DINCAE 2: multivariate convolutional neural network with error estimates to reconstruct sea surface temperature satellite and altimetry observations

Alexander Barth[1], Aida Alvera-Azcárate[1], Charles Troupin[1], and Jean-Marie Beckers[1]

[1]GHER, University of Liège, Liège, Belgium

Correspondence: A. Barth (a.barth@uliege.be)

Abstract. DINCAE (Data INterpolating Convolutional Auto-Encoder) is a neural network to reconstruct missing data (e.g. obscured by clouds or gaps between tracks) in satellite data. Contrary to standard image reconstruction (in-painting) with neural networks, this application requires a method to handle missing data (or data with variable accuracy) already in the training phase. Instead of using a standard L2 (or L1) cost function, the neural network (U-Net type of network) is optimized by minimizing the negative log likelihood assuming a Gaussian distribution (characterized by a mean and a variance). As a consequence, the neural network also provides an expected error variance of the reconstructed field (per pixel and per time instance).

In this updated version DINCAE 2.0, the code was rewritten in Julia and a new type of skip connection has been implemented which showed superior performance with respect to the previous version. The method has also been extended to handle multivariate data (an example will be shown with sea-surface temperature, chlorophyll concentration and wind fields). The improvement of this network is demonstrated in the Adriatic Sea.

Convolutional networks work usually with gridded data as input. This is however a limitation for some data types used in oceanography and in Earth Sciences in general, where observations are often irregularly sampled. The first layer of the neural network and the cost function have been modified so that unstructured data can also be used as inputs to obtain gridded fields as output. To demonstrate this, the neural network is applied to along-track altimetry data in the Mediterranean Sea. Results from a 20-year reconstruction are presented and validated. Hyperparameters are determined using Bayesian optimization and minimizing the error relative to a development dataset.

1 Introduction

Ocean data are generally sparse and inhomogeneously distributed. The data coverage often contains large gaps in space and time. This is in particular the case of in situ observations. Satellite remote sensing only measures the surface of the ocean but generally has better spatial coverage than in situ observations. But still about 75% of the ocean surface is on average covered by clouds that block sensors in the optical and infrared bands (Wylie et al., 2005). Given the sparsity of data, it is natural to aim to combine data representing different parameters as e.g. mesoscale flow structures are often visible in all ocean tracers.

Prior work on using multivariate data in connection with satellite data use for example empirical orthogonal functions (EOF) ~~who~~ which can be naturally extended to multivariate datasets as long as an appropriate norm is defined. For example, ~~Alvera-Azcárate et al. (2007)~~ Alvera-Azcárate et al. (2007) uses sea surface temperature, chlorophyll, and wind satellite fields with Data Interpolating Empirical Orthogonal Functions (DINEOF). Multivariate EOFs have also been used to project surface

5 observations to ~~deepers layers Nardelli and Santoleri (2005)~~ deeper layers (Nardelli and Santoleri, 2005) or to derive nitrate maps in the Southern Ocean (Liang et al., 2018). In the latter case, EOFs linking salinity, and potential temperature and nitrate concentration are derived from model simulations.

As some observations can be measured at much high spatial resolution via remote sensing (in particular the resolution of sea-surface temperature is much higher than the resolution of sea surface salinity products), the so-called "multifractal fusion

10 techniques" are used to improve remote sensed surface salinity estimates using sea surface temperature. The data fusion is implemented as a locally weighted linear regression (Olmedo et al., 2018, 2021). Han et al. (2020) also used an earlier version of the DINCAE code to estimate sea-surface chlorophyll using additional sea surface temperature observations.

The structure of a neural network and in particular its depth is uncertain and to some degree dependent on the used data set. We also investigate the ~~use of deeper~~ influence of the depth of the neural networks in this work. It is known that ~~deeper~~ neural

15 networks are increasingly more difficult to train as their depth increases because of the well-known vanishing gradient problem (Hochreiter, 1998): the derivative of the loss function relative to the weights of the first layers of a neural network have the tendency to either decrease (or increase) exponentially with a increasing number of layers. This prevents effective optimization (training) of these layers using gradient-based optimization methods.

20 Several methods have been proposed in the literature to mitigate such problems using alternative neural network architectures. In the context of the present manuscript, skip connections in the form of residual layers have been tested (similar to residual networks, He et al., 2015). The derivative of the loss function relative to the weights in such layers remains (at least initially) closer to one, so there is a more direct relationship between the loss function and the weights and biases to be optimized. Deeper residual networks include shallower networks as a special case and there, per construction, should perform at

25 least as good as shallower networks.

Gradient of a whole network is computed via back-propagation which is essentially based on the repeated application of the chain rule for differentiation. The information of the observation is injected via the loss function and propagated backward in a way which is similar to the 4D-var backward in time integration of the adjoint model. Another interesting neural network

30 architecture has been proposed in the form the Inception network ~~Szegedy et al. (2015)~~ (Szegedy et al., 2015) where the output of intermediate layers, here in the form of a preliminary reconstruction, are used in the loss function (in addition to the output of the final layer). The result is that the information of the observations are injected not only at the final layer but also in the intermediate layer which also contributes to reducing the vanishing gradient problem.

While for gridded satellite data, approaches based on empirical orthogonal functions and convolutional neural networks have been shown successful, it is difficult to apply similar concepts to non-gridded data as these methods typically require a stationary grid. Another objective of this paper is to show how convolutional neural networks can be used on non-gridded data. This approach is illustrated with altimetry observations.

The objective of this manuscript is to highlight the improvement of DINCAE made relative to the previously published version ~~Barth et al. (2020)~~(Barth et al., 2020). The section 2 presents the updated structure of the neural network. The gridded and non-gridded observations used here are presented in section 3. Details of the implementation are also given (section 4). The results and conclusions are presented in sections 5 and 6.

2 The neural network architecture

The DINCAE network (Barth et al., 2020) is a neural network composed of an encoder and decoder network. The encoder uses the original gappy satellite data (with additional metadata as explained later) and the decoder uses the output of the encoder to reconstruct the full data image (along with an error estimate). The encoder uses a series of convolutional layers followed by max pooling layers reducing the resolution of the datasets. The decoder does essentially the reverse operation by using convolutional and interpolation layers. This is the general structure of a convolutional autoencoder and the classical U-Net networks (Ronneberger et al., 2015). In the following section we will discuss the main components of the DINCAE neural network used for the different test cases and emphasize the changes relative to the previous version.

2.1 Skip connections

In an autoencoder, the inputs are compressed by forcing the data flow through a bottleneck which ensures that the neural network must efficiently compress and decompress the information. However, in U-Net (Ronneberger et al., 2015) and DINCAE skip connections are implemented allowing the information flow of the network to partially bypass the bottleneck to prevent the loss of small scale details in the reconstruction. Skip connections can be realised either by concatenating tensors along the feature map dimension (as it is done in U-Net) or by summing the tensors. The cat-skip connections at the step l in the autoencoder can be written as the following operation:

$$\mathbf{X}^{(l+1)} = \mathrm{cat}(f^{(l)}(\mathbf{X}^{(l)}), \mathbf{X}^{(l)}) \tag{1}$$

where `cat` concatenates two 3-dimensional arrays along the dimension representing the features channels. The function $f^{(l)}$ is a sequence of neural network layers applied to the array of $\mathbf{X}^{(l)}$ (produced by previous network layers). Sum skip-connections are implemented as:

$$\mathbf{X}^{(l+1)} = f^{(l)}(\mathbf{X}^{(l)}) + \mathbf{X}^{(l)} \tag{2}$$

5 Clearly, the output of a cat-skip connection has a size twice as large as the output of a sum-skip connection. These skip connections are followed by a convolutional layer, which ensures that the number of output features are the same for both types of skip connection. In fact, one can show that the sum-skip connection (followed by a convolution layer) is formally a special case of the cat-skip connection. However, sum-skip connections can have an advantage because the weight and ~~basis~~ bias of the convolutional layers are more directly related to the output of the neural network which helps to reduce the so-called vanishing

10 gradient problem (He et al., 2015).

2.2 Refinement step

The whole neural network can be described as two functions that provide the input variable product between the reconstruction $\hat{y}$ and its expected error ~~error variance $\hat{\sigma}$~~ variance $\hat{\sigma}^2$ for every grid cell. The loss function is derived from the negative log-likelihood of a Gaussian with mean $\hat{y}$ and ~~standard deviation of σ~~ variance $\hat{\sigma}^2$:

15 $$J(\hat{y}_{ij}, \hat{\sigma}_{ij}{}^2) = \frac{1}{2N} \sum_{ij} \left[\left(\frac{y_{ij} - \hat{y}_{ij}}{\hat{\sigma}_{ij}} \right)^2 + \log(\hat{\sigma}_{ij}^2) + 2\log(\sqrt{2\pi})) \right] \tag{3}$$

where the sum with the spatial indices i and j runs over all grid points with valid (i.e. non-masked) values and N is the number of valid values. The first term of the right hand-side of the equation is the mean square error, but scaled by the estimated error standard deviation, the second term penalizes any over-estimation of the error standard deviation and the third term is constant and can be neglected as it does not influence the gradient of the cost function. For a convolutional auto-encoder

20 with refinement, the intermediate outputs $\hat{y}$ and $\hat{\sigma}$ are concatenated with the inputs and passed through another auto-encoder with the same structure (~~expect~~ except for the number of filters for the input layer which has to accommodate the two additional fields corresponding to $\hat{y}$ and $\hat{\sigma}$). The weights of the first and second auto-encoder are not related. The final cost function with refinement J_r is given by:

$$J_r = \alpha J(\hat{y}_{ij}, \hat{\sigma}_{ij}{}^2) + \alpha' J(\hat{y}'_{ij}, \hat{\sigma}'_{ij}{}^2) \tag{4}$$

where $\hat{y}'$ and $\hat{\sigma}'^2$ are the reconstruction and its expected error variance produced by the second auto-encoder. The weights α and α' control how much importance is given to the intermediate output (relative to the final output).

