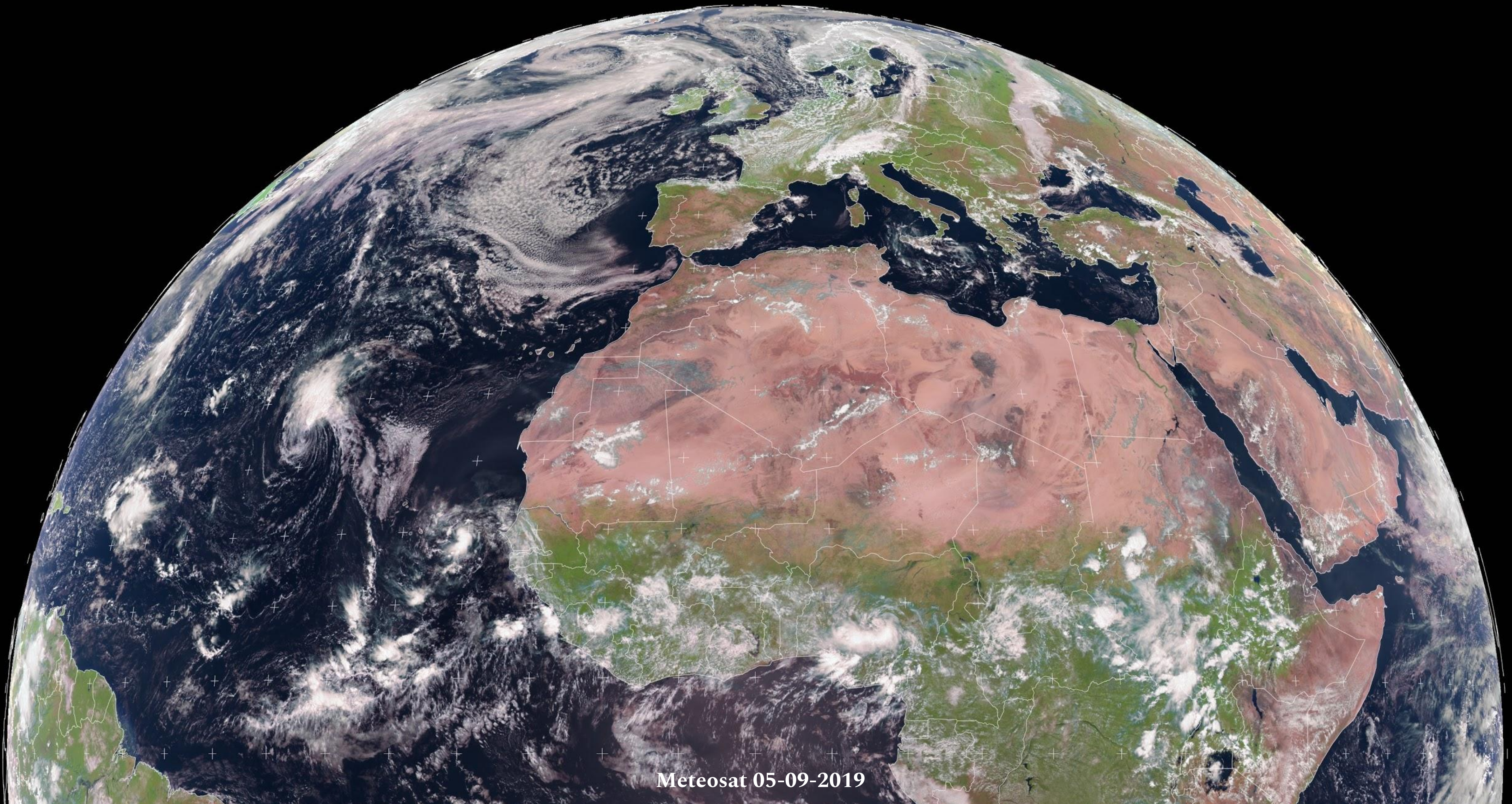
DINCAE: reconstruction of missing satellite data with a convolutional auto-encoder

