Adrien Burq
Advisor : Pr. Gentine

March - August 2022

SUB-GRID PARAMETRIZATION FOR THE CONVECTIVE BOUNDARY LAYER

MAP594 : Probabilistic and statistical modelisation - Machine Learning

COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

CONTENTS

Github repository here

INTEGRITY STATEMENT ON PLAGIARISM

I hereby certify that :

- I am the author of this work which was conducted in Pr. Gentine's lab in the Earth and Environmental Department at Columbia University in the City of New York.
- I have attributed and cited any borrowed content.

August 2022,
Adrien Burq.

ACKNOWLEDGMENTS

First of all, I would like to thank everyone who has made this internship possible and in particular my advisor Pr. Pierre Gentine without whom I would not have been able to do this internship. It was four amazing months and I am grateful for it. I would also like to thank Dr. Sara Shamekh and Dr. Alex Connolly with whom I shared common research interests and who also guided me throughout this experience. Finally, I would like to thank everyone in Pierre's lab for their contribution and advice, as well as the Earth and Environmental department at Columbia University who made the internship possible.

IMPROVING CLIMATE MODELS

Today's weather and climate models are increasingly accurate. Current weather models are now able to predict with great accuracy the weather for at least one week in a particular location. Thanks to these models, there is now a consensus among the scientific community on the general trend of climate change, thanks to these climate models. Climate models have enabled researchers to publish comprehensive reports on the current state of the climate emergency such as the latest 2021 and 2022 IPCC reports. These reports also advise policy makers on how to take action to mitigate the phenomenon.

To devise such reports and act at an international level, one could think that we only need global models which include the entire world and there is no need to go into the details of individual regions. Therefore, why should we still need to improve our climate models when we already know the challenges we are facing are global challenges that need to be addressed at an international level ? Why should we strive to improve climate models when we already have all the data necessary to devise the right course of action at an international level in order to tackle the climate emergency ? One could even think it is contradictory to develop new models since simulating the Earth atmosphere, land and ocean is resource intensive.

Admittedly there is a consensus around the global goal, which is reducing greenhouse gases emissions in order to keep global warming under a certain threshold, but how does each country and each region inside a country devise a plan to achieve this goal ? We need precise models which can capture the specifics of an individual place in order to devise tailor-made measures taking into account the risks and opportunities of each region around the world.

In the case of the atmosphere, the dynamics of the climate system are governed by seven physical principles : conservation of air mass conservation, conservation of water mass, conservation of energy, conservation of air momentum in three directions and the ideal gas law applied to air. These seven equations constrain seven variables : air temperature, pressure, density, water vapor content, wind magnitude in three directions. By solving the equations, we can simulate all seven variables in time and space. However, such simulations require to use meshes with point that are barely centimeters from each other in order to capture the entire physics of the atmosphere. Such precision is impossible for global climate models since it would take centuries to run them with the current processing power available. Therefore, current Global Climate Models (GCM) are based on grids with points that are about 100km apart from each other. Regional Climate Models (RCM) use in general grids with a resolution of the order of 10km (Figure 1).

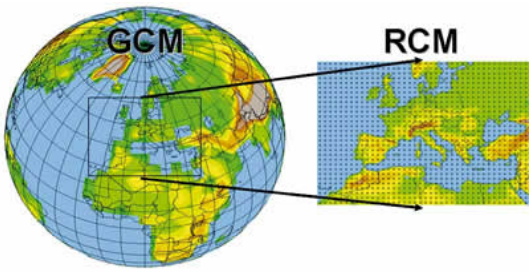
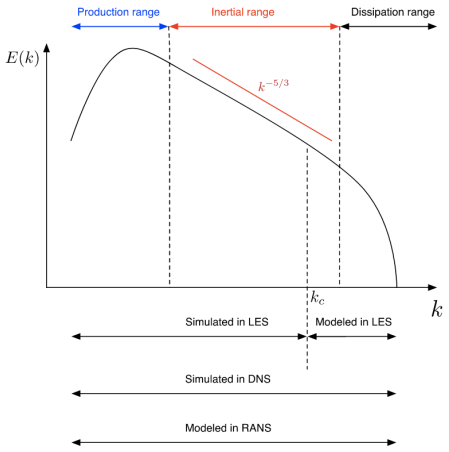
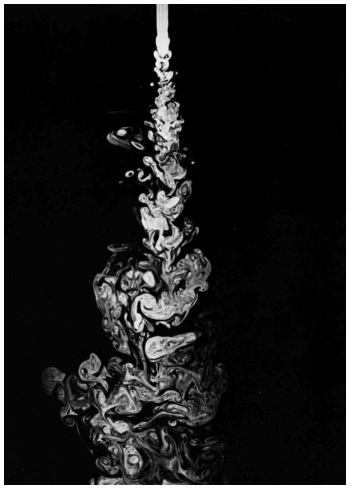


