Improving trajectory calculations by FLEXPART 10.4+ using single image superresolution

Rüdiger Brecht[1], Lucie Bakels[2], Alex Bihlo[3], and Andreas Stohl[2]

[1]Department of Mathematics, University of Bremen
[2]Department of Meteorology and Geophysics, University of Vienna, Josef-Holaubek-Platz 2
[3]Department of Mathematics and Statistics, Memorial University of Newfoundland

Correspondence: Rüdiger Brecht (rbrecht@uni-bremen.de)

Abstract. Lagrangian trajectory or particle dispersion models as well as semi-Lagrangian advection schemes require meteorological data such as wind, temperature and geopotential at the exact spatio-temporal locations of the particles that move independently from a regular grid. Traditionally, this high-resolution data has been obtained by interpolating the meteorological parameters from the gridded data of a meteorological model or reanalysis, e.g. using linear interpolation in space and time. However, interpolation errors are a large source of error for these models. Reducing them requires meteorological input fields with high space and time resolution, which may not always be available and can cause severe data storage and transfer problems. Here, we interpret this problem as a single image superresolution task. That is, we interpret meteorological fields available at their native resolution as low-resolution images and train deep neural networks to up-scale them to higher resolution, thereby providing more accurate data for Lagrangian models. We train various versions of the state-of-the-art *Enhanced Deep Residual Networks for Superresolution (EDSR)* on low-resolution ERA5 reanalysis data with the goal to up-scale these data to arbitrary spatial resolution. We show that the resulting up-scaled wind fields have root-mean-squared errors half the size of the winds obtained with linear spatial interpolation at acceptable computational inference costs. In a test setup using the Lagrangian particle dispersion model FLEXPART and reduced-resolution wind fields, we find that absolute horizontal transport deviations of calculated trajectories from "true" trajectories calculated with undegraded $0.5° \times 0.5°$ winds are reduced by at least 49.5% (21.8%) after 48 hours relative to trajectories using linear interpolation of the wind data when training on $2° \times 2°$ to $1° \times 1°$ ($4° \times 4°$ to $2° \times 2°$) resolution data.

1 Introduction

Recent years have seen a considerable increase of interest in the application of machine learning to virtually all areas of the natural sciences, with meteorology being no exception. Machine learning, and specifically deep learning, which is concerned with training deep artificial neural networks, holds great promise for problems for which vast amounts of data are available (LeCun et al., 2015). This is the case for meteorology where numerical weather prediction and observations generate a large amount of data. Breakthroughs in the availability of affordable graphics processing units and substantial improvements in training algorithms for deep neural networks have equally contributed to making deep learning a promising new tool for applications in computer vision (Krizhevsky et al., 2012), speech generation (Oord et al., 2016), text translation and generation (Vaswani et al., 2017), and reinforcement learning (Silver et al., 2017). Applications of deep learning to meteorology so far include weather nowcasting (Shi et al., 2015), weather forecasting (Rasp et al., 2020; Weyn et al., 2019), ensemble forecasting (Bihlo, 2021; Brecht and Bihlo, 2022; Scher and Messori, 2018), subgrid-scale parameterization (Gentine et al., 2018), and downscaling (Mouatadid et al., 2017).

In recent years image super resolution by neural networks has made considerable progress. The main application is to scale a low resolution image to an image with a higher resolution, which is referred to as *single image superresolution* (SISR), although similar techniques are also used to up-scale both the spatial resolution and the frame rates for videos as well. Before the advent of efficiently trainable convolutional neural networks (i.e., neural networks whose layers are convolutions, which put the input images through a set of filters, each of which activates certain features from the input), the superresolution problem for images was solved using interpolation based methods, such as in the paper of Li and Orchard (2001). These interpolation methods are still the state of the art for Lagrangian models.

SISR is a topic of substantial interest in computer vision, with applications in computational photography, surveillance, medical imaging and remote sensing (Chen et al., 2022). A variety of architectures have been proposed in this regard, essentially all of which use a convolutional neural network architecture, following the seminal contribution of Krizhevsky et al. (2012) which kindled the explosive interest in modern deep learning. Among these SISR architectures, some important milestones are Super-Resolution Convolutional Neural Network (SRCNN) (Dong et al., 2014), a standard convolutional neural network, Very Deep Super Resolution (VDSR) (Kim et al., 2016), a convolutional neural network based on the popular Visual Geometry Group (VGG) architecture (a standard deep convolutional neural network architecture with multiple layers), Super Resolution Generative Adversarial Network (SRGAN) (Ledig et al., 2017), a generative adversarial network, and EDSR (Lim et al., 2017), based on a convolutional residual network architecture. For a recent review on SISR providing an overview over the aforementioned architectures and others, the reader may wish to consult Yang et al. (2019).

While deep learning has been used extensively over the past several years in meteorology for a variety of use cases, there have only been a few applications of deep learning to meteorological interpolation that go beyond downscaling. This is surprising, as many tasks in numerical meteorology routinely involve interpolation, such as the time-stepping in numerical models using the semi-Lagrangian method requiring trajectory origin interpolation (Durran, 2010), or Lagrangian particle models (Stohl et al., 2005).

Semi-Lagrangian advection schemes in numerical weather prediction models rely on simple interpolation methods for the wind components (Durran, 2010). For instance, the semi-Lagrangian scheme in the Integrated Forecast System model of the European Centre for Medium Range Weather Forecasts (ECMWF) uses a linear interpolation scheme. In trajectory models and Lagrangian particle dispersion models, similarly simple interpolation methods are used. Higher-order interpolation schemes such as bicubic interpolation can reduce the wind component interpolation errors compared to linear interpolation (Stohl et al., 1995). However, error reductions for higher-order schemes are less than 30%, while computational costs increase by about an order of magnitude (Stohl et al., 1995). Therefore, many trajectory and Lagrangian particle dispersion models still use linear interpolation, e.g., FLEXPART (Pisso et al., 2019), LAGRANTO (Sprenger and Wernli, 2015), or MPTRAC (Hoffmann et al., 2021).