Alexander Barth, Aida Alvera-Azcárate, Matjaz Licer, and Jean-Marie Beckers

Phi-Week
9-13 September 2019

LIÈGE université

NACIONALNI INŠTITUT ZA BIOLOGIJO
NATIONAL INSTITUTE OF BIOLOGY

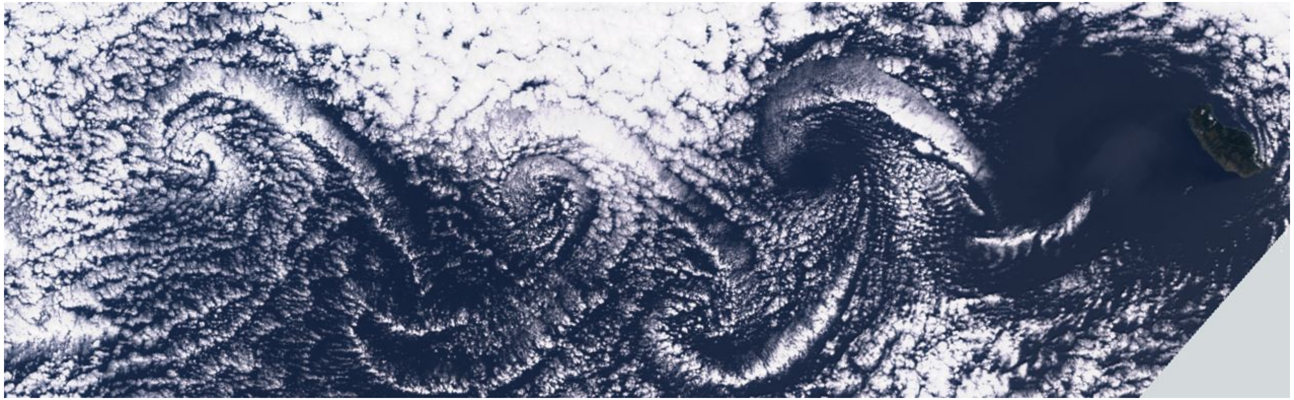


Meteosat 05-09-2019

Objectives

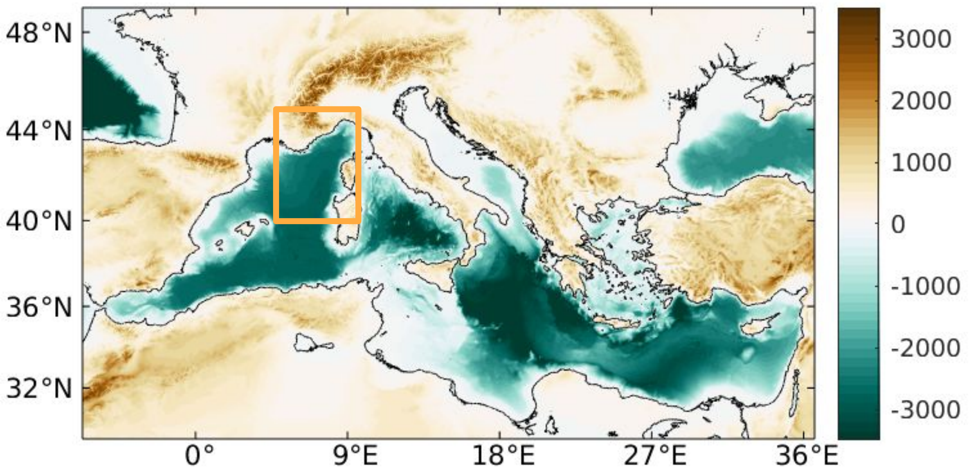
- To derive a methodology to reconstruct missing information in satellite data
    - Based on neural networks
    - Making use of ~four decades of sea surface temperature measurements
    - Able to retain small scale variability

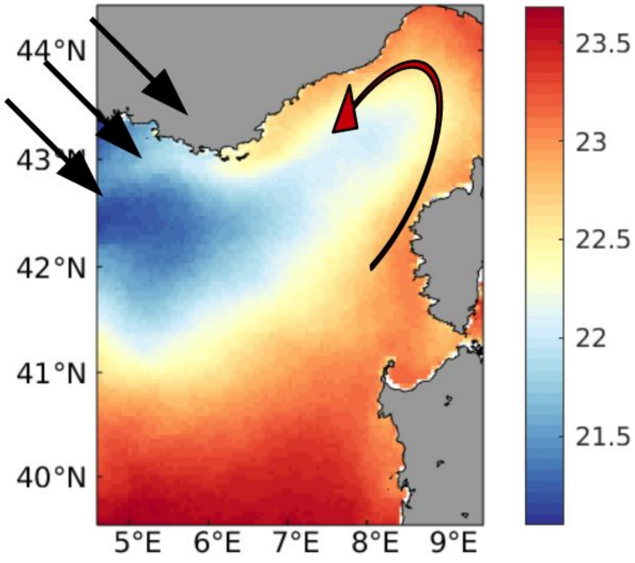
- To assess the benefit of using neural networks in comparison with other state-of-the-art methodologies
    - DINEOF (Data Interpolating Empirical Orthogonal Functions)