Figure 1: GCM and RCM comparison. GCM : $\sim$ 100km, RCM : $\sim$ 10km

These resolutions cannot capture the entire complexity of the atmosphere dynamics by a direct simulation. Indeed, atmospheric phenomena are multi-scale and we need to approximate most of it with parametrizations when using at coarse resolution simulations like GCMs or RCMs. In particular the energy cycle for turbulent flows appears at different scales (Figure 2a) and turbulent flows have a fractal nature (Figure 2b). Therefore, large scale simulations need parametrization techniques in order to take into account atmospheric sub-grid processes.



(a) The energy cascade model of Kolmogorov describes the process of turbulent energy transfer in 3D turbulence.



(b) Turbulent water jet with small eddies inside larger ones [1]

Figure 2: Multi-scale processes.

Due to the complexity of climate processes, models have to reduce the resolution of the mesh grid and the mathematical equations to their most "idealistic" representations of Earth's climate systems, ignoring many variables, such as clouds and turbulence. These equations can be further limited in that they may only work under particular conditions. Such variables could however be a key feature to understand climate change, which is why it is critical to build higher resolution models which can account for those processes and bridge the gap between local high resolution models and the low resolution global models through comprehensive and resource efficient parametrization models.

For such a task, machine learning techniques show promising results. Indeed, instead of using known physical laws, machine learning algorithms can reverse-engineer the process and learn the laws by themselves from a large amount of data. Combining physics and data-driven approaches can lead to less intensive parametrization models, without having to use simplified mathematical models. This would enable climate scientists to simulate climate systems at a higher resolution for the same amount of computing power. Data science for climate science can thus help achieving better and cheaper local and global climate models which are paramount to understanding and mitigating climate change.

In Pr. Gentine's team, Ph.D students, post-doctoral scholars and research scientists strive to advance knowledge in the field of data science for climate science by investigating two main subjects :

- Sub-gid scale parametrization for atmospheric layers in different conditions,
- Carbon capture models.

1
STATE OF THE ART

1.1 The boundary layer and classical climate models

1.1.1 • The Boundary Layer

Human's experience of the Earth's atmosphere is not representative of the whole atmosphere. Indeed, we spend most of our lives near the Earth's surface which influences greatly the lowest layers of air. These lowest layers form what we call the atmosphere's *boundary layer*. The reminder of the troposphere is called the *free atmosphere*. The reader can see the first chapter of [2] for a clear introduction to boundary layer meteorology. Simply put, the boundary layer can be defined as that part of the troposphere that is directly influenced by the presence of the earth's surface, and responds to surface forcings with a timescale of about an hour or less. Indirectly, the whole troposphere can change in response to surface characteristics, but this response is relatively slow outside of the boundary layer which is the reason for the timescale precision in the boundary layer's definition.

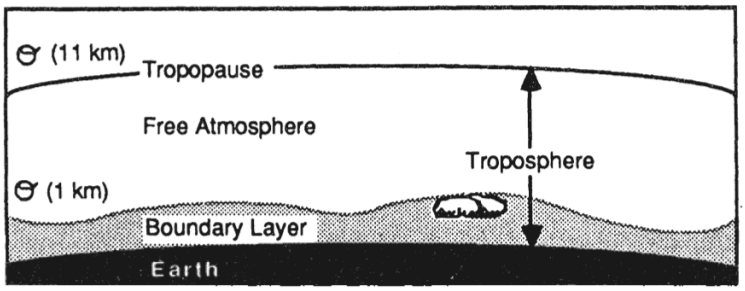


Figure 3: The troposphere can be divided into two parts: a boundary layer (shaded) near the surface and the free atmosphere above it [2].

