SSH Super-Resolution using high resolution SST with a Subpixel Convolutional Residual Network

Théo Archambault, Anastase Alexandre Charantonis, Dominique Béréziat, Carlos Mejia, Sylvie Thiria

Environmental Data Science (2022), 4: 1–9
doi:xxxx

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

SSH Super-Resolution using high resolution SST with a Subpixel Convolutional Residual Network

T. Archambault[1*], A. Charantonis[2,3], D. Béréziat[1], C. Mejia[2] and S. Thiria[2]

[1]Sorbonne Université, CNRS, LIP6, Paris, 75005, France
[2]Laboratoire D'Océanographie et du Climat: Experimentation et Approches Numeriques, Sorbonne University, Paris, 75005, France
[3]École nationale supérieure d'informatique, Evry, 91000, France
*Corresponding author. Email: theo.archambault@lip6.fr

Received xx xxx xxxx

Keywords: Super-Resolution, Subpixel convolution, Satellite Image

Abstract

The oceans have a very important role in climate regulation due to its massive heat storage capacity. Thus, for the past decades oceans have been observed by satellites in order to better understand its dynamics. Satellites retrieve several data with various spatial resolution. For instance Sea Surface Height (SSH) is a low-resolution data field where Sea Surface Temperature (SST) can be retrieved in a much higher one. These two physical parameters are linked by a physical relation that can be learned by a Super-Resolution machine learning algorithm. In this work we present a Subpixel Convolutional Deep learning model that takes advantage of the higher resolution SST field to guide the downscaling of the SSH one. The data fields that we use are simulated by a physic based ocean model at a higher sampling rate than the satellites provide. We compared our approach with a convolutional neural network (CNN) model. Our architecture generalized well with validation performances of 3.94 cm RMSE and training performances of 2.65 cm RMSE.

1. Introduction

The oceans, with their massive heat storage capacity and conveyor belt circulation are the prime drivers of climate regulation. Better understanding and monitoring their response to climate change requires high resolution datasets. Such datasets are usually obtained by satellite imaging, or by numerical modeling, or by a combination of the two through data assimilation. Satellite remote sensing provide a multitude of geophysical data fields with various sampling in space and in time. For instance Sea Surface Temperature (SST) can be obtained at a very high resolution (1.1 and 4.4 km) from the AVHRR instruments. In the other hand, Sea Surface Height (SSH) can be retrieved at a coarser resolution (around 25 km) from different satellites. SSH and SST variables are linked by a hidden physical relation that we aim to use by combining these multi resolution data in order to downscale the coarse SSH. This could lead to estimate ocean currents as they are formed by geostrophy from SSH gradient. This is a Super Resolution (SR) problem applied to oceanography.

Since 2014 and the Super Resolution Convolutional Neural Network (SRCNN) introduced by [2], neural networks have been used to perform Super Resolution tasks, and often achieve state-of-the-art performances [11]. A wide range of deep neural network architectures have been explored such as the residual networks (ResNet). ResNets allow the use of deeper architectures and bigger learning rates [6, 3]. Various up-sampling strategies have also been tested such as the subpixel convolution

method introduced by [8]. However these methods should be adapted to the SSH downscaling problem because the network must be able to extract information from the high resolution SST. This problem has been tackled by the RESAC neural network architecture [10] which is a proof-of-concept Super-Resolution architecture. Its main features are downscaling through consecutive resolutions, and using a cost function that monitors the correct downscaling in each of these resolutions. We propose a subpixel convolutional residual network, called RESACsub, a modified version of the RESAC network. As the RESAC network, RESACsub aims to downscale a low resolution SSH using information from a higher resolution SST. We achieve the SSH downscaling with a higher upsampling factor than in the RESAC proof of concept, but we only focused on the SSH, while RESAC also recovers the longitudinal and latitudinal velocities. In the following we will present RESACsub, as well as its main differences to RESAC, briefly detail the dataset used by both models, present the results obtained and discuss further potential developments.