With a refinement step, the neural network becomes essentially twice as deep and the number of parameters (approximately) doubles. The increased depth would make it prone to the vanishing gradient problem. However, by including the intermediate results in the cost function this problem is reduced. In fact, the information from the observations is injected during back-propagation by the loss function. Due to the refinement step and the loss function which also depends on the intermediate result, the information from the observation is injected at the last layer and at the middle layer of the combined neural network (Szegedy et al., 2015). The relationship between the first layers of the neural network and the cost function is therefore more direct, which helps in the training of these first layers.

The refinement step has been used in image in-painting for computer vision application (Liu et al., 2019) and it has also been applied for oceanographic data for tide gauge data (Zhang et al., 2020). In the present work only one refinement step is tested, but the code supports an arbitrary number of sequential refinement steps.

2.3 Multivariate reconstructions

Auxiliary satellite data (with potentially missing data) can be provided for the reconstruction. The handling of missing data in these auxiliary data is identical to the way missing data are treated for the primary variable. For every auxiliary satellite data, the average over time is first removed. The auxiliary data (divided by its corresponding error variance) and the inverse of the error variance are provided as input. Where data is missing, the corresponding input values are set to zero representing an infinitely large error (as a consequence of the chosen scaling). Multiple time instances centered around a target time can be provided as input.

2.4 Non-gridded input data

Current satellite altimetry mission measures sea surface height along the ground track of the satellite. Satellite altimetry can measure through clouds but the data is only available along a collection of tracks. In order to handle better such data sets, we extended DINCAE to handle unstructured data as input.

The first layer in DINCAE is a convolutional layer which typically requires a field discretized on a rectangular grid. The convolutional layer can be seen as the discretized version of the following integral:

$$g(x,y) = \int_{\Omega_w} w(x - x', y - y') f(x,y) dx' dy' \tag{5}$$

where f is the input field, w are the weights in the convolution (also called convolution kernel) , Ω_w is the support of the function w (i.e. the domain where w is different from zero) and g the output of the convolutional layer. To discretize the integral, the continuous function f is replaced by a sum of Dirac functions using the values $f_{i,j}$ defined on a regular grid:

$$f(x,y) = \sum_{i,j} f_{i,j}\delta(x - i\Delta x, y - j\Delta y)$$

In this case, the continuous convolution becomes the standard discrete convolution as used in neural networks. The weights $w(x,y)$ need only to be known at the discrete locations defined by the underlying grid.

For data points which are not defined on a regular grid we essentially use a similar approach. The function f is again written as a sum of Dirac functions:

$$f(x,y) = \sum_{k} f_k \delta(x - x_k, y - y_k)$$

where now f_k are the values at the locations (x_k, y_k) which can be arbitrary. In order to evaluate the integral (5), it is necessary to know the weights at the location (x_k, y_k). The weights are still discretized on a regular grid, but are interpolated bilinearly to the data location to evaluate the integral. In fact, instead of interpolating the weights w, one can also apply the adjoint of the linear interpolation to f_k (which is mathematically equivalent). This has the benefit that the computation of the convolution can be implemented using the optimized functions in the neural network library.

For data defined on a regular grid, it has been verified numerically that this proposed approach and the traditional approach to compute the convolution give the same results.

3 Data

The improvements are examined on two test cases. For multivariate gridded data, the approach is tested with sea surface temperature, chlorophyll and winds on the Adriatic Sea, and for non-gridded data altimetry observations on the whole Mediterranean Sea was used. As the altimetry observations do not resolve as many small-scales as sea surface temperature, a larger domain was chosen for the altimetry test case.

3.1 Gridded data (Adriatic Sea)

As the previous application (Barth et al., 2020) considered relatively low resolution AVHRR data, we used more modern and higher resolution satellite data in this application for the Adriatic Sea. The datasets used include:

- Sea Surface Temperature (MODIS Terra Level 3 SST Thermal IR Daily 4km Nighttime v2014.0, doi:10.5067/MODST-1D4N4) made available by PO.DAAC (https://podaac.jpl.nasa.gov/, JPL, NASA, US)

- Wind speed (Cross-Calibrated Multi-Platform (CCMP) gridded surface vector winds) made available from Remote Sensing Systems (http://www.remss.com/measurements/ccmp/). These datasets are described in Atlas et al. (2011); Mears

et al. (2019); Wentz et al. (2019). This dataset has a 6 h temporal resolution and a 1/4 degree spatial resolution resolution. The wind fields are averaged as daily mean fields.

– Chlorophyll-a from Ocean Biology Processing Group, NASA (US) https://oceancolor.gsfc.nasa.gov/data/overview/ at a 4 km resolution and L3 processing level.

The data sets span the time period 2003-01-01 to 2016-12-31. They are all interpolated (using bi-linear interpolation) on the common grid defined by the SST fields.

As ocean mixing reacts to the averaged effect of the wind speed (norm of the wind vector), we also smoothed the speed with a Laplacian filter using a time period of 2.2 days and a lag of 4 days (wind speed preceding SST). The optimal lag and time period were obtained by maximizing the correlation between the smoothed wind field and SST from the training data.

3.2 Non-gridded data (Mediterranean Sea)

Altimetry data from 1993-01-01 to 2019-05-13 between 7°W and 37°E and 30°N to 46°N from 22 satellite missions operating during this time frame ~~is~~ are used. This domain contains essentially the Mediterranean Sea but also a small part of the Bay of Biscay and the Black Sea. In preliminary studies we found that including the data from adjacent seas can help a neural network to better ~~generalize~~ generalise and to prevent overfitting (*e.g.* Gong et al., 2019) as the neural network is confronted with a more diverse set of conditions. The data (SEALEVEL_EUR_PHY_L3_~~REP~~MY_OBSERVATIONS_008_061~~) is~~, accessed on 2020-10-13) are made available by the Copernicus Marine Environment Monitoring Service (CMEMS).

~~This data was~~ These data were split along the following fractions:

– 70% training data,

– 20% development data,

– 10% test data.

To reduce the correlation between the different datasets, satellite tracks are not splitted and belong entirely to one of these 3 datasets.

Some experiments of the reconstructed altimetry use gridded sea surface temperature satellite observations as an auxiliary ~~datasets~~ dataset for multivariate reconstruction. We use the AVHRR_OI-NCEI-L4-GLOB-v2.0 datasets (Reynolds et al., 2007; National Centers for Environmental Information, 2016) because it is a single consistent dataset covering the full time period of the altimetry data and because it matches approximately the ~~alimetry~~ altimetry dataset in terms of resolved spatial scales.

4 Implementation

Python code was first ported from TensorFlow 1.12 to 1.15 reducing the training time from 4.5 to 3.5 hours using a GeForce GTX 1080 GPU and Intel Core i7-7700 CPU. We also considered porting DINCAE to TensorFlow 2. TensorFlow 2 programming interface is however quite different from previous versions. As our group gained familiarity with the Julia programming

language (Bezanson et al., 2017), we decided to rewrite DINCAE in Julia. Porting DINCAE to Julia with the package Knet (Yuret, 2016) cut down the runtime from 3.5 to 1.9 hours (thanks to more efficient data transformation) using the AVHRR dataset described in Barth et al. (2020), and using a concatenation skip connection in both cases.

5   For the Adriatic test case, the input is a 3D array with the dimension corresponding to the longitude, latitude and ~~various parameters(an array of the size~~ the different parameters. The input parameters for an univariate reconstruction are three time instances of temperature scaled by the inverse of the error variance (previous, current and next day), the corresponding inverse of the error variance, longitude and latitude of every grid cell, sine and cosine of the day of the year multiplied by $2\pi/365.25$ as in Barth et al. (2020). The assembled array has the size of 168 x 144 x 10 elements (for a single training sample). The input

10 is processed by the encoder which is composed by 5 convolutional layers (with 16, 30, 58, 110 and 209 output filters) with a kernel size of 3 by 3 and a rectified linear unit (RELU) activation function followed by a max-pooling layer with a size of 2 by 2. The RELU activation function is commonly used in neural networks and is defined by $f(x) = \max(x, 0)$.

    The output of the encoder is transformed back to a full image by the decoder which mirrors the structure of the encoder.

15 The decoder is composed by 5 upsampling layers (nearest-neighborhood interpolation or bilinear interpolation) followed by a convolutional layer with the equivalent number output filters from the encoder (except for the final layer, which has only two outputs related to the reconstruction and its ~~standard deviation~~error variance). The final layer produces a 3D-array ~~T^{out}~~ T_{ijk}^{out} (size 168 x 144 x 2) from which the reconstruction $\hat{y}_{ij}$ and its ~~standard deviation~~ error variance $\hat{\sigma}_{ij}^2$ are computed by:

$$\hat{\sigma}_{ij}^2 = \frac{1}{\max(\exp(\min(T_{ij1}^{(\mathrm{out})}, \gamma)), \mu)} \tag{6}$$

$$\hat{y}_{ij} = T_{ij2}^{(\mathrm{out})} \hat{\sigma}_{ij}^2 \tag{7}$$

    where ~~$\gamma = 10$ and $\mu = 10^{-3}$. The~~ the parameters γ and μ are set to 10 and 10^{-3} respectively. The minimum and maximum functions ($\min$ and $\max$ ~~functions~~) are introduced here to prevent a division by a value close to zero or the exponentiation of a too large value. This stabilizes the neural network during the initial phase of the training as the weights and biases are randomly initialized.

