluri, M. (2018). A closer look at spatiotemporal convolutions for action recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 6450–6459.

[Traon et al., ] Traon, P. Y. L., Nadal, F., and Ducet, N. An improved mapping method of multisatellite altimeter data. *Journal of Atmospheric and Oceanic Technology*.

[Ubelmann et al., 2016] Ubelmann, C., Cornuelle, B., and Fu, L. (2016). Dynamic mapping of along-track ocean altimetry: Method and performance from observing system simulation experiments. *Journal of Atmospheric and Oceanic Technology*, 33:1691–1699.

[Ulyanov et al., 2017] Ulyanov, D., Vedaldi, A., and Lempitsky, V. (2017). Deep image prior. *International Journal of Computer Vision*, 128:1867–1888.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Łukasz Kaiser, and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 2017-December:5999–6009.

[Vu et al., 2018] Vu, B. L., Stegner, A., and Arsouze, T. (2018). Angular momentum eddy detection and tracking algorithm (AMEDA) and its application to coastal eddy formation. *Journal of Atmospheric and Oceanic Technology*, 35:739–762.

[Wang et al., 2024] Wang, X., Wang, R., Hu, N., Wang, P., Huo, P., Wang, G., Wang, H., Wang, S., Zhu, J., Xu, J., Yin, J., Bao, S., Luo, C., Zu, Z., Han, Y., Zhang, W., Ren, K., Deng, K., and Song, J. (2024). Xihe: A data-driven model for global ocean eddy-resolving forecasting. arXiv.

[Wang et al., 2018] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Loy, C. C., Qiao, Y., and Tang, X. (2018). Esrgan: Enhanced super-resolution generative adversarial networks. In *European Conference on Computer Vision (ECCV) Workshops*.

[Wang et al., 2021] Wang, Z., Chen, J., and Hoi, S. (2021). Deep learning for image super-resolution: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 3365–3387.

[Woo et al., 2018] Woo, S., Park, J., Lee, J.-Y., and Kweon, I. S. (2018). CBAM: Convolutional block attention module. *Computer Vision and Pattern Recognition*.

[Xiao et al., 2024] Xiao, H., Wang, X., Wang, J., Cai, J. Y., Deng, J. H., Yan, J. K., and Tang, Y. D. (2024). Single image super-resolution with denoising diffusion gans. *Scientific Reports 2024 14:1*, 14:1–18.

[Yaman et al., 2019] Yaman, B., Hosseini, S. A. H., Moeller, S., Ellermann, J., Uğurbil, K., and Akçakaya, M. (2019). Self-supervised learning of physics-guided reconstruction neural networks without fully sampled reference data. *Magnetic Resonance in Medicine*, 84:3172–3191.

[Yang et al., 2013] Yang, J., Gong, P., Fu, R., Zhang, M., Chen, J., Liang, S., Xu, B., Shi, J., and Dickinson, R. (2013). The role of satellite remote sensing in climate change studies. *Nature Climate Change 2013 3:10*, 3:875–883.

[Yang et al., 2023] Yang, L., Zhang, Z., Zhao, Y., Yang, M.-H., Song, Y., Hong, S., Xu, R., Zhang, W., Cui, B., and Yang, M.-H. (2023). Diffusion models: A comprehensive survey of methods and applications. *Comprehensive Survey of Methods and Applications. ACM Comput. Surv*, 1:105.

[Yu et al., 2019] Yu, Y., Si, X., Hu, C., and Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31:1235–1270.

[Zeng et al., 2020] Zeng, X., Atlas, R., Birk, R. J., Carr, F. H., Carrier, M. J., Cucurull, L., Hooke, W. H., Kalnay, E., Murtugudde, R., Posselt, D. J., Russell, J. L., Tyndall, D. P., Weller, R. A., and Zhang, F. (2020). Use of observing system simulation experiments in the united states. *Bulletin of the American Meteorological Society*, 101(8):E1427 – E1438.

[Zhai et al., 2008] Zhai, X., Greatbatch, R. J., and Kohlmann, J.-D. (2008). On the seasonal variability of eddy kinetic energy in the Gulf Stream region. *Geophysical Research Letters*, 35(24).

[Zheng et al., 2020] Zheng, G., Li, X., Zhang, R. H., and Liu, B. (2020). Purely satellite data–driven deep learning forecast of complicated tropical instability waves. *Science Advances*, 6.