2. Data

In order to train a supervised neural network as RESACsub we need the data fields at every resolution. As SSH cannot be retrieved from a satellite at the resolution we want to downscale to, we use simulated grided SSH and SST from an ocean physics-based model: NATL60 [1]. This model is based on the NEMO 3.6 [7] code, with atmospheric forcing, and initial conditions taken from MERCATOR [5]. We use the SSH and SST state variables of this model at a very high resolution denoted R01, (for resolution of 1/60° at the Equator). At this resolution the model has a very high running cost, therefore we were only able to retrieve one year of training data (366 days starting from 1st October 2012), and 4 months of validation in 2008 (March, June, September, December). We are well aware that we should use different test/validation data-set, but considering the importance of the annual cycle and that the main objective is to compare two models, we decided not to separate the 4 validation months. Compared to RESAC, our validation method is more rigorous as there cannot be data-leakage from sampling the validation data from the training year.

We simulate the various resolutions by recursively averaging the pixels in a 3×3 mask. We then call hereafter R01, R03, R09, R27, R81 the five resolutions that we study with respective size at the grid center of (1.5×1.5, 4.5×4.5, 13×14, 40×41, and $120 \times 122\,\mathrm{km}^2$). We renormalize the dataset between 0 and 1 to both stabilize the numerical calculations and equalize the importance of SST and SSH. In order to compare our models we perform ten trainings of each architecture, with different weight initialization on each training.

3. Method

3.1. RESAC Super-Resolution Method

In this paper we compare two CNN architectures that use the same downscaling RESAC method. This method aims to learn the hidden link between high resolution SST and a low resolution SSH. To that end we progressively increase the resolution of our coarse SSH_R81 in three up-sampling steps each one with a up-sampling factor of 3 for a total up-sampling factor of 27. To perform each resolution increase we use a CNN block as shown in the RESAC method in Figure 1.

The Vanilla RESAC method as proposed by [10] performed only two downscaling CNNs starting from SSH_R81 ($120 \times 122\,\mathrm{km}^2$) and retrieving SSH_R09 ($13 \times 14\,\mathrm{km}^2$) for a total up-sampling factor of 9. But the original method also used another CNN block at the end of the network to compute the circulation (U and V currents) from the estimate SSH_R09. Our downscaling method upscales the SSH from $120 \times 122\,\mathrm{km}^2$ down to $4.5 \times 4.5\,\mathrm{km}^2$ (upsampling factor of 27) without retrieving ocean currents. Both methods use cost functions that force the model to correctly downscale through the intermediary resolution. Therefore we use three loss functions at R27, R09, and R03 and the global loss of the model

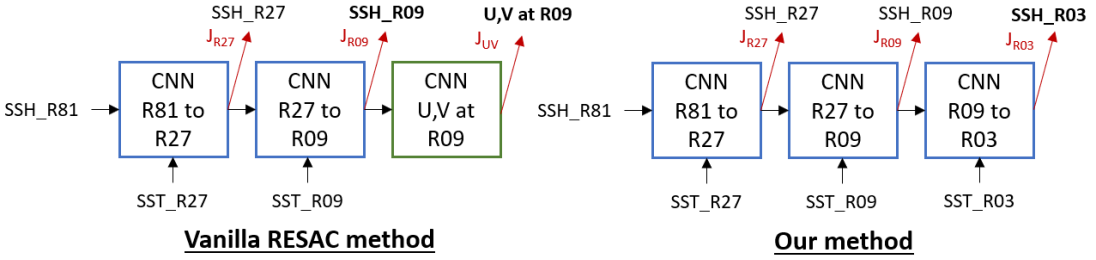

Figure 1. *Comparison between the vanilla downscaling method used in [10] and ours. The main two differences are that we increase the SSH resolution one step further but we do not retrieve the ocean circulation.*

is their sum. The use of three separate loss functions guides the model to correctly reconstruct the intermediary resolutions, which depend on different physical phenomena given their scales. For each of these loss functions we use mean squared error (MSE) between the estimated SSH and the target.