Data used

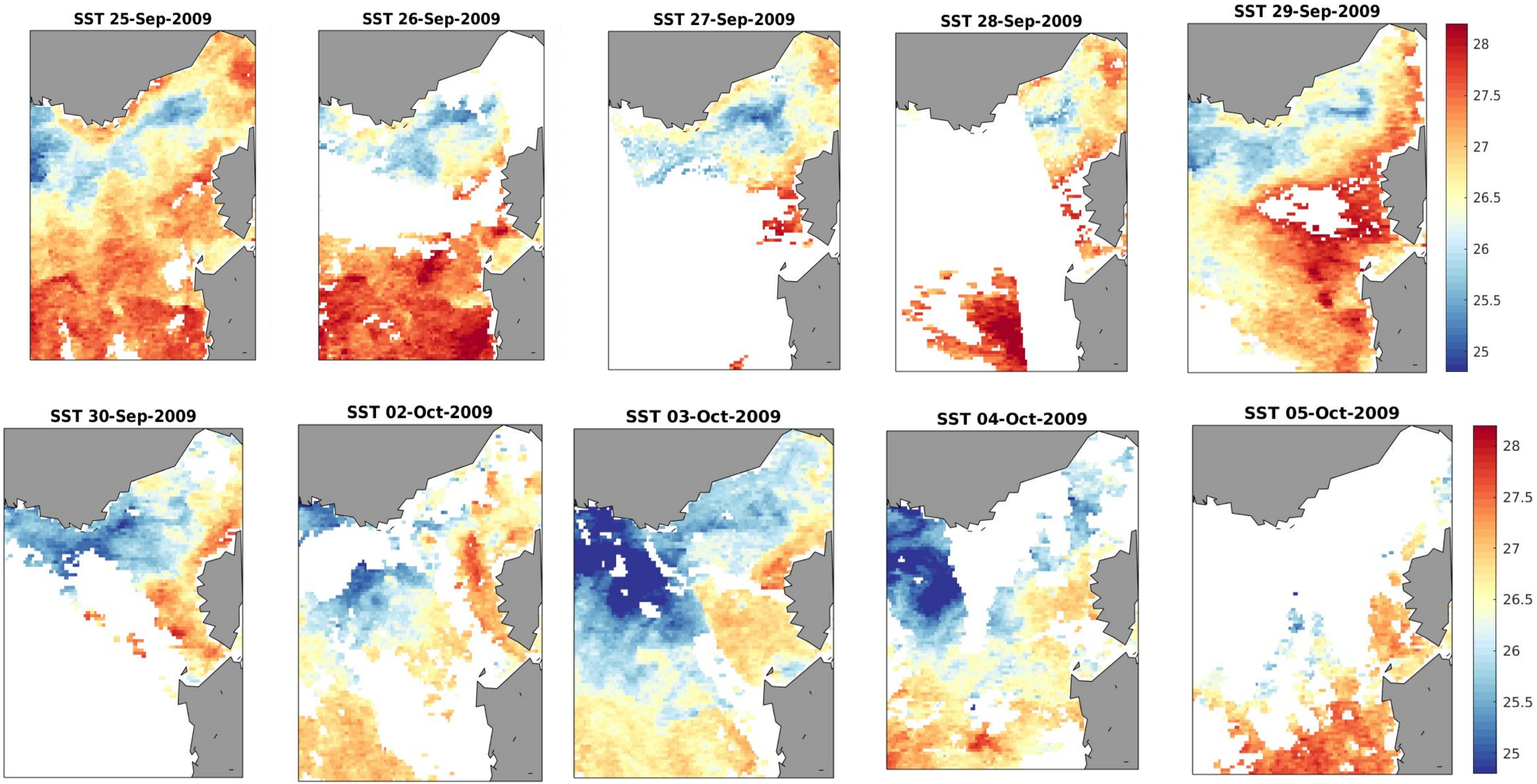
- Daily Advanced Very High Resolution Radiometer (AVHRR) Sea Surface Temperature (SST) data
- 4 km spatial resolution
- Liguro-Provençal basin (western Mediterranean Sea)
- 1 April 1985 to 31 December 2009 (25 years)
- 47 % of missing data





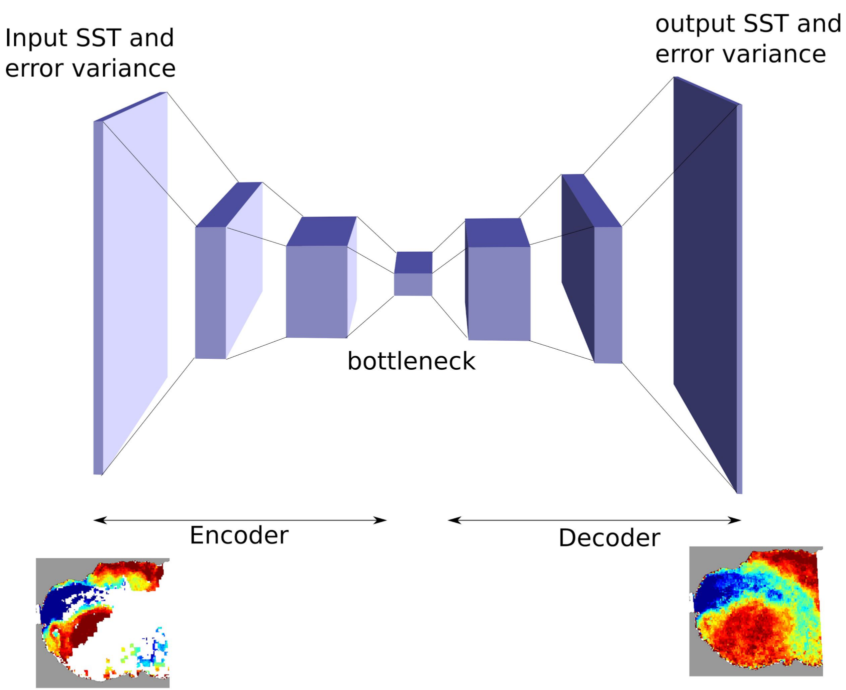
Average SST (°C)

Challenge: **training on gappy data (lots of gaps!)**



Methodology

DINCAE: Data-Interpolating Convolutional Auto-Encoder



Auto-Encoder: used to efficiently compress/decompress data, by extracting main patterns of variability