    In Barth et al. (2020), after the convolutional layers the model included two fully-connected layers (with drop-out). This is no longer used as such layers require that the input matrix for training has exactly the same size as the input matrix of the reconstruction (inference) which makes this architecture difficult for large input arrays (which would arise for global or basin-wide sea surface temperature fields for example). The benefit of replacing dense layers by convolutional layers is further

30 discussed in Long et al. (2015).

number	type	output size	parameters of the layer
1	input	168 x 144 x 10	
2	conv. 2d	168 x 114 x 16	n. filters = 16, kernel size = (3,3)
3	pooling 2d	84 x 72 x 16	pool size = (2,2), strides = (2,2)
4	conv. 2d	84 x 72 x 30	n. filters = 30, kernel size = (3,3)
5	pooling 2d	42 x 36 x 30	pool size = (2,2), strides = (2,2)
6	conv. 2d	42 x 36 x 58	n. filters = 58, kernel size = (3,3)
7	pooling 2d	21 x 18 x 58	pool size = (2,2), strides = (2,2)
8	conv. 2d	21 x 18 x 110	n. filters = 110, kernel size = (3,3)
9	pooling 2d	11 x 9 x 110	pool size = (2,2), strides = (2,2)
10	conv. 2d	11 x 9 x 209	n. filters = 209, kernel size = (3,3)
11	pooling 2d	6 x 5 x 209	pool size = (2,2), strides = (2,2)
12	interpolation	11 x 9 x 209	
13	conv. 2d	11 x 9 x 110	n. filters = 110, kernel size = (3,3)
14	sum output of 13 and 9	11 x 9 x 110	
15	interpolation	21 x 18 x 110	
16	conv. 2d	21 x 18 x 58	n. filters = 58, kernel size = (3,3)
17	sum output of 16 and 9	21 x 18 x 58	
18	interpolation	42 x 36 x 58	
19	conv. 2d	42 x 36 x 30	n. filters = 30, kernel size = (3,3)
20	sum output of 19 and 5	42 x 36 x 30	
21	interpolation	84 x 72 x 30	
22	conv. 2d	84 x 72 x 16	n. filters = 16, kernel size = (3,3)
23	sum output of 22 and 3	84 x 72 x 16	
24	interpolation	168 x 144 x 16	
25	conv. 2d	168 x 144 x 2	n. filters = 2, kernel size = (3,3)

Table 1. Layers of the neural network for the gridded datasets. Note that every convolution is followed by a RELU activation function

.

skip connections
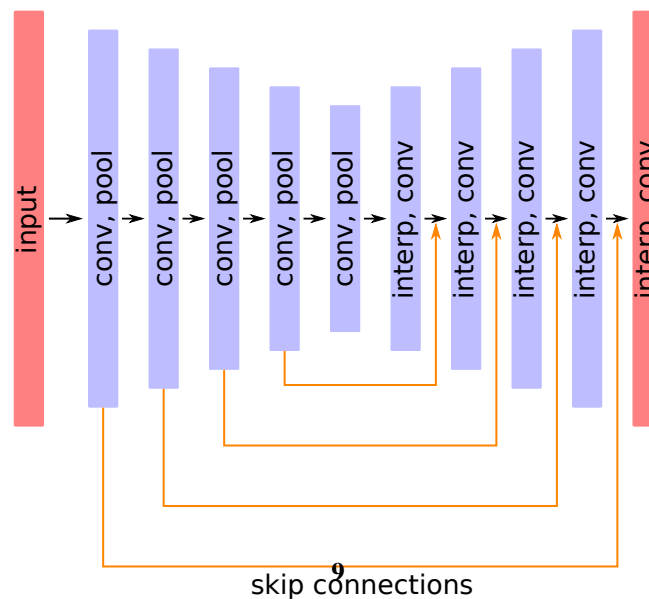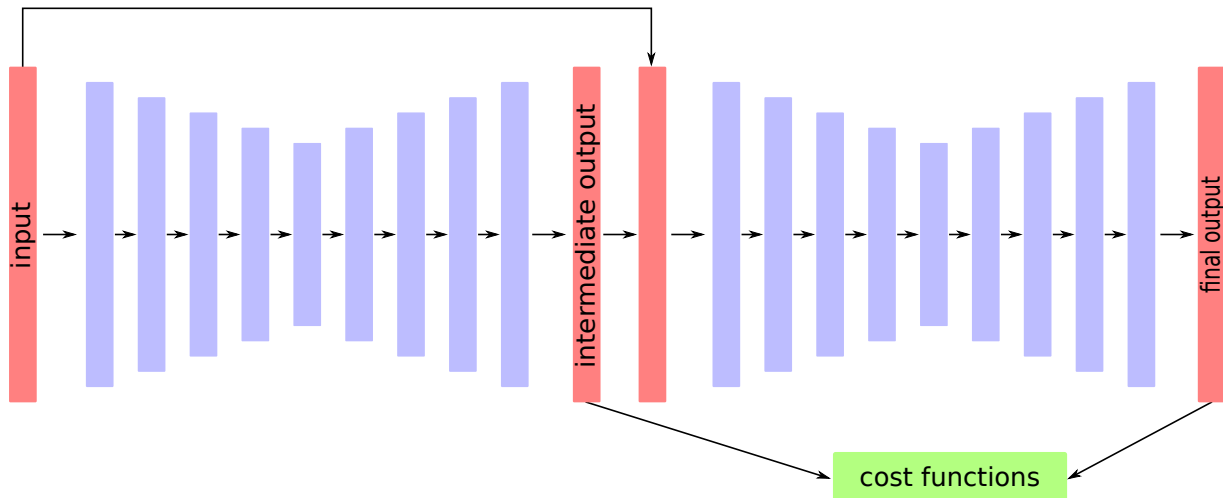
Figure 2. DINCAE with a refinement step composed essentially by two sequential autoencoders coupled such that the second autoencoder uses the output of the first and the input data.

The altimetry data ~~was analysed~~ were analyzed on a 0.25 degree resolution grid covering the area from 7^oW and 37^oE and 30^oN to 46^oN. For this neural network implementation, the input size is 177×69. The resolution is progressively reduced to 89×35, 45×18 and 23×9 by convolutional layers followed by max pooling layers with 32, 64 and 96 convolutional filters respectively. Skip connections are implemented after the 2nd convolutional layer onwards.

During training, Gaussian noise with a standard deviation σ_{pos} is added to the position of the measurements and every track of the current date has a certain probability p_{drop} to be withheld from the input of the neural networtk. The loss function is computed only on tracks from the current date. This helps the neural network to learn how to spread the information spatially. The neural network is optimized during 1000 epochs and the intermediate results are saved every 25 epochs.

The altimetry test case illustrates the results for a non-gridded dataset. Sea surface altimetry is usually gridded with a method like optimal interpolation or variational analysis. The latter can also be seen as a special case of optimal interpolation. For the autoencoder, the following fields are used as inputs:

– longitude, latitude of the measurement

15   – day of the year (~~sinus and cosinus~~sine and cosine) of the measurement multiplied by $2\pi/T_y$ where T_y is 365.25 (the length of a year in days).

– all data within a given centered time window of length Δt_{win}. For instance if the time window of length Δt_{win} is 9, the data from the current day are used as well as the tracks from the 4 previous days and the 4 following.

As in Barth et al. (2020), instead of using the observations directly, the observations are divided by their respective error variance and the inverse of the error variance is used as input. ~~After~~ Due to this scaling, it ~~is thus natural to consider missing~~

5 ~~data as zero values as a zero~~ follows that missing data results in a zero input value as it corresponds to a data point with an "infinitely" large error.

The training is done using mini-batches of size n_{batch}. The weights and biases in the neural network are updated using the gradient of the loss function evaluated at a batch of n_{batch} time instances (chosen at random). Evaluating the gradient using a subset of the training data chosen at random introduces some stochastic fluctuation allowing the optimization procedure to

10 escape a local minima.

All numerical experiments used the Adam optimizer (Kingma and Ba, 2014) with the standard parameter for the exponential decay rate for the first moment $\beta_1 = 0.9$, and for the second moment $\beta_1 = 0.999$, and a regularization parameter $\epsilon = 10^{-8}$. The learning rate α_n is computed for every n-th epoch as follows:

$$\alpha_n = \alpha_0 \, 2^{-\gamma_{\text{decay}} n}$$

where α_0 is the initial learning rate and γ_{decay} controls the exponential decay of the learning rate: every $1/\gamma_{\text{decay}}$ epochs the

15 learning rates is halved. If γ_{decay} is zero, then the learning rate is kept constant.

The batch size includes 32 time instances (all hyper-parameters are determined via ~~Baysian optimization as explained in the following~~ Bayesian optimization as described further on). The learning rate for the Adam optimizer is ~~0.000579728~~ 0.00058. The L2 regularization on the weights has been set to a β value of 10^{-4}. The upsampling method can either be by nearest neighbor or bilinear interpolation. In our tests, the nearest neighbor provided the lowest RMS error relative to the development

20 dataset. The absolute value of the gradients is clipped to 5 in order to stabilize the training. Satellite tracks from Δt_{win} = 27 time days (centred around the target time) are used to derive gridded altimetry data.

The hyper-parameters mentioned previously of the neural network have been determined by Bayesian optimization (Mockus et al., 1978; Jones et al., 1998; Snoek et al., 2012) by minimizing the RMS error relative to the development dataset. The so-called expected improvement acquisition function (Mockus et al., 1978) is used to determine which sequence of parameter

25 values is to be evaluated. Bayesian optimization as implemented by the Python package scikit-optimize (Head et al., 2020) was used in these tests.

5 Results and discussion

5.1 Gridded data

The new type of skip connection was first tested with the AVHRR Test case from the Ligurian Sea (Barth et al., 2020). The

30 previous best result (in terms of RMS) was 0.3835^oC using the cat-skip connection. With the new approach this RMS error is

reduced to 0.3604°C. The new type of skip connection makes the neural network more similar to residual networks which have been shown to be highly successful for image recognition tasks and allow to train deep networks more easily (He et al., 2015).