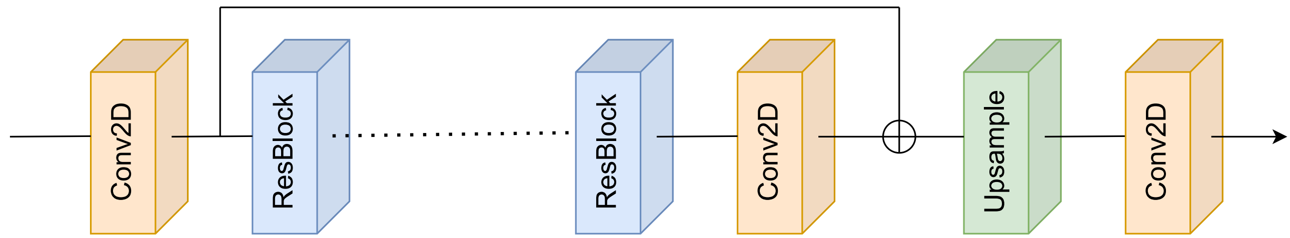
The purpose of this paper is to implement variable-scale superresolution based on deep convolutional neural networks to demonstrate their potential for Lagrangian models. Here, we make use of the self-similarity of meteorological fields, such that a neural network can be repeatedly applied to interpolate a velocity field to higher resolutions. The paper's further organization is as follows. Section 2 describes the numerical setup and the data being used in this work. In Section 3 we present the results of our study, illustrating substantial improvements of both the quality of up-scaled wind fields using the EDSR model in comparison to standard linear interpolation, as well as of trajectory calculations using the Lagrangian particle dispersion model FLEXPART (Pisso et al., 2019). A summary of this paper and thoughts for future research can be found in Section 4.

2 Methods

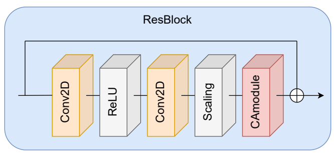
The aim of this study is to train a neural network which can then be used to interpolate meteorological velocity fields. For the training and evaluation we consider that meteorological fields are characterized by self-similarity over a range of spatio-temporal scales. This means that the structure of the field from one resolution to a higher one is similar. This makes it possible to train the neural network model to increase the resolution from a down-sampled velocity field to a higher resolution and then apply the model repeatedly to obtain even higher resolutions. Below we introduce the data used to train the neural network, describe the details of the neural network model and how we train the model. Moreover, we explain how we used the interpolated fields to run a simulation with FLEXPART.

2.1 Neural network architecture

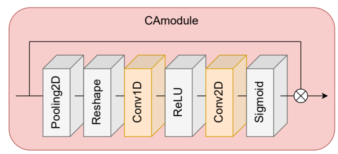
The neural network is designed to take a (n, n) matrix as an input and predict a $(2n, 2n)$ matrix as output. Doing this repeatedly allows one to increase the size of the meteorological fields arbitrarily. We use the Enhanced Deep Residual Network for Single Image Super-Resolution (EDSR) architecture (Lim et al., 2017) with additional channel attention, which gives higher importance to specific channels over others. The main building block of this architecture is a simplified version of a standard convolutional residual network block without batch normalization (Fig. 1b). This residual block consists of two convolutional layers, each of which uses a filter size of 3 in the present work, with the first convolutional layer being followed with a standard rectified linear unit activation function. After the second convolutional layer, a scaling of the output feature maps is performed, where we use the same constant residual scaling factor of 0.1 as proposed in the original work of Lim et al. (2017). The final operation of each residual block is given via channel attention (Fig. 1c). The overall architecture of this channel attention module follows Choi et al. (2020). The purpose of attention mechanisms in a convolutional neural network is to enable it to focus on the most important regions of the neural network's perceptive field. Channel attention re-weights each respective feature map from a convolutional layer following a learn-able scale transformation. The last building block of our architecture is the upsampling module (Fig. 1d). This module consists of a convolutional layer with a total of 64 $\times$ $\texttt{upscale\_factor}^2$ feature maps, followed by a depth-to-space transformation called PixelShuffle (Shi et al., 2016) which re-distributes feature maps into an `upscale_factor`-times larger spatial field. In this work, $\texttt{upscale\_factor} = 2$.



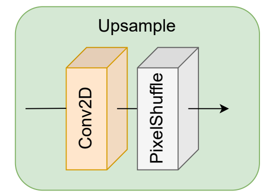
(a) The complete model architecture



(b) The residual block



(c) The channel attention module in the residual block.



(d) The upsampling block.

Figure 1. The overall layout of the neural network model (panel a) consists of an upsampling block (panel d) and residual blocks (panel b), which again contain a channel attention module (panel c). Here, $\bigoplus$ means adding the layers and $\bigotimes$ means multiplying them. The dotted line indicates that the residual block (panel b)is repeated multiple times.

The main residual network blocks are repeated 8 times, with an extra skip connection being added before the first convolutional layer in the network and before the upsample module. The upsampled image passes all convolutions in our architecture, with the exception of the convolutional layer in the upsample module using a total of 64 filters.

We have experimented with a variety of other architectures, including a conditional convolutional generative adversarial network called `pix2pix` (Isola et al., 2017) and the super-resolution GAN (Ledig et al., 2017), but have found the EDSR network giving the lowest interpolation error and having the shortest training time. Hence we only report the results from the EDSR model below.

2.2 Training data

To train our neural networks, we use data from the ECMWF ERA5 reanalysis (Hersbach et al., 2020). These global data are available hourly at a spatial resolution of $0.5° \times 0.5°$ in latitude–longitude coordinates and with a total of 138 vertical levels (level indexes increase upward, contrary to ECMWF). We use a total of 296 hours (from January 1 to January 12, 2000) for training and test our model for 24 hours in each season (on January 15, April 15, July 15, October 15, 2000). Each hourly horizontal layer data point corresponds to a total of 138 layers, yielding a total of roughly 15000 sample fields.

The low-resolution data are obtained from the high-resolution ERA5 data by simply sampling every 1st, 2nd and 4th degree. Notice that our approach of subsampling is driven by our goal to use the neural network as an interpolation algorithm, as required by kinematic trajectory calculations. From a dynamical point of view this is not the ideal approach, since averaged winds would give a dynamically more consistent representation of the flow. However, the goal here is not to provide the dynamically most consistent wind field but rather to obtain the best locally interpolated wind. Furthermore, averaging wind data would reduce the interpolation distance, since averaged winds would always be valid for locations very close to the location we want to interpolate to. For subsampling every 1st degree, for instance, we would need to interpolate only over one quarter of a degree instead of half a degree, thus providing a much less rigorous test against the true wind. Finally, averaging winds is also not ideal for our purpose because the comparison of "true" winds and reconstructed winds would reflect the effect of both averaging and interpolating, rather than interpolation alone.

In this work we focus solely on the spatial upscaling problem. The temporal upscaling problem will be considered elsewhere; see further discussions in the conclusions. We also only interpolate the horizontal wind components, and interpolate only horizontally.

2.3 Neural network training

For each horizontal velocity component (u and v) we train a separate neural network to interpolate a field from degraded resolution data to double its resolution data. Then, (if necessary) the trained neural network is applied multiple times to interpolate $0.5°$ resolution data. We train three sets of neural networks:

`model1` to interpolate a field from degraded $1° \times 1°$ resolution data to $0.5° \times 0.5°$ resolution data.