In this work every architecture uses the same downscaling method: a slightly different method that the one used by [10], that downscale SSH_R81 to SSH_R03 using every intermediate resolution SST and without retrieving ocean currents. We denote this downscaling method as the RESAC method hereafter.

3.2. Network architecture

3.2.1. RESAC network

The original RESAC architecture upsamples the SSH with a bilinear upsampling and then apply $Nloop$ times 2 convolution layers followed by a Batch Normalization (BN) layer as shown in Figure 2. In the following work we set $Nloop = 5$, therefore each downscaling CNN block has 10 convolution layers, each one with 37 filters. We use the swish activation function for every convolution layer, except for the last one that uses a linear activation function.

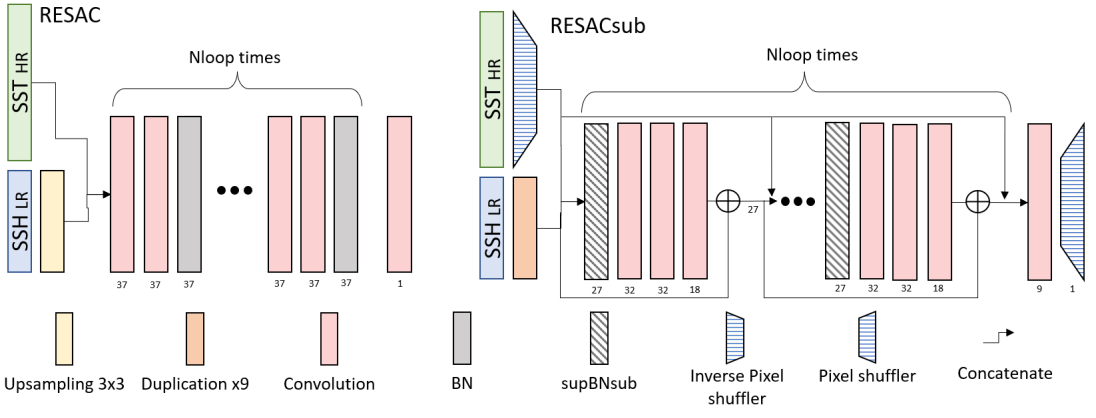

Figure 2. *Comparison of the architectures of one downscaling step for RESAC and RESACsub. For each layer the output number of channels is given below it.*

3.2.2. RESACsub network

RESACsub is a Subpixel Convolutional Residual Network. It is a post network upsampling strategy showed in Figure 2. The principle of a subpixel convolutional layer is to perform the convolution, not in the original image space (that we call supspace hereafer), but in a deeper and smaller space (that we call subspace hereafter). This method has been introduced by Shi *et al* [8]: the upsampling is therefore performed at the end of the network by getting back to the supspace. The supspace is obtained from the subpsace by apply a pixel suffler operation as showed in Figure 3. Two pixels that are spatial neighbors in the supspace are channels neighbors in the subspace. We call hereafter P the operation that transform a subspace image into a supspace image and P^{-1} the inverse operation, see Eq. (1),

$$I^{sub}_{n,n,R^2c} \underset{P^{-1}}{\overset{P}{\rightleftarrows}} I^{sup}_{Rn,Rn,c} \tag{1}$$

where R is the upsampling factor (3 in our work), n is the spatial dimension of the low resolution image, and c the number of channels. Therefore I^{sub}_{n,n,R^2c} is the image in the subspace and $I^{sup}_{Rn,Rn,c}$ is the same image shifted in the supspace.