Air flow, or wind. can be divided into three broad categories: mean wind, turbulence, and waves. Each can exist separately or interact with another in the boundary layer where transport of quantities such as heat or momentum is dominated in the horizontal by the mean wind, and in the vertical by turbulence. Therefore, a common approach for studying turbulence is to split variables such as temperature and wind into a mean part and a perturbation part (the so called Reynolds' decomposition).

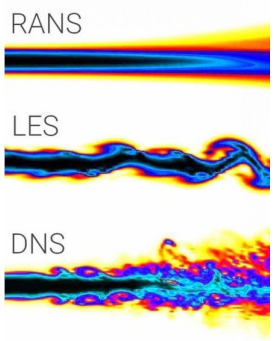
Turbulence in the atmosphere can be visualized like in figure 2b as irregular swirls of motion called eddies. These eddies consist of many different eddies of different sizes superimposed on each other. The different scale of turbulent eddies form the turbulence spectrum which spans on many orders of magnitude and justify the need of high resolution

simulations for the boundary layer. Turbulence is several orders of magnitude more effective at transporting quantities than is molecular diffusivity. It is turbulence that allows the boundary layer to respond to changing surface forcings.

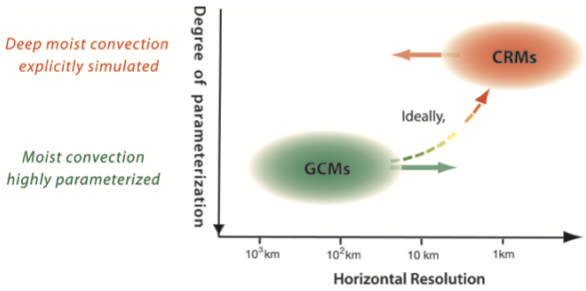
1.1.2 • Climate models

Traditional global models of the atmosphere use grid lengths of the order of 100 km and regional models only use grids as refined as 1 km which is the order of magnitude of the boundary layer. Therefore, turbulent eddies are usually filtered out from meteorological forecast models and turbulent transfer onto larger flows need to be parameterized. These simulations usually use Reynolds Averaged Navier-Stokes methods (RANS). Further details on the average used in RANS models can be found in section 2.2.2.

Large-Eddy Simulation (LES) can model turbulence and clouds on grids with a resolution as low as 10 m. At these scale, the simulations can resolve explicitly the most energetic turbulent motions but still are not as accurately as Direct Numerical Simulations (DNS) which explicitly solve the system's governing equations on meshes with a typical grid length of 10 cm (Figure 4a).



(a) Different resolution levels of resolution for the same flow.



(b) Extent of parametrization required for a simulation given a grid resolution.

Figure 4: Simulation methods comparison [3].

When increasing the resolution of simulations, starting from Global Circulation Models to Large-Eddy Simulations, turbulent motions go from being totally filtered, to being partially resolved and even almost fully resolve some phenomena such as clouds [4]. LES can resolve cloud formation and structure which is key to weather forecast and understanding extreme climate events. The point of many advances and research in climate science like in [3] was to bridge the gap between Global Circulation Models and Cloud Resolving Models (Figure 4b).

1.1.3 • The atmospheric boundary layer "gray zone"

Since the 2010s, weather centers like Météo-France or the UK Met Office use grid length of about 1 to 3 km. In the convective-allowing regime, deep convective structures (i.e. large

cumulus clouds) become partially resolved and no longer occupy small fractional areas of the grid [5]. Therefore, the use of conventional deep convective parametrizations at these resolutions becomes highly questionable and they are often switched off.

In the convective boundary layer *gray zone*, the turbulent kinetic energy (TKE) is only partially resolved, in contrast to the LES resolution regime where it is mostly resolved and in contrast to the mesoscale regime where it is fully parameterized. Thus problem is : how can we use parametrization to account for the turbulent motion which is not resolved, without taking twice into account the turbulent motion that is resolved ?

Such a problem requires a deeper dive into turbulent flows' structure. This paper [6] for example distinguishes coherent and incoherent parts of the flow. In the engineering and geoscience communities, *coherent structures* are used to describe distinct and dominant turbulent flow structure. In this study, *coherent structures* explain most of the vertical transport in the boundary layer. Through the classification of updrafts and downdrafts, the method developed can help track individual structures.

1.1.4 • Data science for climate science

Machine learning (ML) techniques are increasingly used to extract patterns and gain insight from increasingly large geospatial datasets. The recent advances in the fields of machine learning and artificial intelligence combined with the increased computational power offer new perspectives for the climate science field. This paper [7] reviews the current state of data-driven techniques used in climate science and offers a critical point of view towards the remaining challenges to adapt ML frameworks to physical systems.