`model2` to interpolate a field from degraded $2° \times 2°$ resolution data to $1° \times 1°$ resolution data.

`model4` to interpolate a field from degraded $4° \times 4°$ resolution data to $2° \times 2°$ resolution data.

For the evaluation we only use models 2 and 4 since those are the only models that allow us to apply to resolution not seen during training. Model 1 is trained using the highest resolution data and we can not therefore use it to upscale to a higher resolution as we do not have the associated data for comparison.

We found that the wind field structure is different at higher and lower levels in the atmosphere and it is thus beneficial to train separate neural networks for these two regions of the atmosphere. Thus, one neural network is trained for the lower atmospheric levels (up to level 50, approximately below the tropopause) and another one is trained for the higher levels (51 to 138). Each neural network is trained on the data of 294 hourly wind fields with 50 and 88 vertical levels. This results in 14700 or 25578 training samples, respectively.

The model was implemented using TensorFlow 2.8 and is made available in a repository[1]. We trained the model on a dual NVIDIA RTX 8000 machine, with each training step taking approximately 100 ms for the u and v field. Total training took roughly 2.5 hours for each field.

[1] https://zenodo.org/record/7350568

2.4 Interpolation error metrics

To evaluate the accuracy of the interpolation and trajectory simulation we use the root mean square error (RMSE) and the structural similarity index measure (SSIM) as performance measures. The RMSE is defined as

$$\mathrm{RMSE}_z = \sqrt{\overline{(z_{\mathrm{interpolated}} - z_{\mathrm{reference}})^2}}, \tag{1}$$

where $z \in \{u, v\}$, with the bar denoting spatial averaging. The reference solution is given by the original $0.5° \times 0.5°$ ERA5 data, assumed to represent the truth. The smaller the RMSE value, the better the interpolated results coincide with the original reference solution.

The SSIM is a measure of the perceived similarity of two images a, b and is defined as

$$\mathrm{SSIM}(a,b) = \frac{(2\mu_a\mu_b + C_1)(2\sigma_{ab} + C_2)}{(\mu_a^2 + \mu_b^2 + C_1)(\sigma_a^2 + \sigma_b^2 + C_2)}. \tag{2}$$

with μ_a and μ_b denoting the means of the two images a and b (computed with an 11×11 Gaussian filter of width 1.5), σ_a and σ_b denoting their standard deviations and σ_{ab} being their co-variance. The constants C_1 and C_2 are defined as $C_1 = (K_1 L)^2$ and $C_2 = (K_2 L)^2$, respectively, with $K_1 = 0.01$ and $K_2 = 0.03$ and $L = 1$. The closer the SSIM value is to 1, the more similar the two images are. See (Wang et al., 2004) for further details. In the following, $a = z_{\mathrm{interpolated}}$ and $b = z_{\mathrm{reference}}$, with each of them being interpreted as a gray-scale image.

2.5 Trajectory calculations

To test the impact of the neural network interpolated wind fields on trajectory calculations, we used the Lagrangian particle dispersion model FLEXPART (Pisso et al., 2019; Stohl et al., 2005). The calculations are conducted using a stripped down version 10.4 of FLEXPART. We switched off all turbulence and convection parameterizations and used FLEXPART as a simple trajectory model. Ideally, the neural network interpolation should be implemented directly in FLEXPART. However, as the neural network and FLEXPART run on different computing architectures (Graphics vs. Central Processing Unit), directly implementing the neural network interpolation into FLEXPART is outside of the scope of this exploratory study. Instead, we replaced the gridded ERA5 wind data with the gridded up-sampled testing data produced by the neural network. Using gridded up-sampled testing data does not make full use of the neural network capabilities, since the neural network only produces values at a fixed resolution of $0.5° \times 0.5°$ latitude/longitude, while we still use linear interpolation of the wind data to the exact particle position when computing their trajectories. However, the neural network could in principle also determine the wind components almost exactly at the particle positions upon repeatedly using the trained SISR model to increase the resolution high enough to obtain the wind values at the respective particle positions. FLEXPART also needs other data than the wind data, for which we use linear interpolation of the ERA5 data. For temporal and vertical interpolation, we also used linear interpolation, as is standard in FLEXPART.

We started multiple simulations with 10 million trajectories on a global regular grid with 138 vertical levels and traced the particles for 48 hours. This simulation was repeated in each season for the following cases:

- the original ERA5 data at $0.5° \times 0.5°$ resolution, serving as the ground truth reference case;
- a data set, for which the winds were interpolated from degraded $1° \times 1°$ and from $2° \times 2°$ resolution data, using linear interpolation as is standard in FLEXPART;
- a data set, for which the winds were interpolated from degraded $1° \times 1°$ (using the neural network `model2` trained to interpolate $2° \times 2°$ to $1° \times 1°$) and from $2° \times 2°$ resolution data (using the neural network `model4` trained to interpolate $4° \times 4°$ to $2° \times 2°$), and then interpolated to the particle position using linear interpolation in FLEXPART.

2.6 Trajectory error metrics

As in previous studies (Kuo et al., 1985; Stohl et al., 1995), we compared the trajectory positions for trajectories calculated with the interpolated data to those calculated with the reference data set, using the Absolute Horizontal Transport Deviation (AHTD), defined as:

$$\mathrm{AHTD(t)} = \frac{1}{N}\sum_{n=1}^{N} D[(X_n, Y_n, t), (x_n, y_n, t)] \tag{3}$$

where N is the total number of trajectories, $D[(X_n, Y_n, t), (x_n, y_n, t)]$ is the great circle distance of trajectory points with longitude/latitude coordinates (X_n, Y_n, t) for the reference trajectories and (x_n, y_n, t) for the trajectories using interpolated winds at time t, for trajectory pair n starting at the same point. AHTD values are evaluated hourly along the trajectories, up to 48 hours.