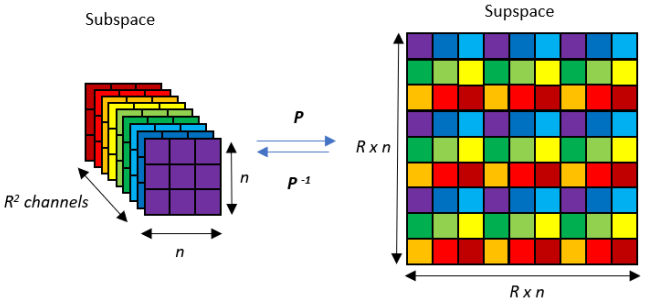

Figure 3. *Pixel shuffler and inverse pixel shuffler.*

In RESACsub we first concatenate the SST and the SSH in the subspace. As the SST is wider than the SSH, we first shift it to the subspace with an inverse pixel shuffling layer. To keep the same weight between SST ans SSH images we duplicate 9 times the low resolution SSH image and concatenates the two images after doing so. We then use a Residual Network to downscale the SSH, with 5 residual loop ($Nloop = 5$). Each residual loop starts with adapted form of Batch Normalization that we detail in 3.2.3, followed by 3 convolution layers with respectively 32, 32 and 18 filters and the swish activation function. We then add the SSH and concatenate the SST. The final output layer is a convolution layer with 9 filters and a linear activation function followed by a Pixel.

3.2.3. Batch Normalization

With this subpixel convolution method, we use the channel dimension to store neighbors pixels and then perform a convolution layer. This implies that using Batch Normalization (called BN hereafter) will create strong checkerboard artifacts as each channel is normalized with a different mean and variance. The tests we performed confirmed this. To explain the impact of batch normalization on the checkerboard artifacts we must get back to the following equation:

$$Y_{b,x,y,c} = \gamma_c \frac{X_{b,x,y,c} - \mu_c}{\sqrt{\sigma_c^2 + \epsilon}} + \beta_c \tag{2}$$

where X and Y are respectively the input and the output of the batch normalization layer. They are 4-dimension tensors with b as the batch coordinate, x and y the two spatial coordinates of the image and c the channel index. Parameters μ_c and σ_c are respectively the mean and the standard deviation of the image across the two space dimensions and the batch dimension for the channel c. Finally, ϵ is a small positive constant that we add for numerical stability, and β_c and γ_c are the output mean and the output standard deviation that are learned during the training.

The batch normalization method thus learns the bias and the standard deviation of each channel independently during training phase. This is inconsistent with the subpixel convolutional method because the channels do not represent different data nor images, but neighbors pixels that have similar distributions and should be normalized the same way.

To fix this issue we use an adapted form of batch normalization that we call *supBNsub* as illustrated in Figure 4. Operator *supBNsub* gets back to the supspace, applies a BN layer and then returns to the subspace. If we call *BN* the Batch Normalization operation described in Eq. (2), we can write the *supBNsub* operator as:

$$supBNsub = P^{-1}(BN(P)) \tag{3}$$

SupBNsub method produces better results in RMSE sense.

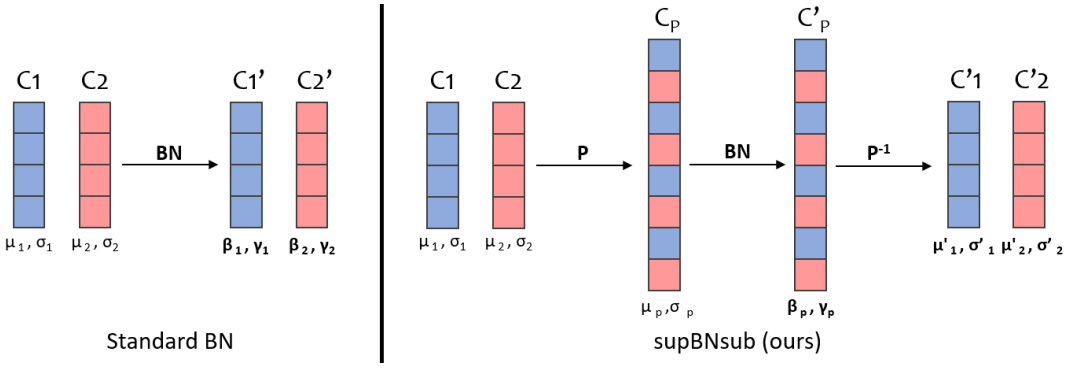

Figure 4. *Comparison of the two Batch Normalization methods for a 1-dimension example. For each channel we write the mean and standard deviation below it and the learned parameters are in bold.*