In table 2 we present the test case for the Adriatic Sea with and without the skip connections and using the multivariate reconstruction. Beside the RMS error, this table also includes the 10% and 90% percentiles of the absolute value of the difference between the reconstructed data and the cross-validation data to provide a typical range of the error. In general, using chlorophyll-a (or the wind fields) together with SST improved the results only marginally. The improvements were more consistent by using again the sum-skip connection instead of the cat-skip connection (in particular in conjunction with the additional refinement step). The analysis in the following uses the result of the lowest RMS, namely the reconstruction with sum skip-connections and refinement and with the considered variables (Chlorophyll a, the wind speed, zonal wind component, meridional wind component) as auxiliary parameters.

Table 2. RMS errors (in ~~degree Celsius~~°C) relative to the test dataset for different configurations (chlor_a: Chlorophyll a, wind_speed: the wind speed, uwnd: zonal wind component, vwnd: meridional wind component). The two numbers in gray correspond to the 10% and 90% percentiles of the absolute value of the error.

Auxilary parameters	cat skip-connections	sum skip-connections	sum skip-connections and refinement
none	0.66 (0.06 - 1.02)	0.60 (0.05 - 0.93)	0.55 (0.04 - 0.84)
chlor_a	0.64 (0.06 - 1.00)	0.59 (0.05 - 0.92)	0.54 (0.04 - 0.82)
chlor_a, wind_speed	0.65 (0.06 - 1.00)	0.58 (0.05 - 0.90)	0.54 (0.04 - 0.82)
chlor_a, wind_speed, uwnd, vwnd	0.66 (0.06 - 1.03)	0.57 (0.05 - 0.88)	0.54 (0.05 - 0.82)

When reconstructing sea surface temperature time series, it is often the case that for some days only very few data points are available. Figure 3 illustrates such a case where a quite clear image was almost entirely masked as missing. DINCAE essentially produces an average SST (still using previous and next time frames) for the time of the year and a realistic spatial distribution of the expected error. The few pixels that are available have a relatively low error as expected, but the overall error structure looks quite realistic as the expected error increases significantly in the coastal areas (where the variability is higher and where the original satellite data expected to be noisier). The reconstructed image matches the original image large-scale patterns relatively well, but as expected some small-scale structures are not reconstructed by the neural network. For this particular image, the RMS error (between the reconstructed data and the withheld data for cross-validation) is 0.43 °C and the 10% and 90 % percentiles of the absolute value of the error are 0.05 °C and 0.63 °C respectively.

Date: 2003-03-21
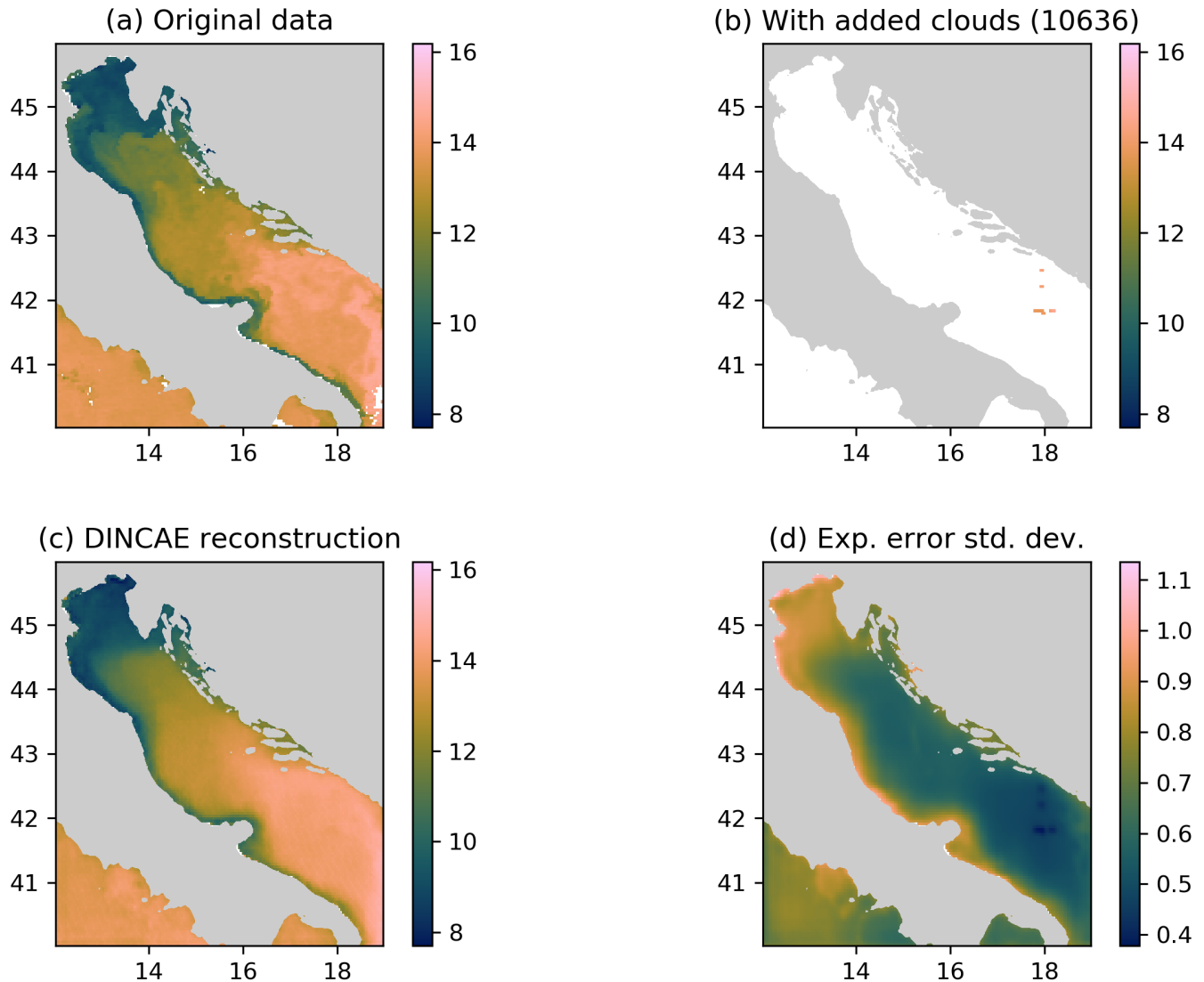
Figure 3. Panel (a) the original MODIS SST, (b) MODIS SST with additional clouds for cross-validation, (c) the DINCAE reconstruction using the data from panel (b), (d) the expected error standard deviation of the DINCAE reconstruction. All panels are in degrees Celsius.

The detection of cloud pixels in the MODIS dataset is generally good, but Figure 4 ~~)~~ for 2003-09-17 shows an example where some pixels were ~~characterised~~ characterized as valid sea points while probably they are (at least partially) obscured by clouds, resulting in an unrealistic low sea surface temperature. For most analysis techniques derived from optimal interpolation, outliers like undetected clouds typically degrade the ~~analysed~~ analyzed field in the vicinity of the spurious observations. The outlier also produced an ~~artefact~~ artifact in the output of DINCAE, but it is interesting to note that in this case, the ~~artefact~~

artifact did not spread spatially and the associated expected error has some elevated values indicating a potential issue at the location. The RMS error for this time instance is with 0.45 °Csimilar to the previously shown image. The typical range of the absolute value of the error is 0.04 - 0.72 °C(10% and 90 % percentiles).
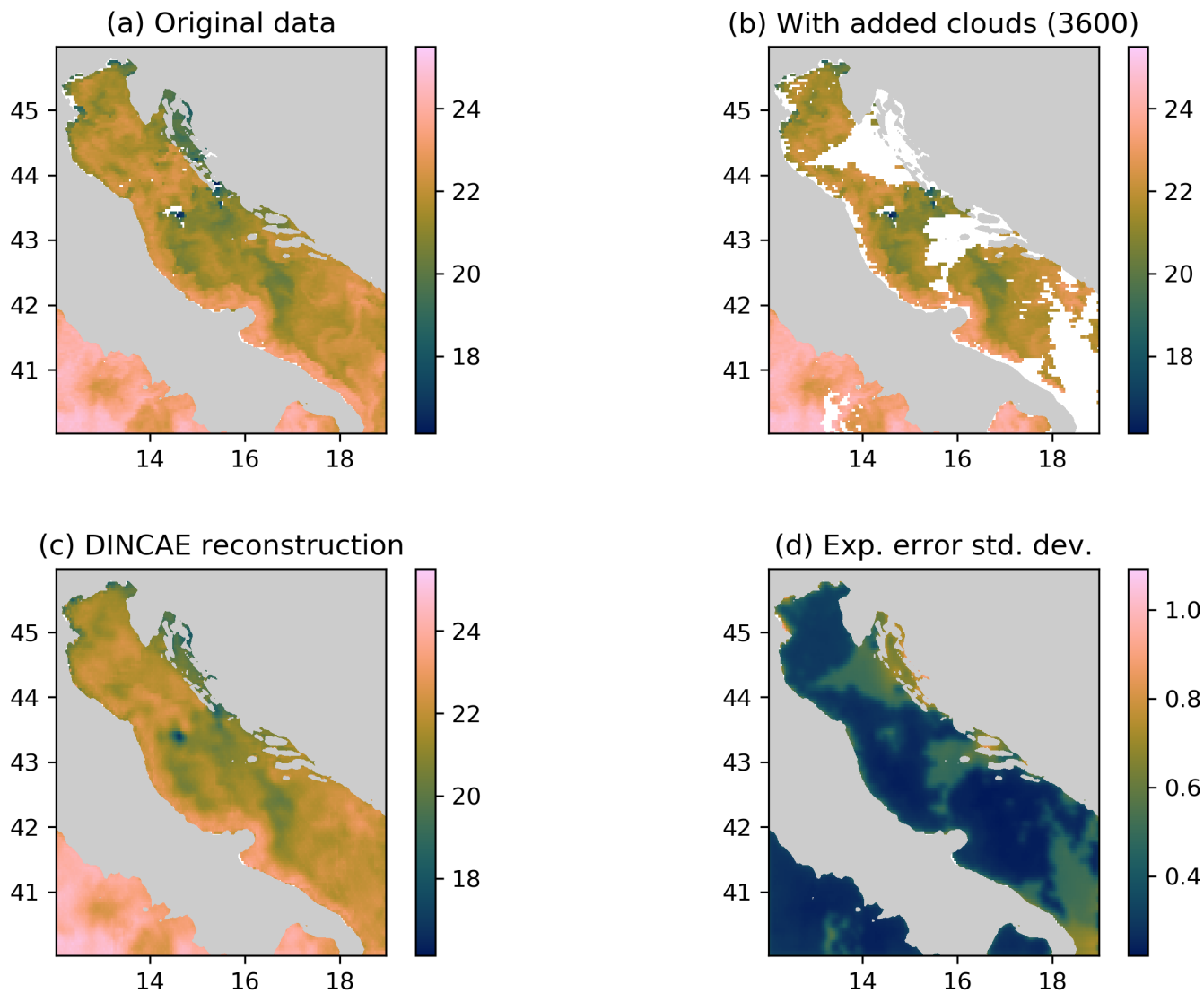
Figure 4. Panel (a) the original MODIS SST, (b) MODIS SST with additional clouds for cross-validation, (c) the DINCAE reconstruction using the data from panel (b), (d) the expected error standard deviation of the DINCAE reconstruction. All panels are in degrees Celsius.