Analytical task	Scientific task	Conventional approaches	Limitations of conventional approaches	Emergent or potential approaches
Classification and anomaly detection				
	Finding extreme weather patterns	Multivariate, threshold-based detection	Heuristic approach, ad hoc criteria used	Supervised and semi-supervised convolutional neural networks[41,42]
	Land-use and change detection	Pixel-by-pixel spectral classification	Shallow spatial context used, or none	Convolutional neural networks[43]
Regression				
	Predict fluxes from atmospheric conditions	Random forests, kernel methods, feedforward neural networks	Memory and lag effects not considered	Recurrent neural networks, long-short-term-memories (LSTMs)[89,99,100]
	Predict vegetation properties from atmospheric conditions	Semi-empirical algorithms (temperature sums, water deficits)	Prescriptive in terms of functional forms and dynamic assumptions	Recurrent neural networks[90], possibly with spatial context
	Predict river runoff in ungauged catchments	Process models or statistical models with hand-designed topographic features[91]	Consideration of spatial context limited to hand-designed features	Combination of convolutional neural network with recurrent networks
State prediction				
	Precipitation nowcasting	Physical modelling with data assimilation	Computational limits due to resolution, data used only to update states	Convolutional–LSTM nets short-range spatial context[92]
	Downscaling and bias-correcting forecasts	Dynamic modelling and statistical approaches	Computational limits, subjective feature selection	Convolutional nets[72], conditional generative adversarial networks (cGANs)[53,93,101]
	Seasonal forecasts	Physical modelling with initial conditions from data	Fully dependent on physical model, current skill relatively weak	Convolutional–LSTM nets with long-range spatial context
	Transport modelling	Physical modelling of transport	Fully dependent on physical model, computational limits	Hybrid physical–convolutional network models[68,94]

Figure 5: Conventional approaches and deep learning approaches to geoscientific tasks [7].

The next step in this process to apply data science schemes in climate science is to use hybrid methods which incorporate physical constraints. The idea is that such physics informed models can couple unique physical processes with the versatility of data-driven machine learning. For instance, incorporating symmetries [8] in neural networks to produce equivariant networks [9] can enable a model to better capture the physical properties of the object. These constraints can be relaxed and tuned to fit the more or less symmetrical nature of the physical object that is simulated. Further constraints can be applied on a model like in this paper [10] where a deep neural network is trained to learn a model for the Reynolds stress anisotropy tensor with an embedded Galilean invariance.

Finally, techniques data-driven methods can help reverse engineer physical laws in some cases. Indeed, by feeding a machine learning algorithm with a large set of similar data depicting a true fluid flow, the algorithm can learn the dependencies between the data and rebuild the physical laws that govern said flow. Techniques such as principal component analysis or layer-wise relevant propagation [11] can help understand these links between variables that machine learning algorithms can find. This would enable simulations to run without (or with few) complex mathematical models which are computationally intensive.

1.2 Parametrization schemes

Let us go back and take a dive into the equations that govern climate systems. In particular, let us take a look at the momentum equations, which will help us understand what are sub-grid scale processes and what are the quantities we need to parameterize. The momentum equations states that :

$$\frac{du_i}{dt} = \text{conservation of momentum}$$

When expanded, this becomes :

$$\frac{\partial u_i}{\partial t} + \frac{\partial u u_i}{\partial x} + \frac{\partial v u_i}{\partial x} + \frac{\partial w u_i}{\partial x} = \text{conservation of momentum}$$

Which can be rewritten with Einstein's notation as :

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = \text{conservation of momentum}$$

On our grid however we are solving the filtered equations. We denote the filtered variables by an overline. Therefore the filtered equation on a grid becomes :

$$\overline{\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j}} = \text{conservation of momentum}$$

And by using the linearity and homogeneity of the filter, we get :

$$\frac{\partial \overline{u_i}}{\partial t} + \frac{\partial \overline{u_i u_j}}{\partial x_j} = \text{conservation of momentum}$$

If we add $\frac{\partial \bar{u_i}\bar{u_j}}{\partial x_j}$ on each side, we get :

$$\frac{\partial \overline{u_i}}{\partial t} + \frac{\partial \bar{u_i}\bar{u_j}}{\partial x_j} = -\frac{\partial[\overline{u_i u_j} - \bar{u_i}\bar{u_j}]}{\partial x_j} + \text{conservation of momentum}$$