- Similarity to EOFs

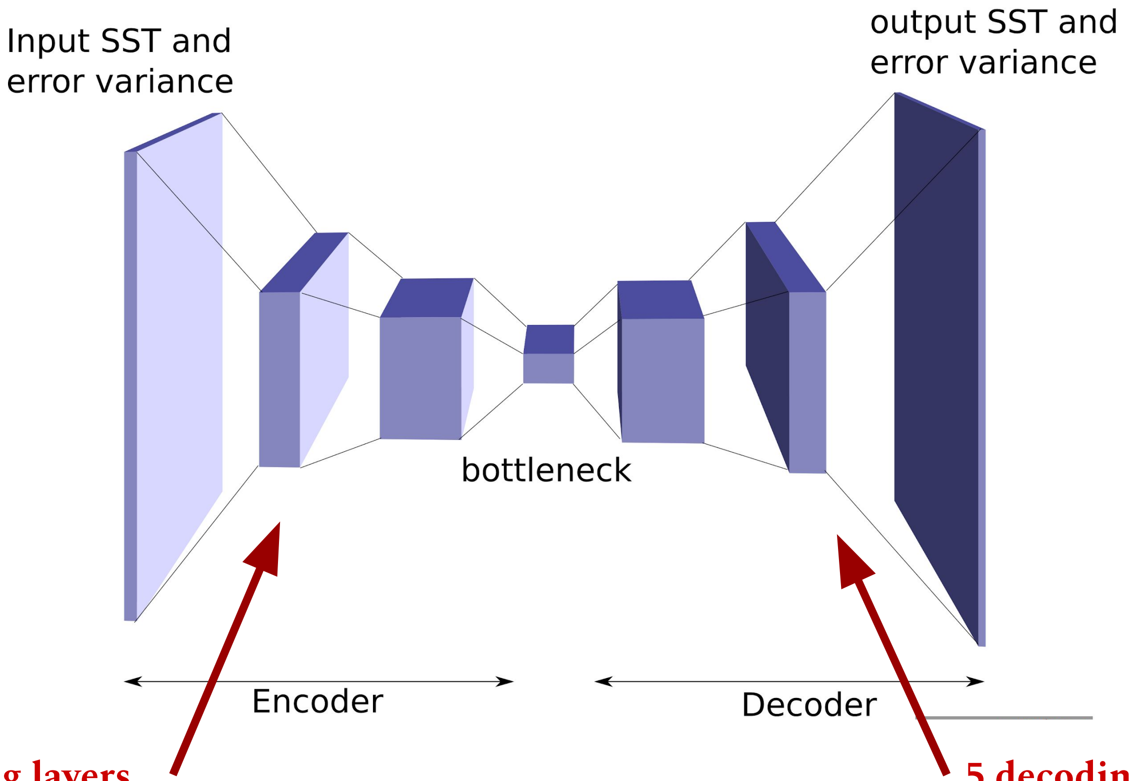
Convolutional: works on subsets of data, i.e. trains on local features

Missing data handled as data with different initial errors

- If missing, error variance (σ^2) tends to ∞

Input data:

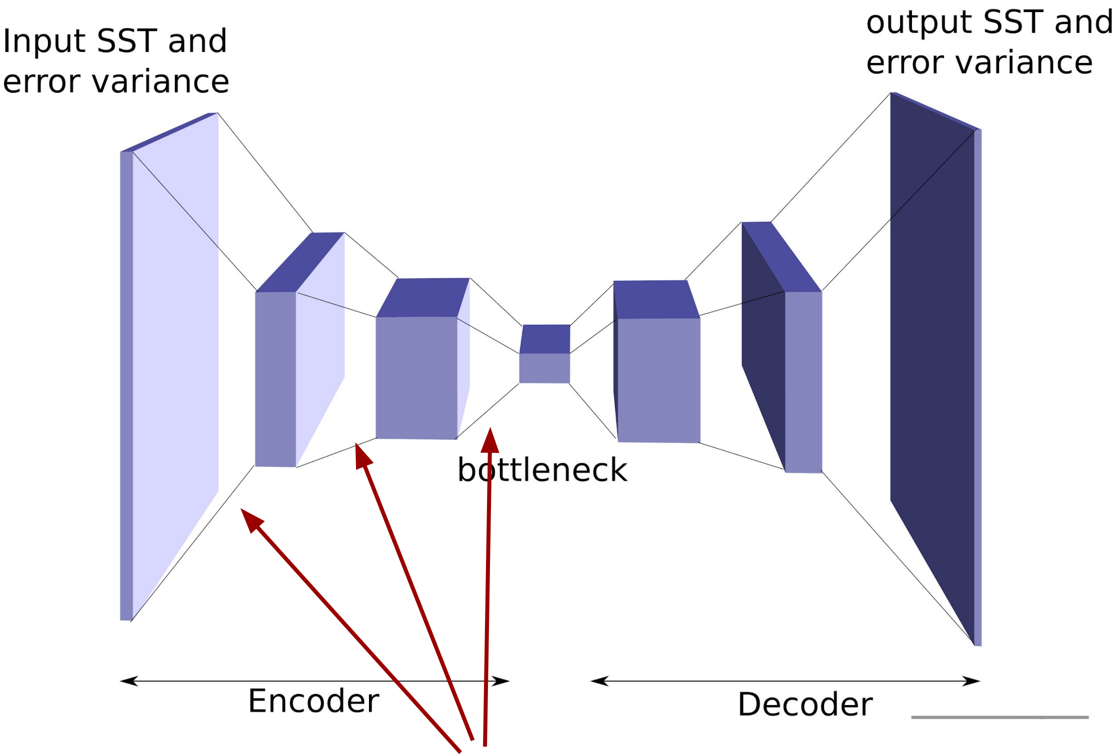
- SST/σ^2 (previous day, current day, following day)
- 1/σ^2 (previous day, current day, following day)
- Longitude
- Latitude
- Time (cosine and sine of the year-day/365.25)