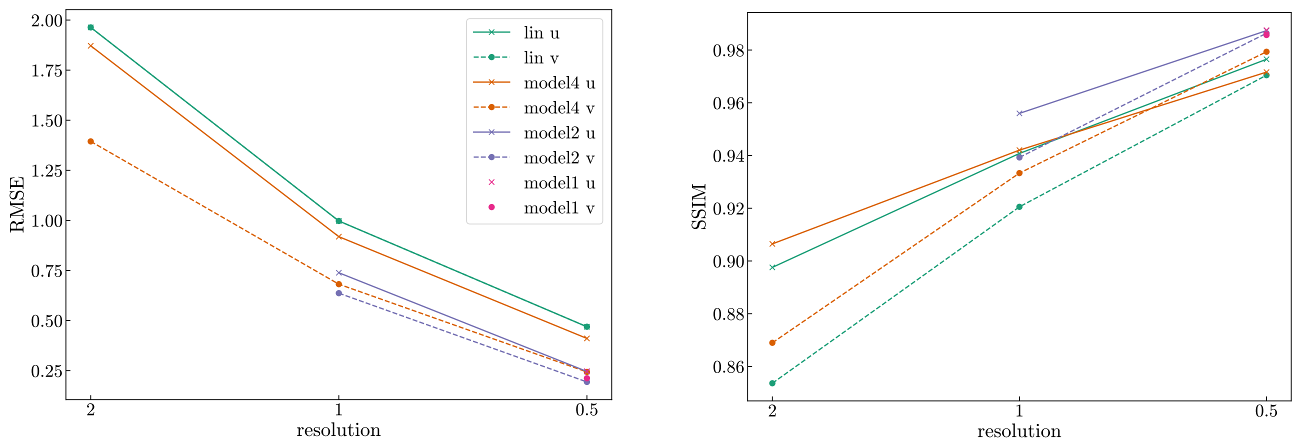
3 Results

In this section, we first compare the neural network interpolated data to the linear interpolated data. Then, we compare the accuracy of the trajectories computed with FLEXPART using the interpolated fields of the neural network to the linear interpolated fields.

3.1 Interpolation

We investigate the self-similarity of the spatial scales by interpolating the fields multiple times using the same model trained to up-scale the wind fields from lower resolutions. For the interpolation we use linear and neural network interpolation. First, we compare the fields which are interpolated from $1° \times 1°$ to $0.5° \times 0.5°$ resolution data. Then, we proceed with applying the neural network interpolation multiple times to generate arbitrary resolution.

In Fig. 2 we show the interpolation results for three different neural networks and for the linear interpolation. Here, each neural network is used to interpolate each resolution, starting with the resolution the model is trained on. The interpolation error reduces with the resolution. We see that each neural network almost always has better metrics (i.e., lower RMSE and higher SSIM values) than the corresponding linear interpolation. This is true both for the resolution the neural network has been trained for, as well as for higher resolutions.



(a) RMSE for different interpolation methods.

(b) SSIM for different interpolation methods.

Figure 2. RMSE (left) and mean SSIM (right) of the validation data set (14th January 2000) for linear and neural network interpolation evaluated at different resolutions. Here, we do not interpolate the fields multiple times (as explained in section 3.1.2) but rather once for each resolution starting from the resolution the model is trained on. The solid lines are computed for the u velocity and the dashed lines for the v velocity.

3.1.1 One time up-scaling

We first consider the neural network `model2` (trained to interpolate $2° \times 2°$ to $1° \times 1°$ resolution data). For the evaluation we interpolate a field from degraded $1° \times 1°$ resolution data to $0.5° \times 0.5°$ resolution data. Notice that we have trained the model separately for levels 0–50 and 51–137 (counting from the ground), and evaluate the correspondingly trained model.

In Fig. 3 we show the RMSE for the interpolated field for level 10 (corresponding approximately to an altitude of 245 m above ground level) and for one particular time (14th of January at 10:00 UTC) as an example. Fig. 3 also shows a blown-up region focusing on a cold front south of Australia, where we clearly see that the largest RMSE errors for both linear and neural network interpolation occur along the cold front. The winds in the boundary layer show a large wind shear between the southwesterly winds to the southeast of the cold front and the northwesterly winds to the northeast of the cold front. This causes large interpolation errors in the frontal zone. However, these large errors are substantially reduced by the neural network interpolation compared to the linear interpolation. This also means that especially the largest interpolation errors are avoided by the neural network, compared to the linear interpolation (see Fig. 4). This finding also holds for other conditions, as can be seen by the general error reduction in other regions on the globe (Fig. 3) and the smaller but still significant error reductions in cases with generally smaller interpolation errors (Fig. 4).

More example figures for the different months (January, April, July and October) can be found in the (zenodo) repository. We note that the results are similar for the different months, as shown in Table 1, where we computed the RMSE and SSIM for

RMSE ↓	Linear		Neural Network		SSIM ↑	Linear		Neural Network	
	u	v	u	v		u	v	u	v
January	0.469	0.398	0.214	0.181	January	0.976	0.970	0.988	0.987
April	0.393	0.36	0.206	0.182	April	0.976	0.968	0.988	0.986
July	0.447	0.444	0.222	0.193	July	0.975	0.968	0.988	0.986
October	0.589	0.532	0.216	0.191	October	0.975	0.969	0.988	0.987

Table 1. RMSE and mean SSIM of the validation data set for linear and neural network interpolation using `model2`. Considering the RMSE, the neural network interpolation is at least 49% more accurate compared to the linear interpolation.

a whole day in the months January, April, July and October and all levels. The neural network interpolation has less than half the RMSE of the linear interpolation and achieves a higher SSIM value.

We note that the interpolation time on our hardware for one field for a given level and time for the linear interpolation is about 10 times faster compared to the same interpolation using the neural network.