To explain why Batch Normalization creates checkerboard artifacts, we must write the mean and standard deviation of the both method. It is trivial that in the case of the standard Batch Normalization, the C_1' channel has a mean of β_1 and a standard deviation of γ_1 (respectively β_2 and γ_2 for the C_2' channel). According to notations in Figure 4, in the supBNsub case we have (see Appendix A for details):

$$\mu_p = \frac{\mu_1 + \mu_2}{2} \qquad \sigma_p^2 = \frac{\sigma_1^2 + \sigma_2^2}{2} + \left(\frac{\mu_1 - \mu_2}{2}\right)^2 \qquad \mu_1' = \gamma_p \frac{\mu_1 - \mu_p}{\sigma_p} + \beta_p \qquad \sigma_1'^2 = \frac{\gamma_p^2 \sigma_1^2}{\sigma_p^2} \tag{4}$$

In both cases, the output mean and standard deviation of the C_1' and the C_2' channels are not equal. During the upspamling operation to get back to the supspace, neighbors pixels will have slightly different values due to their different means, therefore some checkerboard artifacts will appear as shown in Figure 6. However the supBNsub layer performs better than the standard Batch Normalization because it has a higher regularization effect as all the channels are normalized the same way. The supBNsub layer has less degrees-of-freedom where the standard BN bias and standard deviation are completely unrelated between two different channels.

The checkerboard artifacts are intrinsically linked to the subpixel convolutional layers, as different filters weights are applied to neighbor pixels. To get rid of this problem we use a post-trained denoising

filter: at the end of RESACsub, we apply 2 convolution layers with 32 filters with a 7×7 kernel size and a ReLu activation function. This convolution operation is wide enough to perceive a 12×12 area where several checkerboard patterns should be repeated. After these 2 layers we perform a last convolution layer with one filter 1×1 and no activation to get back to the image dimension.

4. Results

We compare the results of 4 models on the validation data set. We call hereafter RESAC is the network presented in [10], RESACsub supBNsub is the proposed network with the supBNsub Batch Normalization layer, RESACsub BN is the same network but with the standard BN, and Denoiser stands for the denoising filter applied after the RESACsub supBNsub network. For each upsampling architecture (a CNN block in Figure 1), both models have 5 loops ($Nloop = 5$) in Figure 2. For RESAC, that correspond to 10 convolution layers with 37 filters, and for RESACsub it correspond to 10 convolution layers with 32 filters and 5 with 18 filters. We adjusted the number of filters in RESAC up to 37 so that the two models are comparable in terms of weight number. However, RESACsub is still around 5 times faster to compute. We train each network 10 times with different initialization, so the scores presented in Table 1 are the mean ± standard deviation on the different training. The RESAC, RESACsub supBNsub and RESACsub BN networks are trained with a batch size of 32 and a adaptive learning rate. On the other hand the Denoiser is trained with a batch of a single image.

The proposed model RESACsub with the supBNsup layer outperforms both RESAC and the RESACsub with a standard BN. The denoising network improves the performances of RESACsub supBNsub on RMSE but also on the visual aspect by removing the checkerboard artifacts as the Figure 6 shows. We also compare the RMSE on the first and the last decile of the ground truth i.e. on the spatial location where the SSH is the higher and the lower. As expected all the networks have a higher RMSE on the first decile than on the global image. This is because the North zone that has a low SSH is very energetic with very strong currents. Therefore the SSH variations are more important and less predictable. In Table 1 it is clear that most of the error is made in the North zone (corresponding to the 1^{st} decile).