A problem with techniques like optimal interpolation, variational analysis and to some degree also DINEOF, is that the reconstruction smoothes-out some small-scale features present in the initial data. For optimal interpolation and variational

analysis, this smoothing is explicitly induced by using a specific correlation length. In EOF-based methods, this is related to the truncation of the EOFs series. In DINCAE, the input data ~~is~~ are also compressed by a series of convolution and max-pooling layers, and some smoothing is also expected, as in figure 4. Figure 5 shows an example where the initial data ~~has~~ have almost no clouds and only few clouds are added for validation. The reconstructed field retains the filament and other small-scale structures. For this image, the RMS error (between the reconstructed data and the withheld data for cross-validation) is 0.55 °C and its typical range (as defined earlier) is 0.05 to 0.77 degrees Celsius. The degree of smoothing can be quantified by the RMS difference of the reconstructed data and the input data (which is not an independent validation metric). This RMS difference is 0.15 °C which is relatively low given the typical variability in sea-surface temperature in this region.

Date: 2004-08-29
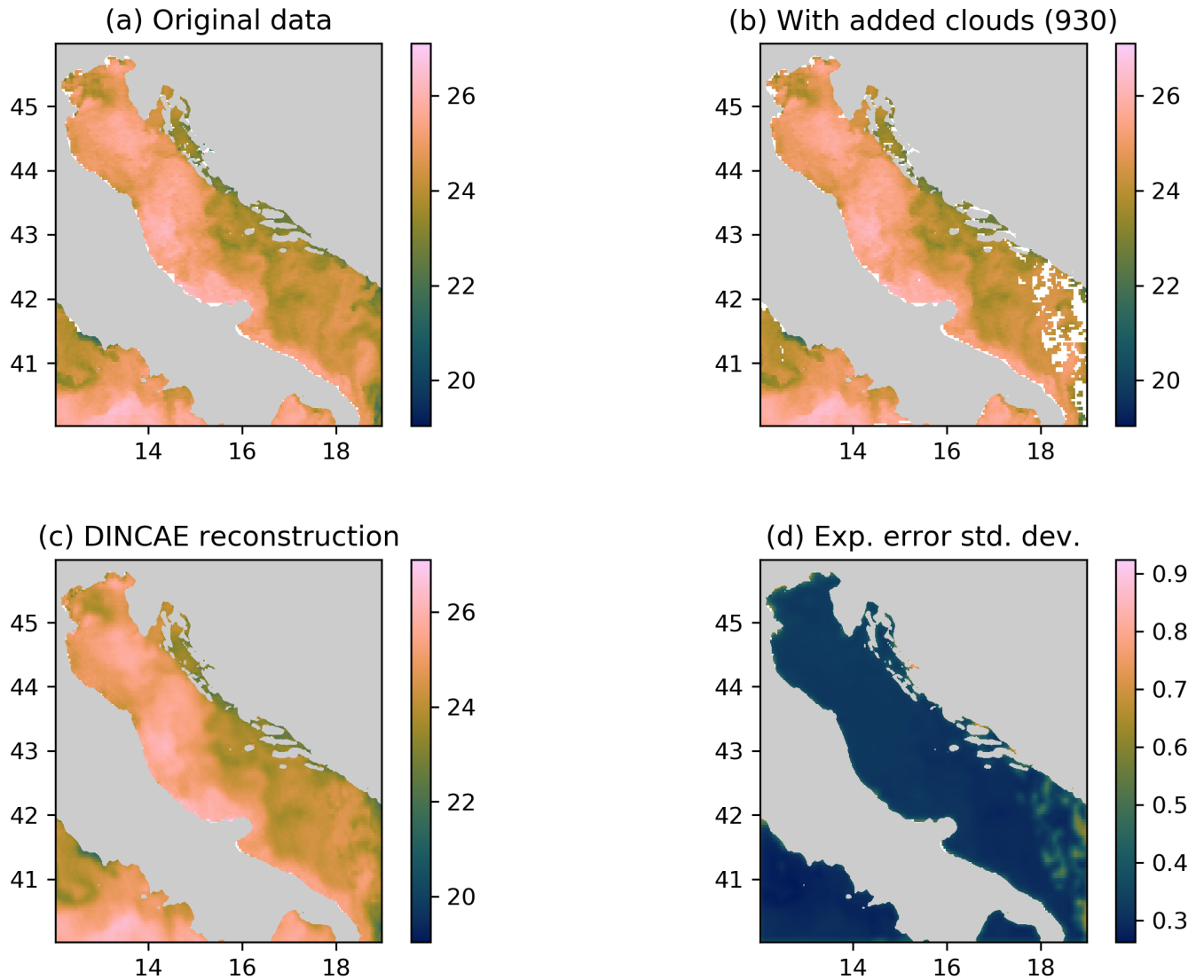
Figure 5. Panel (a) the original MODIS SST, (b) MODIS SST with additional clouds for cross-validation, (c) the DINCAE reconstruction using the data from panel (b), (d) the expected error standard deviation of the DINCAE reconstruction. All panels are in degrees Celsius.
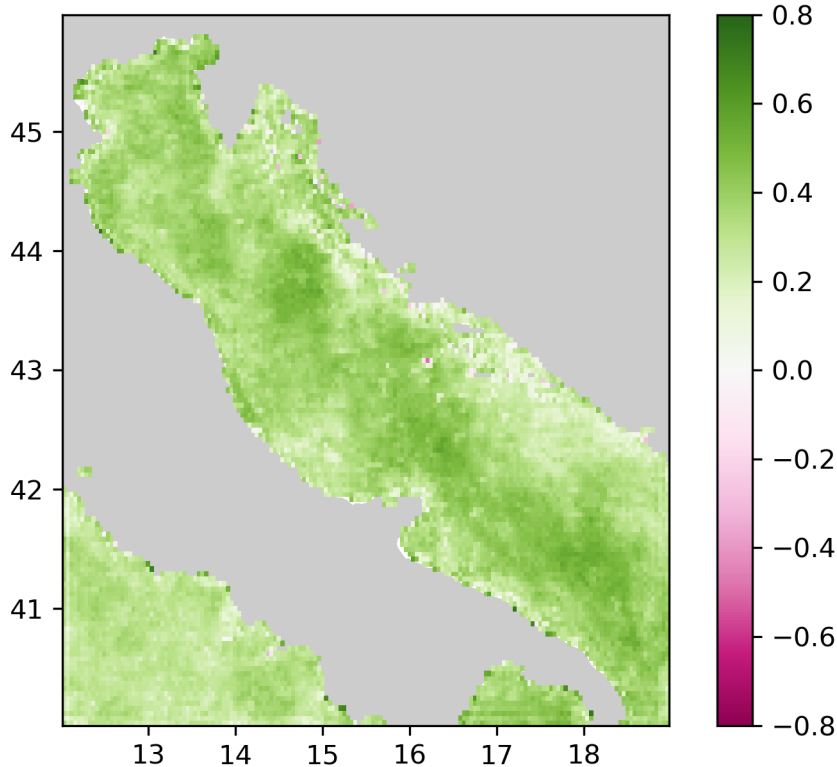
Figure 6. Mean square error skill score of the monovariate reconstruction corresponding to DINCAE 1.0 and the multivariate case (considering all variables) and with an additional refinement step.