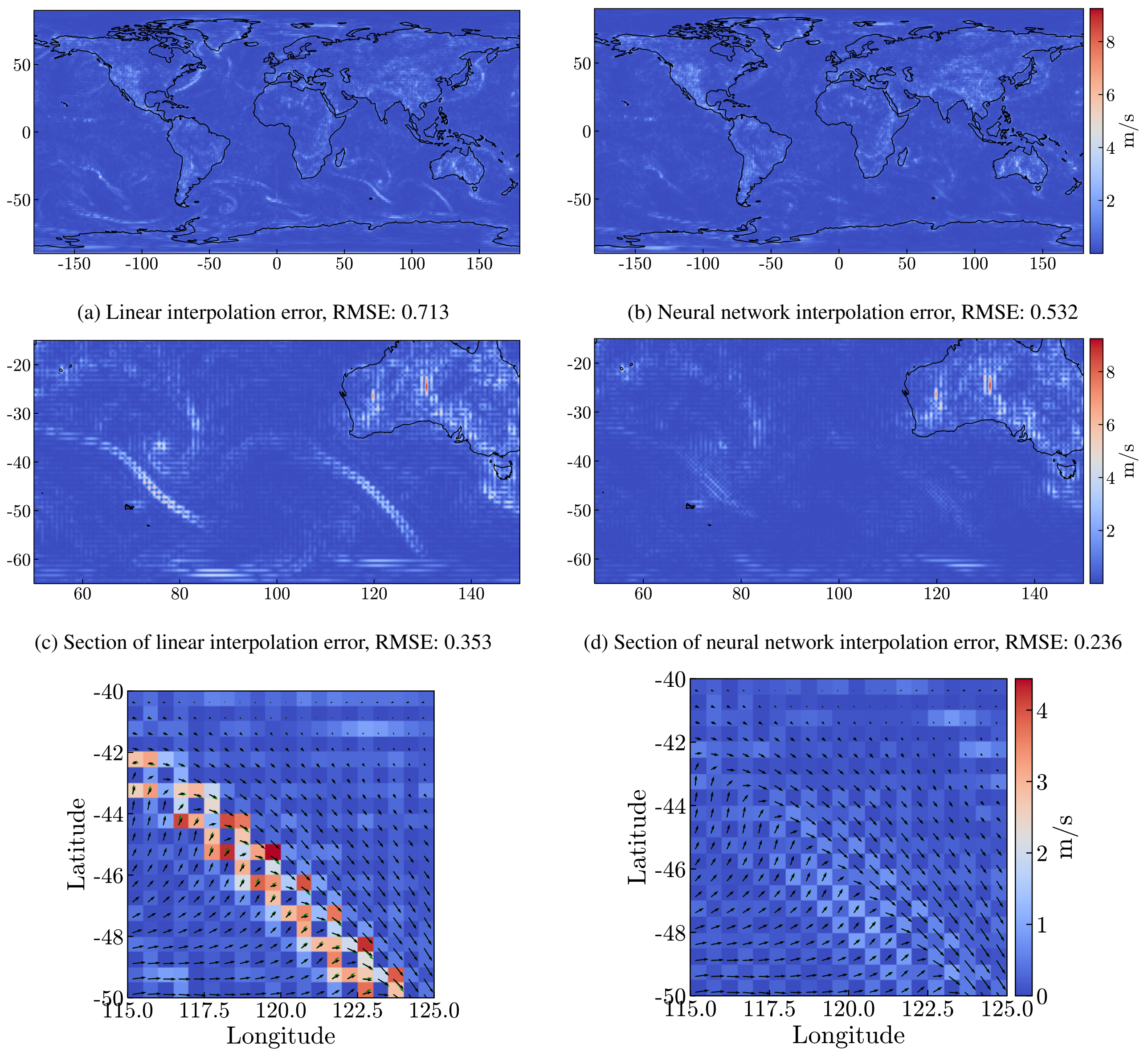
3.1.2 Multiple time up-scaling

To demonstrate that the neural network can be used to interpolate a field to arbitrary resolution, we consider the neural network `model4` (trained to interpolate $4° \times 4°$ to $2° \times 2°$ resolution data). We apply the network to interpolate $2° \times 2°$ to $1° \times 1°$ and apply it another time to interpolate $1° \times 1°$ to $0.5° \times 0.5°$ resolution data. In Fig. 5 we evaluate the RMSE of an interpolated field in January at level 10 (around 245m) and compare it to the RMSE of the linear interpolation.

For each method the RMSE is higher compared to the one time up-scaling, since we start with a lower resolution and less information. Nevertheless, the RMSE of the neural network interpolation is again lower compared to the linear interpolation, albeit the relative error reduction is smaller than with one-time up-scaling (see Table 2). This also holds for other samples in different seasons which we omitted showing here. When evaluating the RMSE for a day in January, April, July and October for all levels (Table 2), we observe that the neural network interpolation is 19% more accurate than the linear interpolation. Here, we are limited to the resolution of the reference data. Thus, we can only demonstrate the interpolation for two times. Coarser data does not have enough small scales represented, and it is therefore not meaningful to train the network on coarser data and upscale the data more times.

3.2 Trajectory accuracy

In the previous sections, we showed that our neural network interpolated wind velocity fields are more similar to the original $0.5° \times 0.5°$ resolution data than their linearly interpolated equivalents. However, this does not necessarily mean that trajectories advanced using the neural network interpolated fields are more accurate. Trajectories are not always equally sensitive to wind



(a) Linear interpolation error, RMSE: 0.713

(b) Neural network interpolation error, RMSE: 0.532

(c) Section of linear interpolation error, RMSE: 0.353

(d) Section of neural network interpolation error, RMSE: 0.236

(e) Section of linear interpolation error with interpolated vector field (black) and true vector field (green).

(f) Section of neural network interpolation error with interpolated vector field (black) and true vector field (green).

Figure 3. Differences between the $\mathbf{v} = (u, v)$ fields on the 14th of January 2000 at 10:00 UTC for level 10 (around 245m) and the truth ERA5 data, for linear interpolation (left panels) and for the interpolation by the `model2` neural network (right panels). The data are interpolated from degraded $1^\circ \times 1^\circ$ resolution data. The top row shows the error on the globe and the panels below show a blown-up section of the top panels.

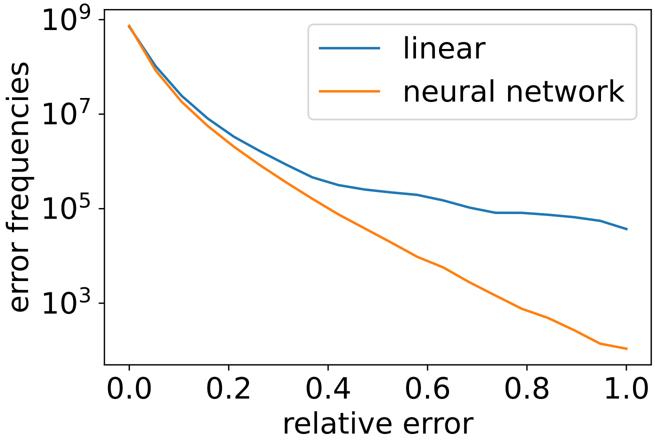


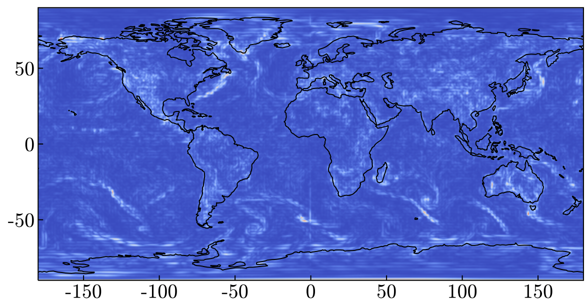
Figure 4. Comparison of the error frequencies for linear and neural network interpolation (`model2`) for 14th January 2000 (over all 137 vertical levels and over 24 hours). Here, we normalized the error and split the error frequencies into 20 bins of different intensities.