Model	RESAC	RESACsub BN	RESACsub supBNsub	Denoiser
weights	344,976	335,442	334,722	51,841
RMSE (cm)	5.50 ± 0.47	4.43 ± 0.14	4.00 ± 0.26	3.94 ± 0.28
RMSE 1^{st} dec (cm)	8.03 ± 0.91	5.33 ± 0.76	5.09 ± 0.93	5.03 ± 0.99
RMSE 10^{th} dec (cm)	4.65 ± 0.14	4.78 ± 0.22	4.42 ± 0.12	4.36 ± 0.12
RMSE cropped (cm)	5.24 ± 0.48	4.22 ± 0.14	3.82 ± 0.25	3.80 ± 0.27

Table 1. *Mean and standard deviation scores on 10 trainings of each architecture with different weight initializations. The scores are given on the validation data set. We compare the models in RMSE (root mean squared error): the global RMSE is given, along with the RMSE on the first and the last decile of the target image. We also give a cropped RMSE (the RMSE of a smaller interior image to avoid boarder effects)*

5. Conclusion

We have proposed RESACsub, a subpixel convolutional residual network that outperforms RESAC in the Super Resolution task of downscaling SSH with high resolution SST. Our method (RESACsub + denoising network) achieve a downscaling of 27 upsampling factor (from a resolution of 120×122 km^2 to

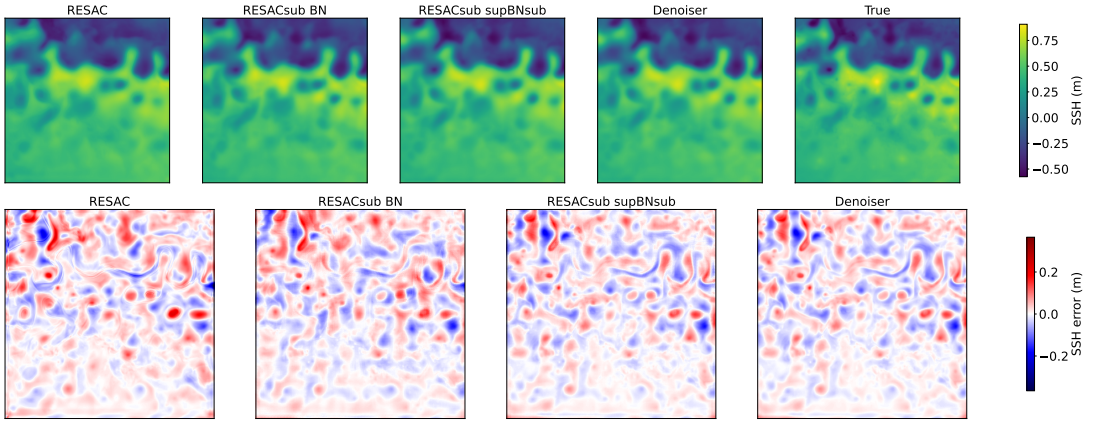

Figure 5. *Network output for The SSH at R03. The first line is the estimate SSH of the same day (10^{th} of March), where the second line is the error map associated .*