5 encoding layers

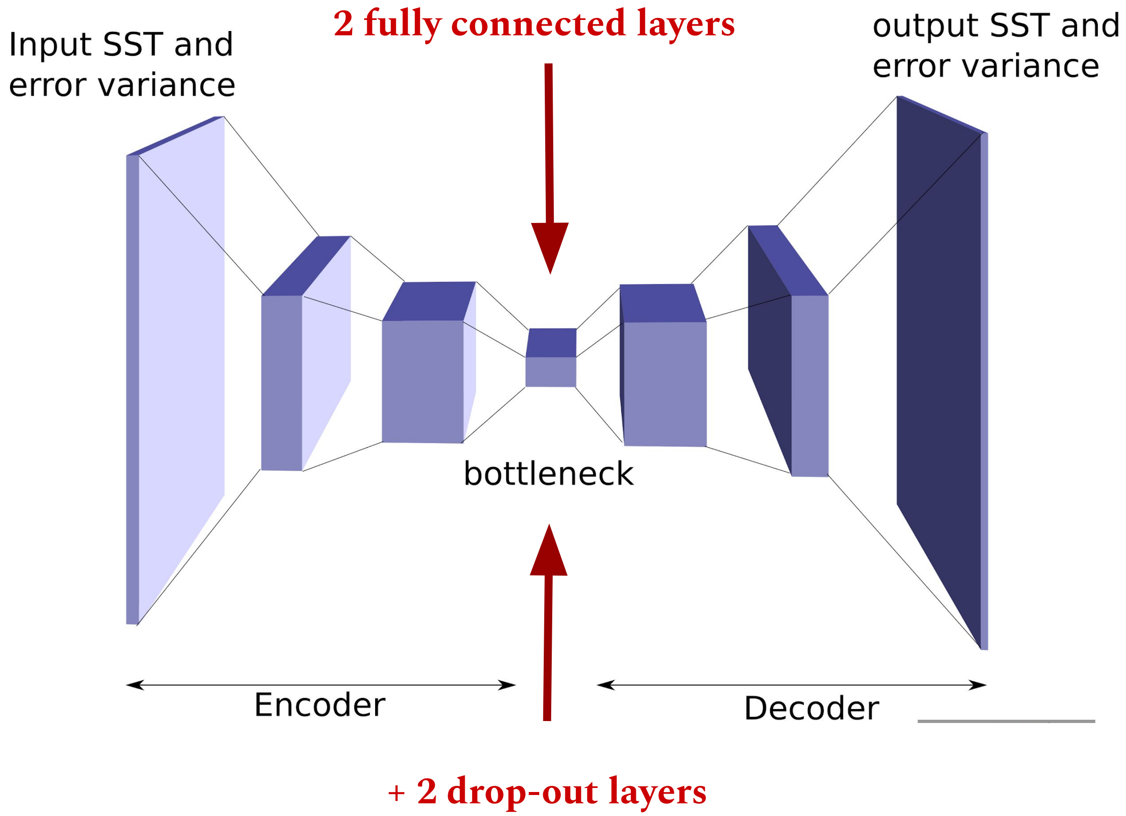
5 decoding layers

3x3 filters applied at each layer

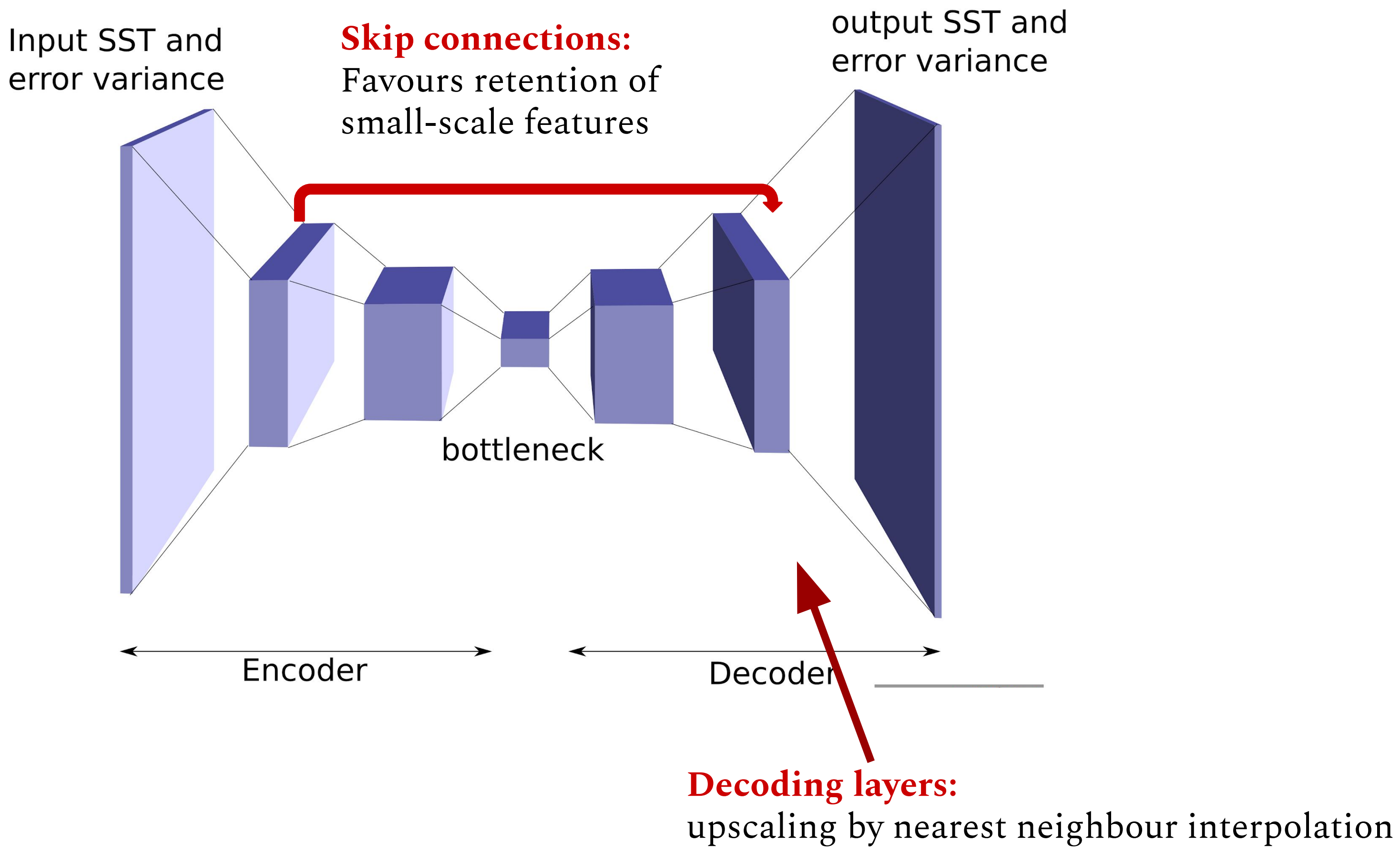


Average pooling layers

Reduce size by retaining the average
value on 2x2 boxes



Take out 30% of neurons (pixels) to avoid
overfitting


Input SST and error variance
Skip connections:
Favours retention of small-scale features
output SST and error variance
bottleneck
Encoder
Decoder
Decoding layers:
upscaling by nearest neighbour interpolation

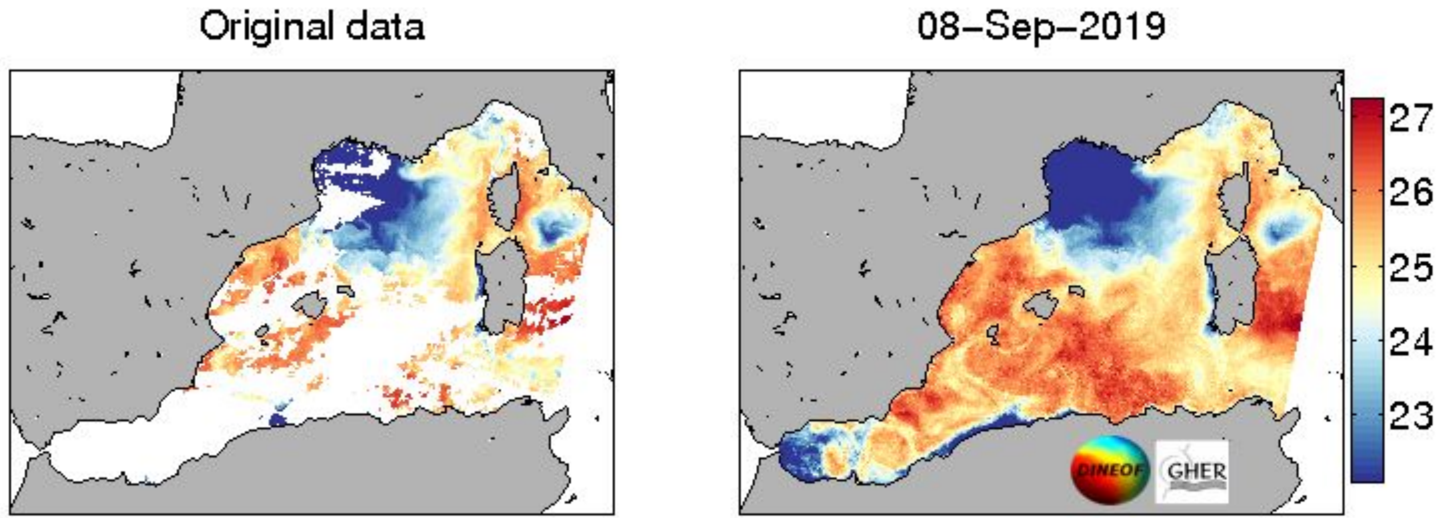
Baseline method to be improved

DINEOF (Data Interpolating Empirical Orthogonal Functions)

A reconstruction method based on the EOF basis from the dataset

~15 years of development & improvements

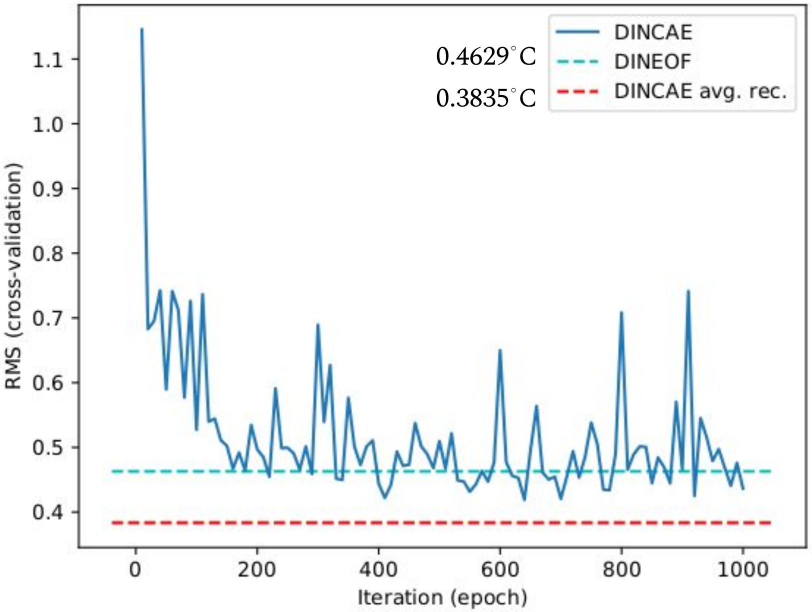
http://www.dineof.net/DINEOF/



Results

Cross-validation: data removed from the last 50 images of the times series (with cloud mask from first 50 images)