Which allow us to define the Reynolds stress tensor τ with :

$$\tau_{ij} = \overline{u_i u_j} - \bar{u_i}\bar{u_j}$$

What we can explicitly simulate are the $\bar{u_i}$ variables. However, we have no way of directly simulating $\overline{u_i u_j}$ since we do not have the u_i inside the grid. Therefore, τ_{ij} is a sub-grid scale process which cannot be directly simulated. We want to approximate τ_{ij} with values that we directly simulate. In other words, we want to define a function f of the resolved variables $(\overline{u_i}, \bar{\theta}, ...)$, such that :

$$\tau_{ij} = f(\text{resolved variables})$$

The question at stake is then : how do we find such a parametrization ? How do we define f ? We can do this reasoning for the other governing equations and come up with similar quantities that we want to parameterize.

1.2.1 • Some classical parametrization schemes

Two main classes of sub-grid scale (SGS) models exist : functional models and structural models. The functional ones are often simpler than structural ones, focusing on representing energy dissipation at a physically correct rate. They model turbulent effects as an eddy viscosity and the dissipation of energy resembles molecular diffusion. In such models, we have :

$$\tau_{ij} = -2\nu_t \overline{S_{ij}}$$

Where ν_t is the turbulent eddy viscosity and $\overline{S_{ij}} = \frac{1}{2}(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i})$ is the rate of strain tensor evaluated over the filtered grid. The dimension of ν_t is $\frac{m^2}{s}$ so it is often approximated by the product of a characteristic length scale and a characteristic velocity scale.

The Smagorinsky model, which is considered as the pioneer model for sub-grid scale parametrization for LES. It assumes that small scale energy production and dissipation are at equilibrium therefore it does not account for backscatter (reverse energy cascade) [12]. It models the eddy-viscosity ν_t as

$$\nu_t = C\Delta^2\sqrt{2\bar{S_{ij}}\bar{S_{ij}}} = C\Delta^2|S|$$

Where Δ is the grid size and C a constant.

The model has proven to be a good baseline for Reynolds numbers that are sufficiently high. Indeed, in this case it is plausible to assume that energy is only transfered from the large to the small scales. It shows limits however in regions of intense shear for example and is constrained by the need to tune the constant C.

Another parametrization that does not require tuning is the Bardina model, which is a scale-similarity model. The principle for this model is to say that the energy transfer from all unresolved scales to resolved scales is dominated by the transfer from the first, largest unresolved scale to the smallest resolved scale [13]. In our case, the smallest resolved scale are the filtered variables $\overline{u_i}$ at each point on the grid. The above-mentioned transfer can then be written by applying the filter a second time :

$$\tau_{ij} \sim \overline{\overline{u}_i \overline{u}_j} - \bar{\bar{u}}_i \bar{\bar{u}}_j$$

This parametrization is easier to compute but is not sufficiently dissipative. Some more sophisticated models (called mixed-models) which use simultaneously scale-similarity model (like the Bardina model) and eddy-viscosity models (such as the Smagorinsky model [12]).

1.2.2 • Deep learning for sub-grid scale parametrization

The parametrizations of sub-grid scale processes in climate models can induce substantial uncertainty in the predictions. Machine learning of new parametrization approaches can show great promises in terms of computation time but the approaches thus far have not been able to generalize well between different flow structures. The interpolation to different Reynolds numbers is not yet well grasped and the performances when generalizing one framework to different grid scales has not yet been thoroughly investigated [14]. Convective parametrization in particular remains particularly tricky since convective processes play a crucial role in vertical structure heat and moisture, which impact precipitation intensity and cloud cover which in turn impact circulations on larger scales. Therefore a small error in the convective process can build up to larger errors for global circulation models [15].

The starting point for data-driven parametrization models is to use high resolution simulations as the ground truth of the underlying physical processes [14]. These simulations can then be coarse grained onto lower resolution grids. We then train a model on the coarse-grained simulation to predict the sub-grid physical processes.To do so, we use the high resolution simulation to compute a loss function. For instance, this work [16] uses Direct Numerical Simulations to learn the sub-grid stress tensor τ in a Large-Eddy Simulation. In this study, using Deep Neural Networks enabled the authors to achieve better performances on their test set than predictions that used the Smagorinsky model or the Smagorinsky-Bardina mixed model.