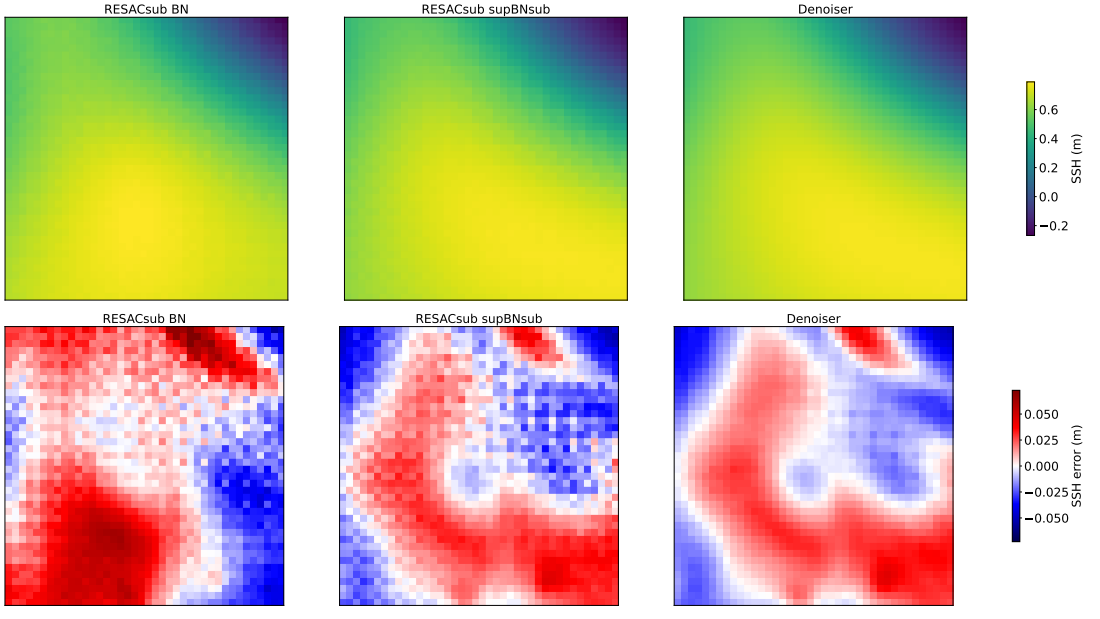

Figure 6. *Zoom on the error map of the 2 subpixel models RESACsub BN and RESACsub supBNsub, and the denoising network applied on RESACsub supBNsub. We can clearly see the* 3×3 *pattern on the 2 subpixel model where the denoiser removes it.*

$4.5 \times 4.5\,\mathrm{km}^2$) with a RMSE of 3.94 cm. We have compared two forms of Batch Normalization: supBN-sub, a subpixel adapted form of Batch Normalization and the standard BN. We show that the supBNsub method has a higher regularizing power than the standard one that result in a performance improvement. However the subpixel convolutional method has some drawback: it creates strong checkerboard artifacts on the output image. We were able to get rid of most of the cherckerboard artifacts with a two layers denoising network that learns the 3×3 pattern and successfully denoise the SSH with a 0.06 cm RMSE improvement.

Going further with this work, we will push our approach further by not limiting it to a number of parameters similar to the ones in RESAC, which we suspect will yield even better results. We also intend to test and adapt our approach with other state-of-the-art architectures. Using the RESACsub Network as a generator for a Conditional Generative Adversarial Neural network (GAN) similar to the SRGAN model introduced by [4] could improve its performance, given the advantages of using a discriminator network as a cost function for image reconstruction tasks. Further work also includes adapting this method to real world satellite data using transfer learning to retain the skill obtained over the numerical model's dataset.

References

1. A. Ajayi, J. Le Sommer, E. Chassignet, J.-M. Molines, X. Xu, A. Albert, and E. Cosme. Spatial and temporal variability of north atlantic eddy field at scale less than 100 km. *Earth and Space Science Open Archive*, page 28, 2019.
2. C. Dong, C.C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, pages 184–199, 2014.
3. J. Kim, J.K. Lee, and K.M. Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016.
4. C. Ledig, L. Theis, H. Huszár, J. Caballero, A. Cunningham, A. Acosta, A.P. Aitken, A. Tejani, J. Totz, and Z. Wang. Photo-realistic single image super-resolution using a generative adversarial network. *CVPR*, pages 4681–4690, 2016.
5. J.-M. Lellouche, E. Greiner, O. Le Galloudec, C. Regnier, M. Benkiran, C.-E. Testut, R. Bourdalle-Badie, M. Drevillon, G. Garric, and Y. Drillet. Mercator ocean global high-resolution monitoring and forecasting system. *New Frontiers in Operational Oceanography*, pages 563–592, 2018.
6. B. Lim, S. Son, H. Kim, S. Nah, and K. Lee. Enhanced deep residual networks for single image super-resolution. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1132–1140, 2017.
7. G. Madec, R. Bourdallé-Badie, P.-A. Bouttier, C. Bricaud, D. Bruciaferri, D. Calvert, J. Chanut, E. Clementi, A. Coward, D. Delrosso, et al. Nemo ocean engine, 2017.
8. W. Shi, J. Caballero, F. Huszar, J. Totz, A. Aitken, R. Bishop, R. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, pages 1874–1883, 2016.
9. W. Shi, J. Caballero, L. Theis, F. Huszar, A. Aitken, A. Tejani, J. Totz, C. Ledig, and Z. Wang. 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, 2016.
10. S. Thiria, C. Sorror, C. Mejia, J.-M. Molines, and M. Crepon. Downscaling of ocean fields by fusion of heterogeneous observations using deep learning algorithms. *Submitted to Ocean Modeling, currently in revision*, 2022.
11. Z. Wang, J. Chen, and S. Hoi. Deep learning for image super-resolution: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 3365–3387, 2021.