Averaging epochs 200 to 100 improved DINCAE results



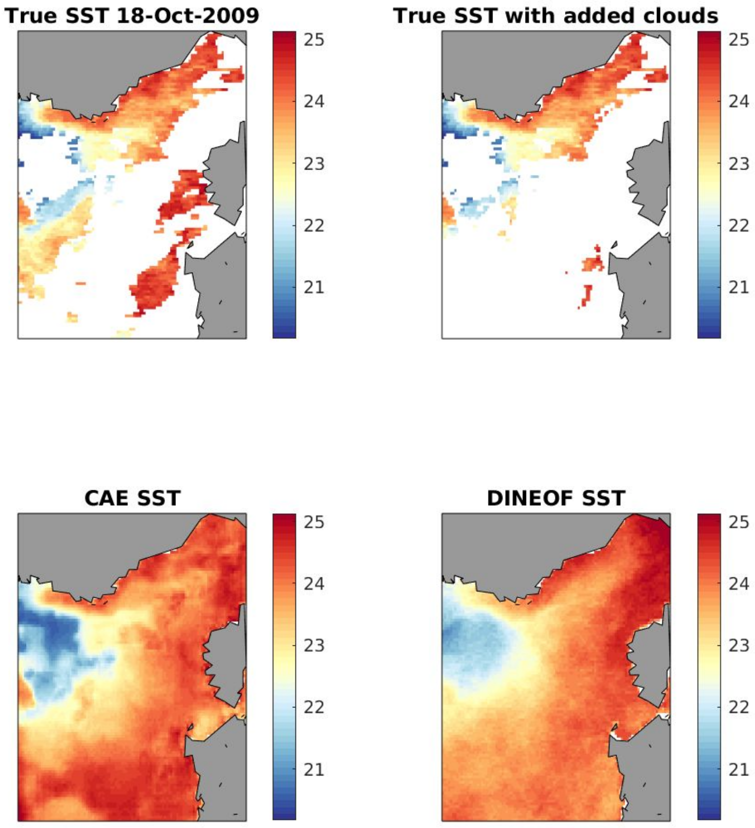
Reconstruction results -full time series- compared to WOD in situ data (under clouds)

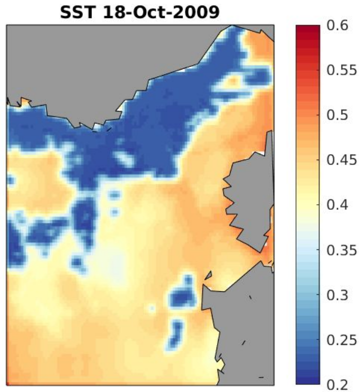
RMS (DINEOF) 1.1676°C

RMS (DINCAE) 1.1362°C

Results

Reconstruction examples



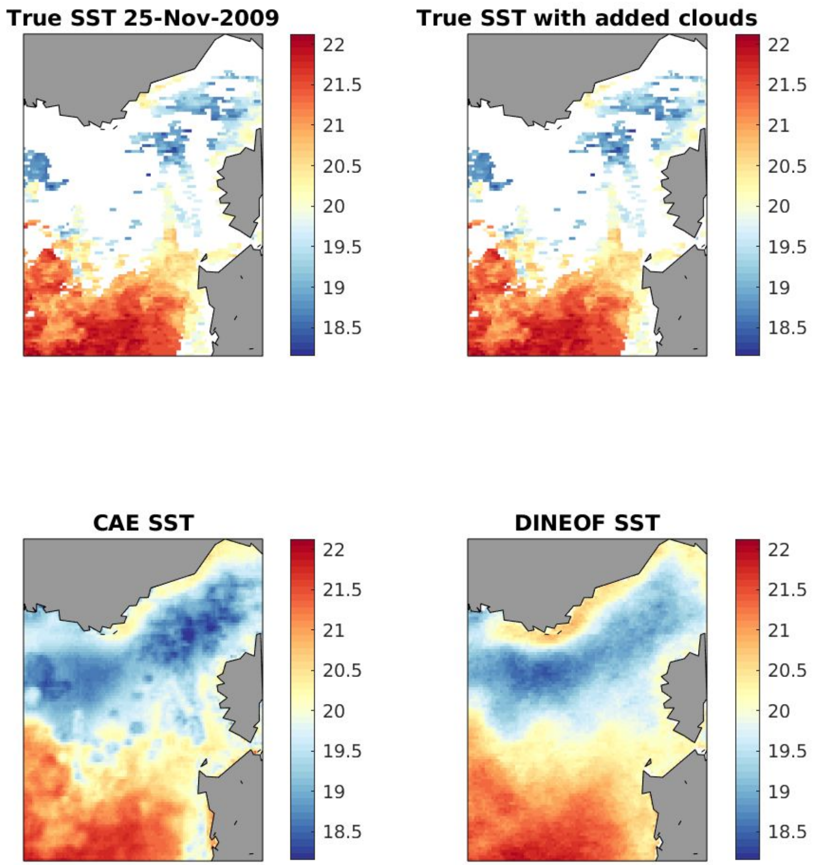


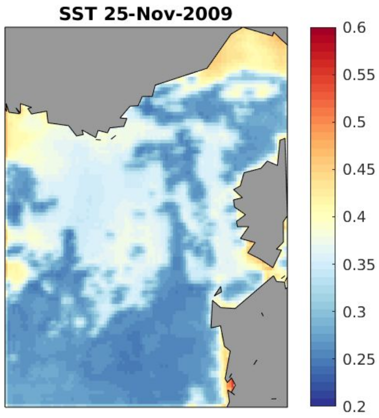
Expected error (std. dev.)