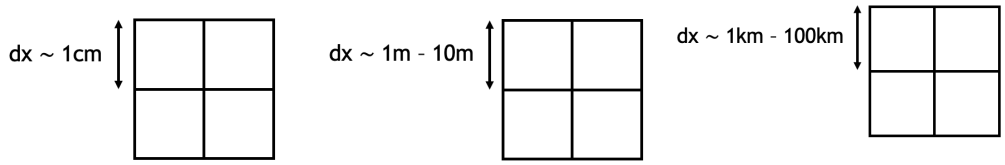
Turbulent flows are characterized by chaotic motions and intermittency, which are difficult to predict in addition to the multi-scales of the motions. Direct Numerical Simulation techniques are computationally expensive and data-driven approaches do not generalize well due to the lack of physical constraints embedded in the model. Physics informed model have been introduced to help bridge this gap. One method commonly used is to add physical constraints in the loss function to ensure that some boundary conditions are respected and that the output is physically plausible. Another technique is to incorporate physical principles into a deep neural network's architecture. [17] for example uses spatio-temporal filters to decompose the velocity field into multiple scales that respect the orders of magnitude of the Kolmogorov cascade.

2
SUB-GRID FLUXES PREDICTION ON A FIXED MESH

2.1 Project goal

Much of the boundary layer turbulence is generated by forcings from the ground [2]. In what follows, we study in particular the mixed layer which is a part of the boundary layer where turbulence is usually convectively driven. The convective boundary layer commonly occurs during daytime over continental land, and is characterized by a surface that is warm compared to the air immediately above, resulting in strong surface heat fluxes. These heat fluxes generate updrafts and downdrafts motions in the atmosphere called thermals.

2.1.1 • Simulation scales



(a) DNS scale (b) LES scale (c) RANS scale

Figure 6: Simulation grid scales comparison.

Our work here is based on LES simulations. The LES simulations will be used as the ground truth for training different Deep Learning models. Since LES simulations can resolve clouds and convective processes in general, the goal is to use these simulations as ground truth to parameterize lower resolution simulations such as Global Circulation Models that have a resolution of about 10km. There is a huge gap between the resolution of Large-Eddy Simulations and Reynolds Averaged Navier-Stokes based simulations. Therefore, the idea is to bridge this gap by devising a model that can generalize across the different resolution scales. Based on a ground truth of a LES, we aim at developing a model that can predict accurate sub-grid fluxes on different larger scales.

2.1.2 • Project outline

The first part of the project consisted in getting familiar with the high resolution simulations available, preprocessing the data and designing a predictive model for sub-grid fluxes over an arbitrary resolution scale. The second part of the project was adapting this model so that it could adapt to different resolutions and be accurate on grids with a resolution of 100m as well as on grids with a resolution of 1km.

2.2 Data and preprocessing

2.2.1 • High Resolution Simulation

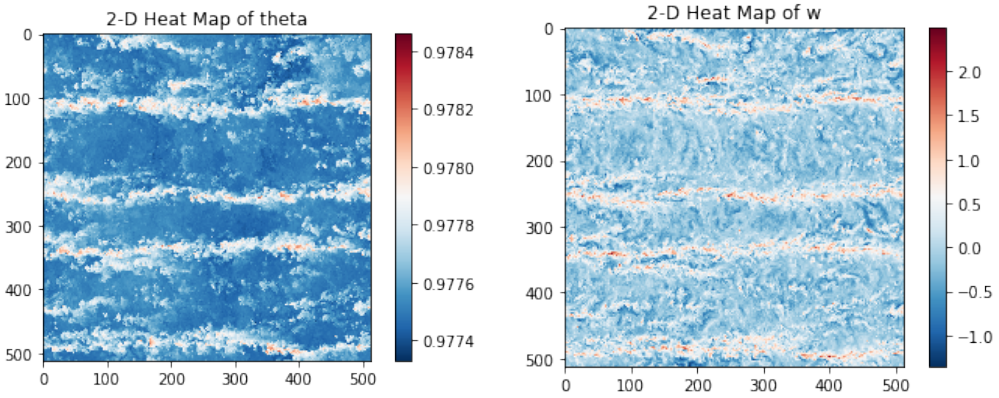


Figure 7: Visualization of w and θ in the horizontal plane at an altitude of 50m.