A. Appendix: Proof of the mean and variance of the supBNsub layer

$$\begin{aligned}
\sigma_p^2 &= \frac{1}{2n}\sum_{i=1}^{2n}(C_p(i)-\mu_p)^2 \\
&= \frac{1}{2n}\left[\sum_{i=1}^{n}(C_1(i)-\mu_p)^2+\sum_{i=1}^{n}(C_2(i)-\mu_p)^2\right] \\
&= \frac{1}{2n}\left[\sum_{i=1}^{n}\left(C_1(i)-\mu_1+\frac{\mu_2-\mu_1}{2}\right)^2+\sum_{i=1}^{n}\left(C_2(i)-\mu_2+\frac{\mu_1-\mu_2}{2})\right)^2\right] \\
&= \frac{\sigma_1^2+\sigma_2^2}{2}+\left(\frac{\mu_2-\mu_1}{2}\right)^2+\frac{1}{2n}\left[\sum_{i=1}^{n}2\left(C_1(i)-\mu_1\right)\left(\frac{\mu_2-\mu_1}{2}\right)+\sum_{i=1}^{n}2\left(C_2(i)-\mu_2\right)\left(\frac{\mu_1-\mu_2}{2}\right)\right] \\
&= \frac{\sigma_1^2+\sigma_2^2}{2}+\left(\frac{\mu_2-\mu_1}{2}\right)^2
\end{aligned} \tag{A.1}$$

Because the BN do not changes the order of the pixels we can see that:

$$C_1'(i) = \gamma_p\frac{C_1(i)-\mu_p}{\sigma_p}+\beta_p$$

Then we can write μ_1' and σ_1' as:

$$\begin{aligned}
\mu_1' &= E[C_1'] = \gamma_p\frac{E[C_1]-\mu_p}{\sigma_p}+\beta_p = \gamma_p\frac{\mu_1-\mu_p}{\sigma_p}+\beta_p \\
\sigma_1' &= Var[C_1'] = \frac{\gamma_p^2}{\sigma_p^2}Var[C_1] = \frac{\gamma_p^2\sigma_1^2}{\sigma_p^2}
\end{aligned} \tag{A.2}$$

Acknowledgments. We are grateful for the technical assistance of Pr. M.Crépon.

Funding Statement. This research was supported by grant for T.Archambault PhD from Sorbonne Université.

Competing Interests. None

Data Availability Statement. The SARGAS60 datasets we used (extraction of the CJM155 and MJM155 NATL60 model experiments over the Gulf steam area, in the North Atlantic Ocean) are already available in the IPSL Thredds catalog at the address : {https://doi.org/10.14768/c3c33afe-2a37-42e1-bb29-21428387f658}.

Ethical Standards. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

Author Contributions. Conceptualization: T.Archambault; A.Charantonis; D.Béréziat; S.Thiria
Methodology: T.Archambault; A.Charantonis
Data curation: C.Meija
Data visualization: T.Archambault
Project administration: D.Béréziat; S.Thiria
Writing original draft: T.Archambault
Writing – review & editing: T.Archambault; D.Béréziat; A.Charantonis; S.Thiria
All authors approved the final submitted draft.

Supplementary Material. Supplementary material can be found in https://gitlab.lip6.fr/archambault/resacsub.git