To assess the improvement spatially, the mean square skill score S is computed (Figure 6) for every grid cell.

$$S = 1 - \frac{\text{MSE}}{\text{MSE}_{\text{ref}}} \tag{8}$$

where MSE_{ref} is the mean square error of the monovariate reconstruction corresponding to DINCAE 1.0 relative to the validation dataset (per grid cell and averaging over time) and MSE is the mean square error of the multivariate case (considering all variables) and with an additional refinement step. The improvement is spatially quite consistent. The mean square error is mostly reduced in the northern and central parts of the Adriatic. Only on some isolated grid cells a degradation is observed. The skill score reflects the combined improvement due to the 3 changes implemented in this version: updated skip-connections, refined step and multivariate reconstruction.
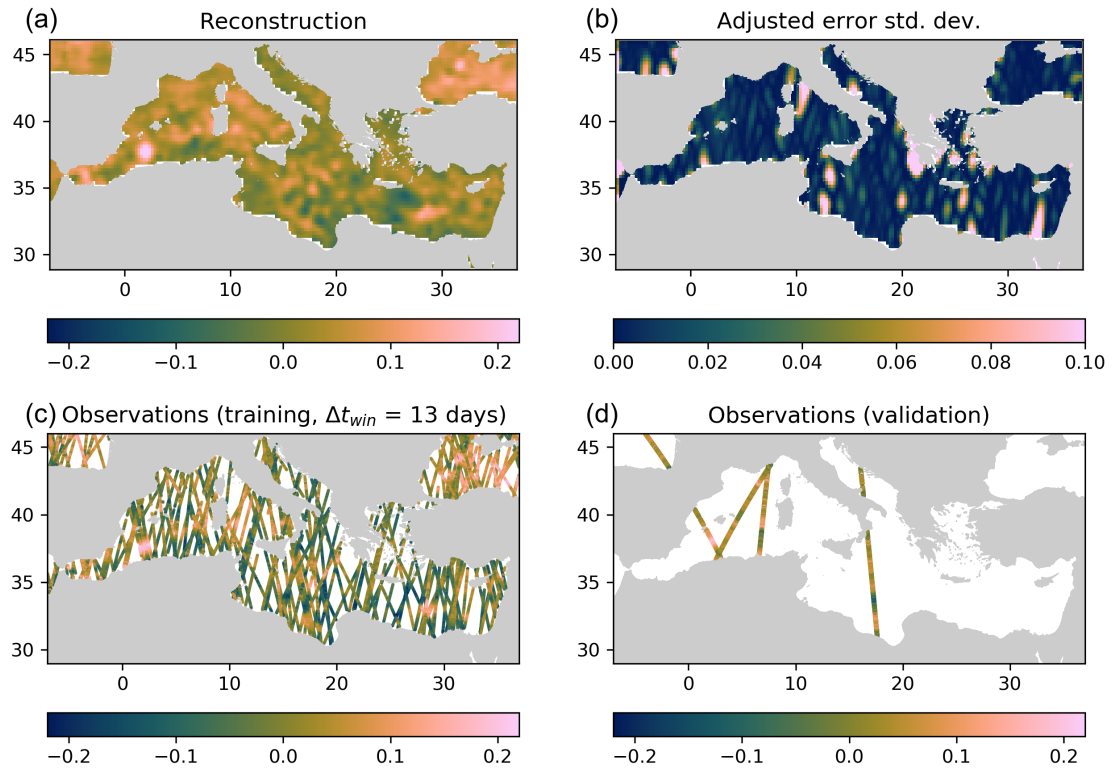
Figure 7. Panel (a) Reconstructed SLA by DIVAnd, (b) Expected error standard deviation by DIVAnd (~~without~~ with adjustment), (c) Data used during training (partial), (d) Independent data for validation withheld during analysis. All panels are in meters.
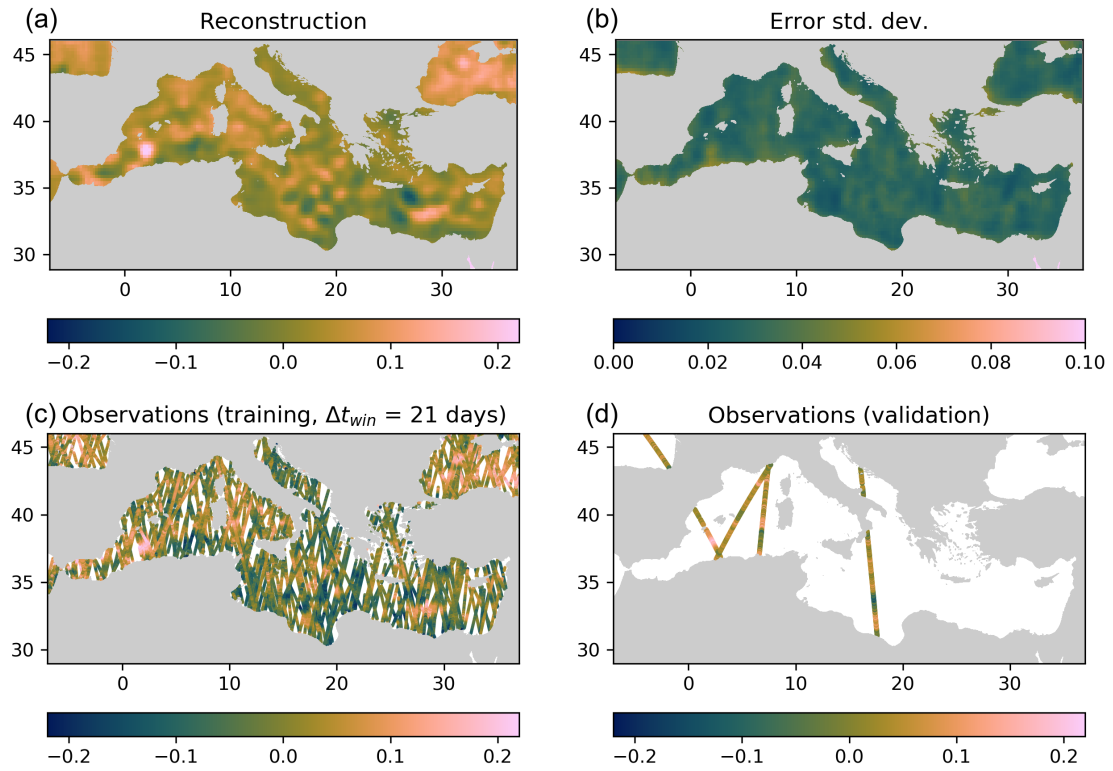
Figure 8. Panel (a) Reconstructed SLA by DINCAE, (b) Expected error standard deviation by DINCAE, (c) Data used during training (partial), (d) Independent data for validation withheld during analysis. All panels are in meters.
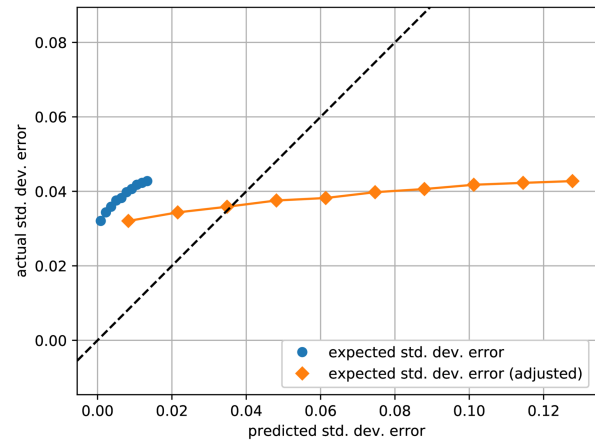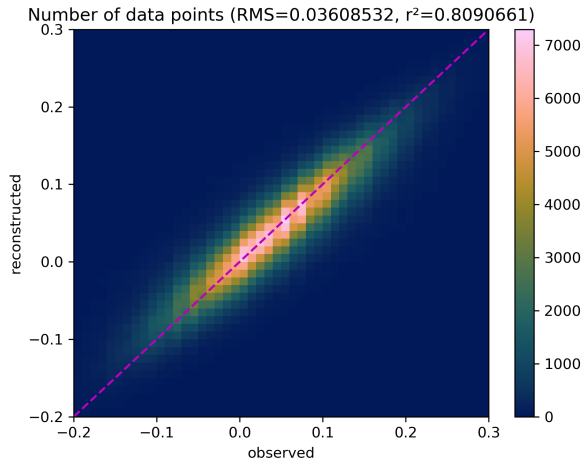
Figure 9. Left panel: altimetry observation from the test data versus the reconstructed values from DIVAnd (using only the training data). Right panel: expected standard deviation of the reconstruction (before and after adjustment) ~~relave~~ relative to the actual standard deviation of the reconstructed missfit. The x and y axes of both plots are expressed in meters.
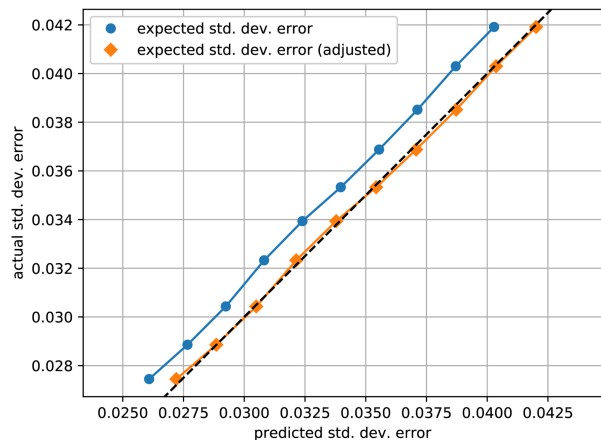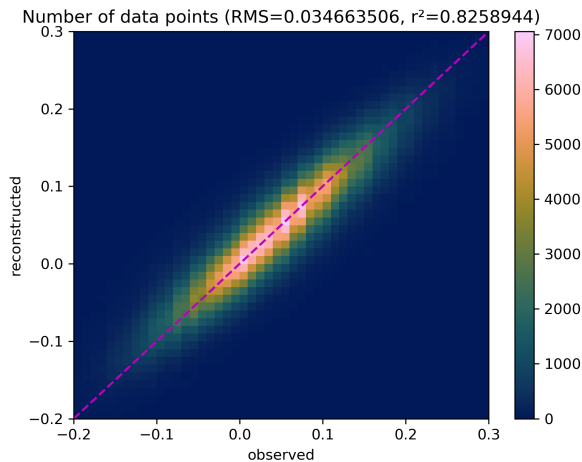
Figure 10. Left panel: altimetry observation from the test data versus the reconstructed values from DINCAE with SST as auxiliary parameter (using only the training data). Right panel: expected standard deviation of the reconstruction (before and after adjustment) ~~relave~~ relative to the actual standard deviation of the reconstructed missfit. The x and y axes of both plots are expressed in meters.