RMSE ↓	Linear		Neural Network		SSIM ↑	Linear		Neural Network	
	u	v	u	v		u	v	u	v
January	1.107	0.938	0.787	0.708	January	0.864	0.824	0.892	0.849
April	0.96	0.863	0.777	0.677	April	0.860	0.808	0.882	0.837
July	1.095	1.031	0.84	0.77	July	0.856	0.809	0.881	0.835
October	1.289	1.155	0.796	0.744	October	0.862	0.820	0.890	0.845

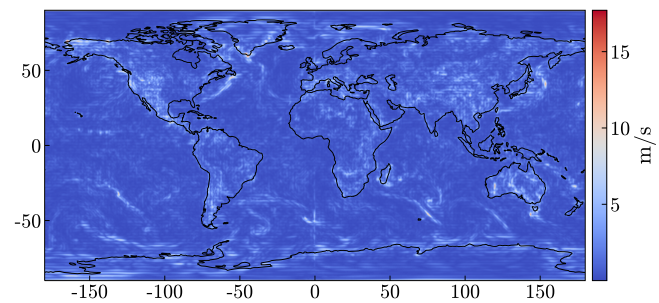
Table 2. RMSE and mean SSIM of the validation data set for linear and neural network interpolation, using `model4`. Considering the RMSE, the neural network interpolation is at least 19% more accurate compared to the linear interpolation.

interpolation errors, and it is therefore important to show that the individual trajectories that are advanced using neural network interpolated wind fields are indeed more similar to trajectories that are advanced using the original "ground-truth" wind fields.

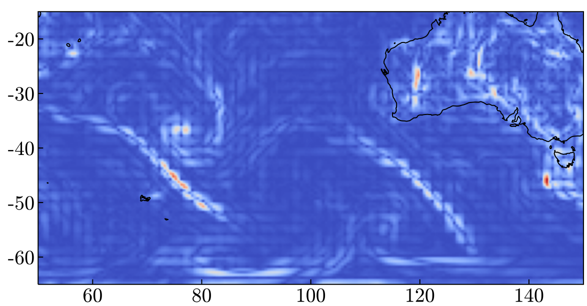
Fig. 6 shows that trajectories that are advanced using neural network interpolated wind fields are closer to trajectories that are advanced using the original "ground-truth" wind fields compared to trajectories using linear interpolated wind fields. Here, we show the results of the horizontal transport deviation (Eq. (3)) and standard deviations of particles advanced for 48 hours, using FLEXPART, after being initially globally distributed. Both the average horizontal transport deviation from the original *ground truth* trajectories as well as its standard deviation are smaller for the neural network as compared to the linear interpolation. The absolute deviations after 48 hours are on average $\sim 53.5\%$ ($1° \times 1°$ resolution) and $\sim 29.4\%$ ($2° \times 2°$ resolution) smaller for all seasons when using the neural network. Moreover, the standard deviation of the neural network is consistently smaller, no matter the season (on average $\sim 36.1\%, \sim 17.9\%$ smaller for the $1° \times 1°$ and $2° \times 2°$ resolution, respectively).



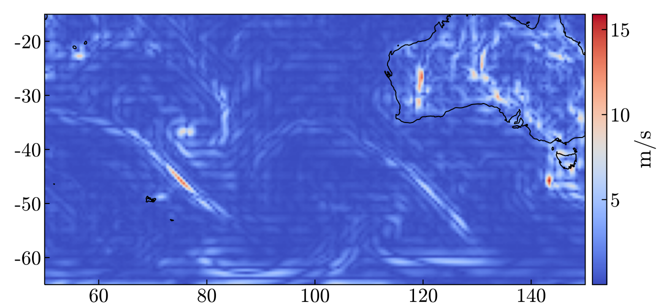
(a) Linear interpolation error, RMSE: 1.632



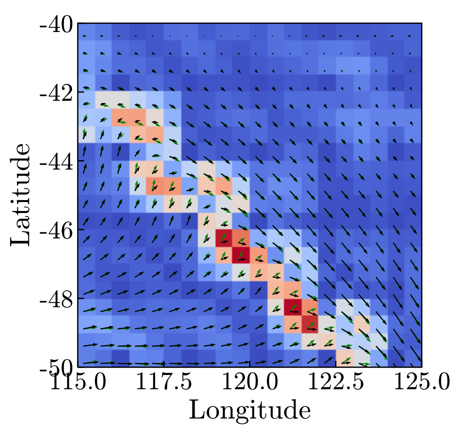
(b) Neural network interpolation error, RMSE: 1.443



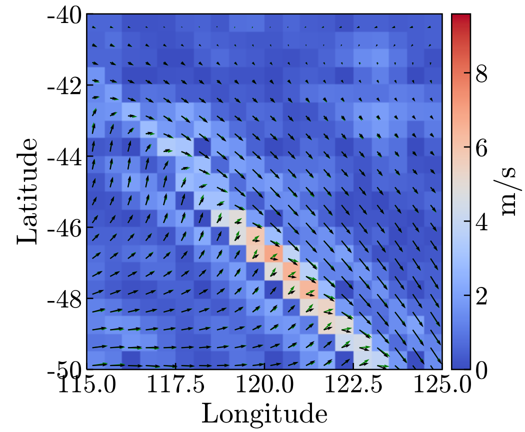
(c) Section of linear interpolation error, RMSE: 1.028



(d) Section of neural network interpolation error, RMSE: 0.904



(e) Section of linear interpolation error with interpolated vector field (black) and true vector field (green).



(f) Section of neural network interpolation error with interpolated vector field (black) and true vector field (green).

Figure 5. RMSE of $\mathbf{v} = (u, v)$ field at level 10 when compared to the truth ERA5 data. The date of the field is the 14th of January 2000 at 10:00 UTC. Linear interpolation (left panels) and the interpolation by the neural network `model4` (right panels). The data is interpolated from degraded $2^\circ \times 2^\circ$ resolution data. The top row shows the error on the globe and the panels below show a blown-up section of the top panels