Small-scale variability

Results

Reconstruction examples





Expected error (std. dev.)

Some artifacts appear when too few data

If you want to know more...

- Manuscript under revision (open review) in GMD

- Code available at:

https://github.com/gher-ulg/DINCAE

Conclusions & future work

A convolutional Autoencoder approach to reconstruct missing data

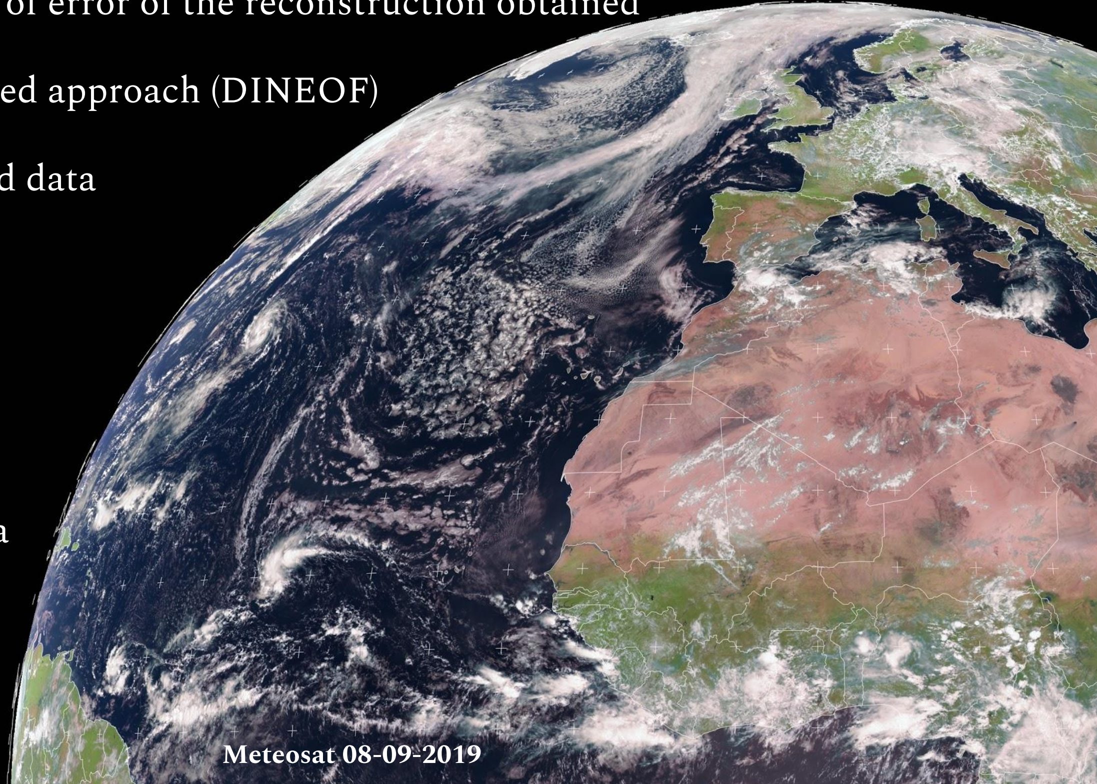
- Missing data handled by including expected error variance in the input data
- Estimation of missing data + estimation of error of the reconstruction obtained

Results similar (& slightly better) than established approach (DINEOF)

Small scale variability retrieved in reconstructed data

Future work includes:

- Multivariate analyses
- Work with higher spatial resolution data
- Use of other NN architectures



number	type	output size	parameters
1	input	112 x 112 x 8	
2	conv. 2d	112 x 112 x 16	n. filters = 16, kernel size = (3,3)
3	pooling 2d	56 x 56 x 16	pool size = (2,2), strides = (2,2)
4	conv. 2d	56 x 56 x 24	n. filters = 24, kernel size = (3,3)
5	pooling 2d	28 x 28 x 24	pool size = (2,2), strides = (2,2)
7	conv. 2d	28 x 28 x 36	n. filters = 36, kernel size = (3,3)
8	pooling 2d	14 x 14 x 36	pool size = (2,2), strides = (2,2)
9	conv. 2d	14 x 14 x 54	n. filters = 54, kernel size = (3,3)
10	pooling 2d	7 x 7 x 54	pool size = (2,2), strides = (2,2)
11	fully connected layer	529	
12	drop-out layer	529	drop-out rate for training = 0.3
13	fully connected layer	2646	
14	drop-out layer	2646	drop-out rate for training = 0.3
15	nearest neighbor interpolation	14 x 14 x 54	
16	concatenate output of 15 and 8	14 x 14 x 90	
17	conv. 2d	14 x 14 x 36	n. filters = 36, kernel size = (3,3)
18	nearest neighbor interpolation	28 x 28 x 36	
19	concatenate output of 18 and 5	28 x 28 x 60	
20	conv. 2d	28 x 28 x 24	n. filters = 24, kernel size = (3,3)
21	nearest neighbor interpolation	56 x 56 x 24	
22	concatenate output of 21 and 3	56 x 56 x 40	
23	conv. 2d	56 x 56 x 16	n. filters = 16, kernel size = (3,3)
24	nearest neighbor interpolation	112 x 112 x 16	
25	concatenate output of 24 and 1	112 x 112 x 26	
26	conv. 2d	112 x 112 x 2	n. filters = 2, kernel size = (3,3)