5.2 Non-gridded data

The altimetry data is first gridded by the tool DIVAnd (Barth et al., 2014). The main parameters here are the spatial correlation length (in km), the temporal correlation scale (days), the error variance of the observations (normalized by the background error variance) and the duration of the time window Δt_{win} determining which observations are used to compute the reconstruction

at the center of the time window.

All parameters of DIVAnd are also optimized using Bayesian minimization with expected improvement as acquisition function by minimizing the RMS error relative to the developpement datasets.

The best DIVAnd result is obtained with a horizontal correlation length of 74.8 km, a temporal correlation length of 5.5 days, a time window of 13 days and a normalized error variance of the observations of 20.5. An example reconstruction for the date 2017-06-07 is illustrated in Figure 7. The parameters are determined by Bayesian optimization minimizing the error relative to the developpement dataset. The RMS error of the analysis for these parameters is 3.61 cm relative to the independent test dataset (Figure 9).

The best performing neural network had a RMS error of 3.58 cm which is only slightly better than results of DIVAnd (3.60 cm). When using the Mediterranean sea-surface temperature as a co-variables we obtained a RMS error relative to the test dataset of 3.47 cm resulting in a clearer advantage of the neural network approach~~(Figure 10).~~. The left panel of Figures 9 and 10 shows on the x and y axis the observed altimetry (withheld during the analysis) and the corresponding reconstructed altimetry respectively. The range of altimetry values from -20 cm to 30 cm was divided in 51 bins of 1 cm. The colors indicate then the number of data points within each bin. For both reconstruction methods, the results scatter around the dashed line which corresponds to the case when reconstructed data correspond exactly to the observed altimetry. The eddy field of ~~this data is~~ the DINCAE dataset is also quite similar to the one obtained from DIVAnd, but the anomalies of the structure are more pronounced in the DINCAE reconstruction (Figure 8).

DINCAE and DIVAnd provide a field with the estimated expected error. For DIVAnd we used the so-called "clever poorman's method" as described in Beckers et al. (2014) and the background error variance is estimated by fitting an empirical covariance based on a random pair of points binned by their distance (Thiebaux, 1986; Troupin et al., 2012). The estimated error ~~standard deviation~~ variance is later adjusted by a factor to account for uncertainties in estimating the background error variance.

We made 10 categories of pixels based on the expected standard deviation error, evenly distributed between the ~~10th and 90th percentile~~ 10% and 90% percentiles of the expected standard deviation error. For every category we computed the actual RMS relative to the test dataset. Ideally this should correspond to the estimated expected error of the reconstruction (including the observational error). A global adjustment factor is also applied so that the average RMS error matches the mean expected error standard deviation which is represented in the left panels of Figures 9 and 10. This adjustment factor is also applied to panel (b) of Figure 7. The main advantage of DINCAE relative to DIVAnd is the improved estimate of the error variance of the results.

In summary, the accuracy of the DINCAE reconstruction is slightly better than the accuracy of the DIVAnd analysis. However, the main improvement of the DINCAE approach here is that the expected error variance of the analysis is much more reliable than the expected error variance of DIVAnd.
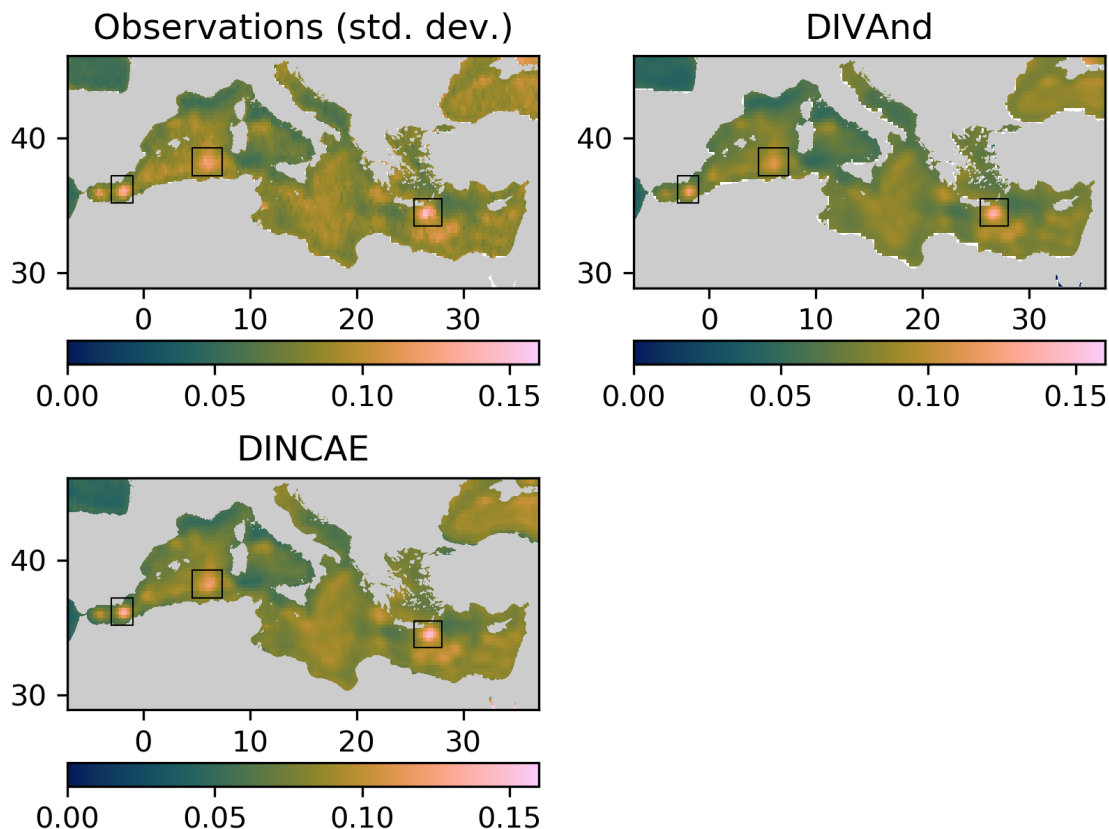
Figure 11. Standard deviation of the sea-level anomaly for the DIVAnd method and DINCAE (including SST as auxiliary parameter)

Figure 11 shows the standard deviation of the sea-level anomaly computed over the whole time period. From this figure, three areas in particular stand out corresponding (from East to West) to the East Alboran Gyre, regions of the Algerian current and the Ierapetra Anticyclone (annotated with the black rectangle in Figure 11). The maximum standard deviation (related to the surface transport variability) for these 3 areas is shown in Table 3. The standard deviation is also computed from the satellite altimetry data considering all satellite observations falling within a given grid cell (excluding coastal grid cells with less than 10 observations). The standard deviation of the DINCAE reconstruction is in all 3 regions higher than the standard deviation

for DIVAnd despite that the DINCAE reconstruction has a lower RMS error than DIVAnd. Also the standard deviation of DINCAE is in general closer to the observed standard deviation.

Table 3. Maximum standard deviation in three selected areas

	Obs. std. dev.	DIVAnd std. dev.	DINCAE std. dev.
East Alboran Gyre	0.136	0.123	0.141
regions of the Alboran current	0.125	0.112	0.121
Ierapetra Anticyclone	0.153	0.138	0.151

6 Conclusions

In this paper, we discussed improvements of the previous described DINCAE method. The code has been extended to handle multi-variate reconstructions which were also described in Han et al. (2020). We also found that multivariate reconstruction can improve the reconstruction, but the largest improvement was obtained by changing the structure of the neural network by using a newly implemented different type of skip connection and refinement pass. Interestingly, this type bears some similarities to the hierarchical multigrid method for solving partial differential equations. The handling of different types of satellite data was also improved. While most ocean satellite observations are gridded data (like sea surface temperature, ocean color, sea surface salinity, sea ice concentration), some parameters can only be inferred by nadir looking satellites along tracks. For such non-gridded datasets the first input layer is extended to handle such arbitrarily location input data. We were able to show that the DINCAE method applied with altimetry data produces better reconstructions, but the main advantages are the significantly improved estimates of the expected reconstruction error variance. In this case, the DINCAE method compares to the DIVAnd method (which is similar to optimal interpolation) favorably in terms of reliability of the expected error variance, accuracy of the reconstruction relative to the test dataset and realism of the temporal standard deviation of the reconstruction assessed from the standard deviation of the observations.

Code and data availability. The source code is released as open source under the terms of the GNU General Public Licence v3 (or, at your option, any later version) and available at the address https://github.com/gher-ulg/DINCAE.jl in the branch `dev` (doi: 10.5281/zenodo.5575067). The sea surface temperature (MODIS Terra Level 3 SST Thermal IR Daily 4km Nighttime v2014.0, doi:10.5067/MODST-1D4N4) is available by PO.DAAC (https://podaac.jpl.nasa.gov/, JPL, NASA, US), Wind speed (Cross-Calibrated Multi-Platform (CCMP) gridded surface vector winds) is available from Remote Sensing Systems (http://www.remss.com/measurements/ccmp/). Chlorophyll-a from Ocean Biology Processing Group, NASA (US) can be accessed at. https://oceancolor.gsfc.nasa.gov/data/overview/. Altimetry data are made available by the Copernicus Marine Environment Monitoring Service (CMEMS, https://marine.copernicus.eu/) (dataset `SEALEVEL_EUR_PHY_L3_MY_OBSERVATIONS_008_061`, accessed on 2020-10-13). The L4 gridded SST over the Mediterranean is the NOAA Optimum Interpolation 1/4 Degree Daily Sea Surface Temperature (OISST) Analysis, Version 2 available at https://www.ncei.noaa.gov/metadata/geoportal/rest/metadata/item/gov.noaa.ncdc:C00844/html (doi:10.7289/V5SQ8XB5).