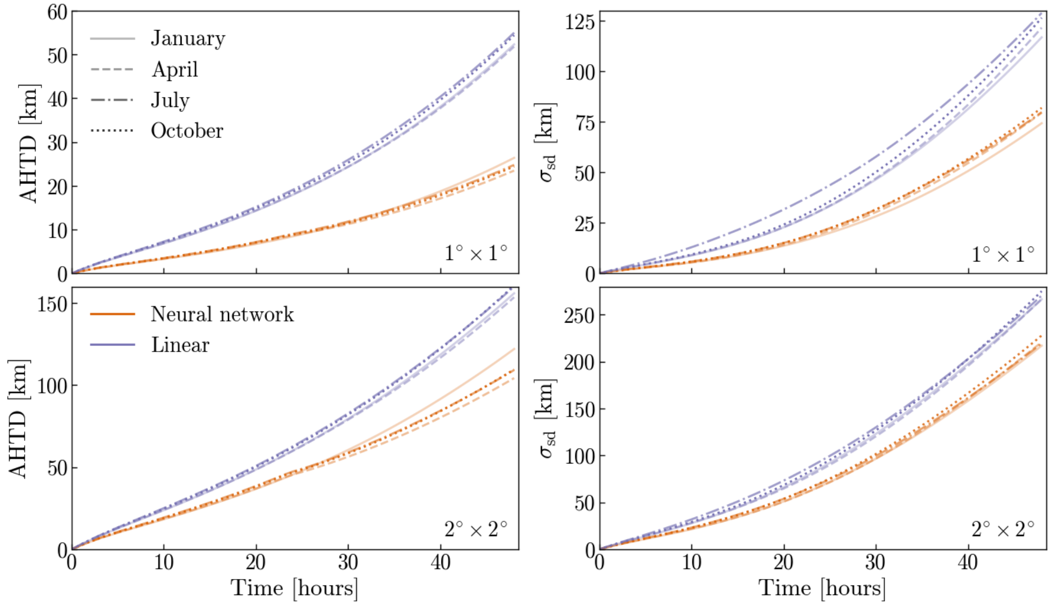


Figure 6. Absolute horizontal transport deviations (Eq. (3)) and standard deviations of 10 million particles advanced with FLEXPART, using two different degraded resolution data (as described in section 3.1) and two interpolation methods, as compared to the same particles advanced using the original full resolution data. The top panels show the results for the degraded $1^\circ \times 1^\circ$ resolution data (neural network interpolation using `model2`), and the bottom panels those of the degraded $2^\circ \times 2^\circ$ resolution data (neural network interpolation using `model4`). Orange lines show the AHTD (left panels) and standard deviation (right panels) of particles advanced using the neural network, and purple lines show these for the linearly interpolated data. Results for different seasons are shown with different line styles.

The improvement of the neural network over the linear interpolation is smaller with multiple time up-scaling from $2^\circ \times 2^\circ$ resolution than with one time up-scaling from $1^\circ \times 1^\circ$ resolution, since the latter has smaller wind interpolation errors (see Fig. 2). The reduced standard deviation we see in the neural network interpolated trajectories as compared to the linear interpolated ones is likely a result of the lower frequency of extreme deviations found in the neural network interpolated wind fields as compared to the linear interpolated ones (see Fig. 4). Thus, trajectories using the neural network interpolation are not only more accurate on average than trajectories using linear interpolation, but large trajectory errors are avoided more efficiently as well. This is important for avoiding misinterpretation when trajectories are used to interpret source-receptor relationships, e.g. for air pollutants or greenhouse gases.

We have also checked how well the quasi-conserved meteorological property of potential vorticity is conserved along the trajectories by computing absolute and relative transport conservation errors along trajectories in the stratosphere. We excluded particles affected by convection or boundary layer turbulence by selecting trajectories within the stratosphere that never traveled through space where the relative humidity exceeded 90%. A full explanation of the method we used can be found in Stohl and

Seibert (1998). The absolute and relative transport conservation errors of potential vorticity showed insignificant differences between the different trajectory data sets.

Note that we have not changed vertical and time interpolation of the winds and that we have not changed the interpolation of the vertical wind. Furthermore, we have not made full use of the neural network horizontal interpolation of the horizontal winds, as interpolation below $0.5^\circ \times 0.5^\circ$ resolution was still done using linear interpolation. We therefore consider substantial further error reductions possible, if neural network interpolation both in space and time is fully implemented directly in the trajectory model. This way neural network interpolation could make semi-Lagrangian advection schemes much more accurate.

4 Conclusions

In this paper we have considered the problem of increasing the spatial resolution of meteorological fields using techniques of machine learning, namely using methods originally proposed for the problem of single image superresolution. Higher-resolution meteorological fields are relevant for a variety of meteorological and engineering applications, such as particle dispersion modeling, semi-Lagrangian advection schemes, down-scaling, and weather nowcasting.

What sets the present work apart from a pure computer vision problem is that meteorological fields are characterized by self-similarity over a variety of spatio-temporal scales. This gives rise to the possibility of training a neural network to learn to increase the resolution from a down-sampled meteorological field to the original native resolution of that field, and then to repeatedly apply the same model to further increase the resolution of that field beyond the native resolution. We have shown in this paper that this is indeed possible. Wind interpolation errors are at least 49% and 19% smaller than errors using linear interpolation, with one time up-scaling, and with multiple time up-scaling, respectively. Here, we note that the multiple time up-scaling has a lower improvement than the one time up-scaling because we use different neural networks based on the available resolution. This means that the neural network trained on lower resolution data has less information and is less accurate than the neural network trained on the higher resolution data. We have also shown that corresponding absolute horizontal transport deviations for trajectories calculated using these wind fields are 52% (from degraded $1^\circ \times 1^\circ$ resolution data) and 24% (from degraded $2^\circ \times 2^\circ$ resolution data) smaller than with winds based on linear interpolation. This is a substantial reduction, given that we have not changed vertical and time interpolation and that we have not at all changed the interpolation of the vertical wind. Furthermore, we have not even made full use of the neural network interpolation, as interpolation below $0.5^\circ \times 0.5^\circ$ resolution was still done using linear interpolation.

While in the present work we have exclusively focused on the spatial interpolation improvement problem, similar techniques as presented here are applicable to the temporal interpolation case as well. Here, the problem can be interpreted as increasing the frame rate in a given video clip, with the native resolution given by the temporal resolution as made available by numerical weather prediction centres. We are presently working on this problem as well, and the results will be presented in future work. Subsequently, spatial and temporal resolution improvements can be combined to provide a seamless way to increase the overall resolution of meteorological fields for a variety of spatio-temporal interpolation problems.

Lastly, we should like to stress that meteorological fields are quite different from conventional photographic images as typically considered for superresolution tasks. Namely, meteorological fields follow largely a well-defined system of partial differential equations, which we have not considered when increasing the spatial resolution of the given data sets. This means that potentially important meteorological constraints such as energy, mass and potential vorticity conservation may be violated by obtained up-scaled data sets, as is also the case for other interpolation methods. Incorporating these meteorological constraints would be critical if these fields would be used in conjunction with numerical solvers, and correspondingly the proposed methodology would have to be modified to account for these constraints. This will constitute an important area of future research, with a potential avenue being provided through so-called physics-informed neural networks. See e.g. the studies by Raissi et al. (2019) and Bihlo and Popovych (2022) for an application of this methodology to solving the shallow-water equations on the sphere. Physics-informed neural networks allow one to take into account both data and the differential equations underlying these data, which would enable one to train a neural network based interpolation method that is also consistent with the governing equations of hydro-thermodynamics. Including consistency with these differential equations will be another potential avenue of research in the near future.

Code and data availability. The code to train the neural network and data to reproduce the plots is available at: https://zenodo.org/record/7350568. ERA5 re-analysis data (Hersbach et al., 2020) was downloaded from the Copernicus Climate Change Service (C3S) Climate Data Store. The results contain modified Copernicus Climate Change Service information. Neither the European Commission nor ECMWF is responsible for any use that may be made of the Copernicus information or data it contains. We used flex_extract to download the ERA5 re-analysis data, see (Tipka et al., 2020). The documentation can be found here https://www.flexpart.eu/flex_extract/ecmwf_data.html.