Author contributions. A.B. designed and implemented the neural network. A.B., A.A.A., C.T. and J.M.B. contributed to the planning and discussions and to the writing of the manuscript.

Competing interests. The authors have no competing interests

Acknowledgements. The F.R.S.-FNRS (Fonds de la Recherche Scientifique de Belgique) is acknowledged for funding the position of Alexander Barth. This research was partly performed with funding from the Belgian Science Policy Office (BELSPO) STEREO III program in the framework of the MULTI-SYNC project (contract SR/00/359). Computational resources have been provided in part by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the F.R.S.-FNRS under Grant No. 2.5020.11 and by the Walloon Region. The authors wishes also to thanks to Julia community and in particular Deniz Yuret from the Koç University (Istanbul, Turkey) for the Knet.jl package and Tim Besard (Julia Computing, Massachusetts, United States) for the CUDA.jl package as well as the developers of the python library scikit-optimize. We thank the reviewers for their careful reading of the manuscript and their constructive and insightful comments.

References

Alvera-Azcárate, A., Barth, A., Beckers, J.-M., and Weisberg, R. H.: Multivariate reconstruction of missing data in sea surface temperature, chlorophyll and wind satellite field, Journal of Geophysical Research, 112, C03 008, https://doi.org/10.1029/2006JC003660, http://hdl.handle.net/2268/9485, 2007.

5   Atlas, R., Hoffman, R. N., Ardizzone, J., Leidner, S. M., Jusem, J. C., Smith, D. K., and Gombos, D.: A cross-calibrated, multiplatform ocean surface wind velocity product for meteorological and oceanographic applications, Bulletin of the American Meteorological Society, 92, 157–174, https://doi.org/10.1175/2010BAMS2946.1, 2011.

Barth, A., Beckers, J.-M., Troupin, C., Alvera-Azcárate, A., and Vandenbulcke, L.: divand-1.0: n-dimensional variational data analysis for ocean observations, Geoscientific Model Development, 7, 225–241, https://doi.org/10.5194/gmd-7-225-2014, http://www.geosci-model-dev.net/7/225/2014/, 2014.

Barth, A., Alvera-Azcárate, A., Licer, M., and Beckers, J.-M.: DINCAE 1.0: a convolutional neural network with error estimates to reconstruct sea surface temperature satellite observations, Geoscientific Model Development, 13, 1609–1622, https://doi.org/10.5194/gmd-13-1609-2020, 2020.

Beckers, J.-M., Barth, A., Troupin, C., and Alvera-Azcárate, A.: Approximate and efficient methods to assess error fields in spa-
15   tial gridding with data interpolating variational analysis (DIVA), Journal of Atmospheric and Oceanic Technology, 31, 515–530, https://doi.org/10.1175/JTECH-D-13-00130.1, http://hdl.handle.net/2268/161069, 2014.

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B.: Julia: A fresh approach to numerical computing, SIAM review, 59, 65–98, https://doi.org/10.1137/141000671, 2017.

Gong, Z., Zhong, P., and Hu, W.: Diversity in Machine Learning, IEEE Access, 7, 64 323–64 350,
20   https://doi.org/10.1109/ACCESS.2019.2917620, 2019.

Han, Z., He, Y., Liu, G., and Perrie, W.: Application of DINCAE to Reconstruct the Gaps in Chlorophyll-a Satellite Observations in the South China Sea and West Philippine Sea, Remote Sensing, 12, https://doi.org/10.3390/rs12030480, 2020.

He, K., Zhang, X., Ren, S., and Sun, J.: Deep Residual Learning for Image Recognition, 2015.

Head, T., Kumar, M., Nahrstaedt, H., Louppe, G., and Shcherbatyi, I.: scikit-optimize/scikit-optimize,
25   https://doi.org/10.5281/zenodo.4014775, v0.8.1, 2020.

Hochreiter, S.: The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 06, 107–116, https://doi.org/10.1142/S0218488598000094, 1998.

Jones, D. R., Schonlau, M., and Welch, W.: Efficient Global Optimization of Expensive Black-Box Functions, Journal of Global Optimization, 13, 455–492, https://doi.org/10.1023/A:1008306431147, 1998.

30   Kingma, D. P. and Ba, J.: Adam: A Method for Stochastic Optimization, CoRR, abs/1412.6980, http://arxiv.org/abs/1412.6980, 2014.

Liang, Y.-C., Mazloff, M. R., Rosso, I., Fang, S.-W., and Yu, J.-Y.: A Multivariate Empirical Orthogonal Function Method to Construct Nitrate Maps in the Southern Ocean, Journal of Atmospheric and Oceanic Technology, 35, 1505 – 1519, https://doi.org/10.1175/JTECH-D-18-0018.1, 2018.

Liu, H., Jiang, B., Xiao, Y., and Yang, C.: Coherent Semantic Attention for Image Inpainting, in: 2019 IEEE/CVF International Conference on
35   Computer Vision (ICCV), pp. 4169–4178, https://doi.org/10.1109/ICCV.2019.00427, http://openaccess.thecvf.com/content_ICCV_2019/html/Liu_Coherent_Semantic_Attention_for_Image_Inpainting_ICCV_2019_paper.html, 2019.

Long, J., Shelhamer, E., and Darrell, T.: Fully convolutional networks for semantic segmentation, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440, https://doi.org/10.1109/CVPR.2015.7298965, 2015.

Mears, C. A., Scott, J., Wentz, F. J., Ricciardulli, L., Leidner, S. M., Hoffman, R., and Atlas, R.: A Near Real Time Version of the Cross Calibrated Multiplatform (CCMP) Ocean Surface Wind Velocity Data Set, Journal of Geophysical Research: Oceans, https://doi.org/10.1029/2019JC015367, 2019.

Mockus, J., Tiesis, V., and Zilinskas, A.: The Application of Bayesian Methods for Seeking the Extremum, Towards Global Optimization, 2, 2, 1978.

Nardelli, B. B. and Santoleri, R.: Methods for the Reconstruction of Vertical Profiles from Surface Data: Multivariate Analyses, Residual GEM, and Variable Temporal Signals in the North Pacific Ocean, Journal of Atmospheric and Oceanic Technology, 22, 1762 – 1781, https://doi.org/10.1175/JTECH1792.1, 2005.

National Centers for Environmental Information: Daily L4 Optimally Interpolated SST (OISST) In situ and AVHRR Analysis. Ver. 2.0. PO.DAAC, CA, USA., https://doi.org/10.5067/GHAAO-4BC02, dataset accessed 2020-07-14, 2016.

Olmedo, E., Taupier-Letage, I., Turiel, A., and Alvera-Azcárate, A.: Improving SMOS Sea Surface Salinity in the Western Mediterranean Sea through multivariate and multifractal analysis, Remote Sensing, 10, 485, https://doi.org/10.3390/rs10030485, 2018.

Olmedo, E., González-Haro, C., Hoareau, N., Umbert, M., González-Gambau, V., Martínez, J., Gabarró, C., and Turiel, A.: Nine years of SMOS sea surface salinity global maps at the Barcelona Expert Center, Earth System Science Data, 13, 857–888, https://doi.org/10.5194/essd-13-857-2021, 2021.

Reynolds, R. W., Smith, T. M., Liu, C., Chelton, D. B., Casey, K. S., and Schlax, M. G.: Daily High-resolution Blended Analyses for sea surface temperature, Journal of Climate, 20, 5473–5496, https://doi.org/10.1175/2007JCLI1824.1, 2007.

Ronneberger, O., Fischer, P., and Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, edited by Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., pp. 234–241, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-319-24574-4_28, 2015.

Snoek, J., Larochelle, H., and Adams, R. P.: Practical Bayesian optimization of machine learning algorithms, in: Advances in Neural Information Processing Systems, pp. 2951–2959, 2012.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A.: Going Deeper with Convolutions, in: Computer Vision and Pattern Recognition (CVPR), http://arxiv.org/abs/1409.4842, 2015.

Thiebaux, J.: Anisotropic correlation functions for Objective Analysis., Monthly Weather Review, 104, 994–1002, https://doi.org/10.1175/1520-0493(1976)104<0994:ACFFOA>2.0.CO;2, 1986.

Troupin, C., Barth, A., Sirjacobs, D., Ouberdous, M., Brankart, J.-M., Brasseur, P., Rixen, M., Alvera-Azcárate, A., Belounis, M., Capet, A., Lenartz, F., Toussaint, M.-E., and Beckers, J.-M.: Generation of analysis and consistent error fields using the Data Interpolating Variational Analysis (DIVA), Ocean Modelling, 52–53, 90–101, https://doi.org/10.1016/j.ocemod.2012.05.002, http://hdl.handle.net/2268/125731, 2012.

Wentz, F., Scott, J., Hoffman, R., Leidner, M., Atlas, R., and Ardizzone, J.: Remote Sensing Systems Cross-Calibrated Multi-Platform (CCMP) 6-hourly ocean vector wind analysis product on 0.25 deg grid, Version 2.0, Remote Sensing Systems, Santa Rosa, CA. Available online at www.remss.com/measurements/ccmp. Accessed 19 July 2019, 2019.

Wylie, D., Jackson, D. L., Menzel, W. P., and Bates, J. J.: Trends in Global Cloud Cover in Two Decades of HIRS Observations, Journal of Climate, 18, 3021–3031, https://doi.org/10.1175/JCLI3461.1, 2005.

Yuret, D.: Knet: beginning deep learning with 100 lines of julia, in: Machine Learning Systems Workshop at NIPS, 2016.

Zhang, Z., Stanev, E. V., and Grayek, S.: Reconstruction of the Basin-Wide Sea-Level Variability in the North Sea Using Coastal Data and Generative Adversarial Networks, Journal of Geophysical Research: Oceans, 125, e2020JC016402, https://doi.org/10.1029/2020JC016402, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020JC016402, e2020JC016402 2020JC016402, 2020.