Author contributions. All authors contributed equally to the conceptualization, methodology and writing of the manuscript. RB and LB carried out the numerical simulations and code development.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. This research was undertaken in part thanks to funding from the Canada Research Chairs program, the InnovateNL LeverageR&D program, the NSERC Discovery Grant program and the NSERC RTI Grant program. The study was also supported by the Dr. Gottfried and Dr. Vera Weiss Science Foundation and the Austrian Science Fund in the framework of the project P 34170-N, "A demonstration of a Lagrangian re-analysis (LARA)". Moreover, this project is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 274762653 – TRR 181.

References

Bihlo, A.: A generative adversarial network approach to (ensemble) weather prediction, Neural Netw., 139, 1–16, arXiv:2006.07718, 2021.

Bihlo, A. and Popovych, R. O.: Physics-informed neural networks for the shallow-water equations on the sphere, J. Comput. Phys., 456, 111 024, 2022.

Brecht, R. and Bihlo, A.: Computing the ensemble spread from deterministic weather predictions using conditional generative adversarial networks, arXiv:2205.09182, 2022.

Chen, H., He, X., Qing, L., Wu, Y., Ren, C., Sheriff, R. E., and Zhu, C.: Real-world single image super-resolution: A brief review, Inf. Fusion, 79, 124–145, 2022.

Choi, M., Kim, H., Han, B., Xu, N., and Lee, K. M.: Channel attention is all you need for video frame interpolation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 10 663–10 671, 2020.

Dong, C., Loy, C. C., He, K., and Tang, X.: Learning a deep convolutional network for image super-resolution, in: European conference on computer vision, pp. 184–199, Springer, 2014.

Durran, D. R.: Numerical methods for fluid dynamics: With applications to geophysics, vol. 32, Springer, New York, 2010.

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., and Yacalis, G.: Could machine learning break the convection parameterization deadlock?, Geophys. Res. Lett., 45, 5742–5751, 2018.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., et al.: The ERA5 global reanalysis, Q. J. R. Meteorol. Soc., 146, 1999–2049, 2020.

Hoffmann, L., Baumeister, P. F., Cai, Z., Clemens, J., Griessbach, S., Günther, G., Heng, Y., Liu, M., Haghighi Mood, K., Stein, O., et al.: Massive-Parallel Trajectory Calculations version 2.2 (MPTRAC-2.2): Lagrangian transport simulations on Graphics Processing Units (GPUs), Geoscientific Model Development Discussions, pp. 1–51, 2021.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A.: Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1125–1134, 2017.

Kim, J., Lee, J. K., and Lee, K. M.: Accurate image super-resolution using very deep convolutional networks, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 1646–1654, 2016.

Krizhevsky, A., Sutskever, I., and Hinton, G. E.: Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105, Curran Associates, 2012.

Kuo, Y.-H., Skumanich, M., Haagenson, P. L., and Chang, J. S.: The accuracy of trajectory models as revealed by the observing system simulation experiments, Monthly Weather Review, 113, 1852–1867, 1985.

LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning, nature, 521, 436–444, 2015.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 4681–4690, 2017.

Li, X. and Orchard, M. T.: New edge-directed interpolation, EEE Trans. Image Process., 10, 1521–1527, 2001.

Lim, B., Son, S., Kim, H., Nah, S., and Mu Lee, K.: Enhanced deep residual networks for single image super-resolution, in: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 136–144, 2017.

Mouatadid, S., Easterbrook, S., and Erler, A. R.: A machine learning approach to non-uniform spatial downscaling of climate variables, in: 2017 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 332–341, IEEE, 2017.

Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K.: Wavenet: A generative model for raw audio, arXiv preprint arXiv:1609.03499, 2016.

Pisso, I., Sollum, E., Grythe, H., Kristiansen, N. I., Cassiani, M., Eckhardt, S., Arnold, D., Morton, D., Thompson, R. L., Groot Zwaaftink, C. D., et al.: The Lagrangian particle dispersion model FLEXPART version 10.4, Geoscientific Model Development, 12, 4955–4997, 2019.

Raissi, M., Perdikaris, P., and Karniadakis, G. E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys., 378, 686–707, 2019.

Rasp, S., Dueben, P. D., Scher, S., Weyn, J. A., Mouatadid, S., and Thuerey, N.: WeatherBench: a benchmark data set for data-driven weather forecasting, J. Adv. Model. Earth Syst., 12, e2020MS002 203, 2020.

Scher, S. and Messori, G.: Predicting weather forecast uncertainty with machine learning, Q. J. R. Meteorol. Soc., 144, 2830–2841, 2018.

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1874–1883, 2016.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-k., and Woo, W.-c.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting, in: Advances in Neural Information Processing Systems, vol. 28, pp. 802–810, Curran Associates, arXiv:1506.04214, 2015.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al.: Mastering the game of Go without human knowledge, Nature, 550, 354–359, 2017.

Sprenger, M. and Wernli, H.: The LAGRANTO Lagrangian analysis tool–version 2.0, Geoscientific Model Development, 8, 2569–2586, 2015.

Stohl, A. and Seibert, P.: Accuracy of trajectories as determined from the conservation of meteorological tracers, Quarterly Journal of the Royal Meteorological Society, 124, 1465–1484, 1998.

Stohl, A., Wotawa, G., Seibert, P., and Kromp-Kolb, H.: Interpolation errors in wind fields as a function of spatial and temporal resolution and their impact on different types of kinematic trajectories, Journal of Applied Meteorology and Climatology, 34, 2149–2165, 1995.

Stohl, A., Forster, C., Frank, A., Seibert, P., and Wotawa, G.: The Lagrangian particle dispersion model FLEXPART version 6.2, Atmos. Chem. Phys., 5, 2461–2474, 2005.

Tipka, A., Haimberger, L., and Seibert, P.: Flex_extract v7. 1.2–a software package to retrieve and prepare ECMWF data for use in FLEXPART, Geoscientific Model Development, 13, 5277–5310, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I.: Attention is all you need, Advances in neural information processing systems, 30, 2017.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P.: Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process., 13, 600–612, 2004.

Weyn, J. A., Durran, D. R., and Caruana, R.: Can Machines Learn to Predict Weather? Using Deep Learning to Predict Gridded 500-hPa Geopotential Height From Historical Weather Data, J. Adv. Model. Earth Syst., 11, 2680–2693, 2019.

Yang, W., Zhang, X., Tian, Y., Wang, W., Xue, J.-H., and Liao, Q.: Deep learning for single image super-resolution: A brief review, EEE Trans. Multimed., 21, 3106–3121, 2019.