The main simulation we have been working with considers a box of size $6.3 * 6.3\text{km}^2$ horizontally and 1.5km vertically on a desert planet (there is no humidity). We have a mesh of 512^3 pixels and therefore a resolution of 12m horizontally and 3m vertically. The simulation is characterized by a velocity forcing of 16m.s^{-1} in the horizontal plane: if we define u, v and w as the velocities in the x, y and z axes respectively, we impose $u = 16\text{m.s} - 1$ for $z = 0$. The simulation uses a timestep of 0.25s to resolve the governing equations and outputs data every 10s.

The simulation computes for each timestep the values of u, v, w as well as the temperature θ and the tracer s which represents a variable which variations are monotonous along the vertical axis and has a long mixing time-scale. This means that it can record whether a flow particle is coming from above or below and therefore if we have an updraft or a downdraft in a particular area :

- We define : $s_{\text{average}}(z) = \text{average}(s(x, y, z))$
- Let's assume that s_{average} is increasing with z.
- If $s_{\text{particle}}(x, y, z) < s_{\text{average}}(z)$ then this particle must come from below and is part of an updraft at the location (x, y).
- On the contrary, if $s_{\text{particle}}(x, y, z) > s_{\text{average}}(z)$ then this particle must come from above and is part of an downdraft at the location (x, y).
- The reasoning is similar if s_{average} is decreasing with z.

2.2.2 • Reynolds Averaging and further preprocessing

On the LES simulation, the boundary layer corresponds only to the first 1100m. Therefore, We only keep the first the first 376 layers over the vertical axis.

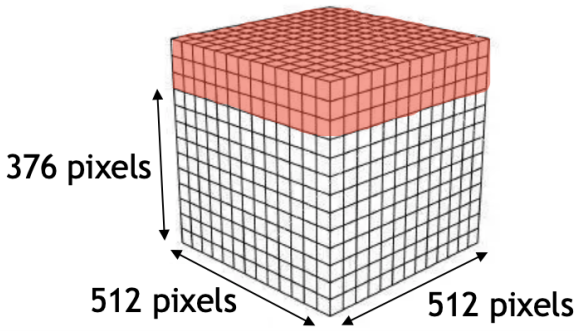


Figure 8: Final mesh of the high resolution baseline.

We now introduce Reynolds Averaging : we split the flow into the sum of a mean flow and a variation. Since the final objective is to make online predictions, we cannot use a time average for the mean flow but only a spacial average. We use an average over the whole domain of the simulation and use the notation [variable] for this average (which is different from the average inside a grid cell for sub-grid scale processes which we denote by an overline). For the temperature variable for example, we have :

$$\theta(x, y, z, t) = [\theta](t) + \theta'(x, y, z, t)$$

In the first part, we had defined the stress tensor τ with : $\tau_{ij} = \overline{u_i u_j} - \bar{u}_i \bar{u}_j$. By using the heat equation instead of the momentum equations, we can derive a similar quantity for heat transport : the heat flux at each point along the mesh, which is defined along every direction by :

$$\mathrm{hf}(x, y, z, t) = \overline{u_i' \theta'}$$

In particular, in our setting we are interested in the formation of vertical convective structures so we look more closely at the vertical heat flux : $\overline{w'\theta'}$.

- if $\overline{w'\theta'} > 0$:
  - either the wind is going up (ie : $w' > 0$) at a certain point where the temperature is higher than on average (ie : $\theta' > 0$) and heat is transported upward,
  - or the wind is going down (ie : $w' > 0$) at a certain point where the temperature is lower than on average (ie : $\theta' > 0$) and cool is transported downward, which means that heat is transported upward as well.
- similarly, if $\overline{w'\theta'} < 0$, heat is transported downward.

We can then use different coarse graining factors for each variable to train our models on different resolutions. From the high resolution simulation, we can coarse grain either vertically or horizontally. Indeed, in the convective boundary layer, the vertical motion is linked from top to bottom, whereas the horizontal motion is not as much linked across the horizontal plane. Therefore, defining a horizontal scale and defining a vertical scale are two different problems that we have separated. See the appendix for more information. We looked more closely at the horizontal coarse graining. From a $512 * 512$ grid, we downscaled the resolutions to either $8 * 8$, $16 * 16$ or $32 * 32$ grids.

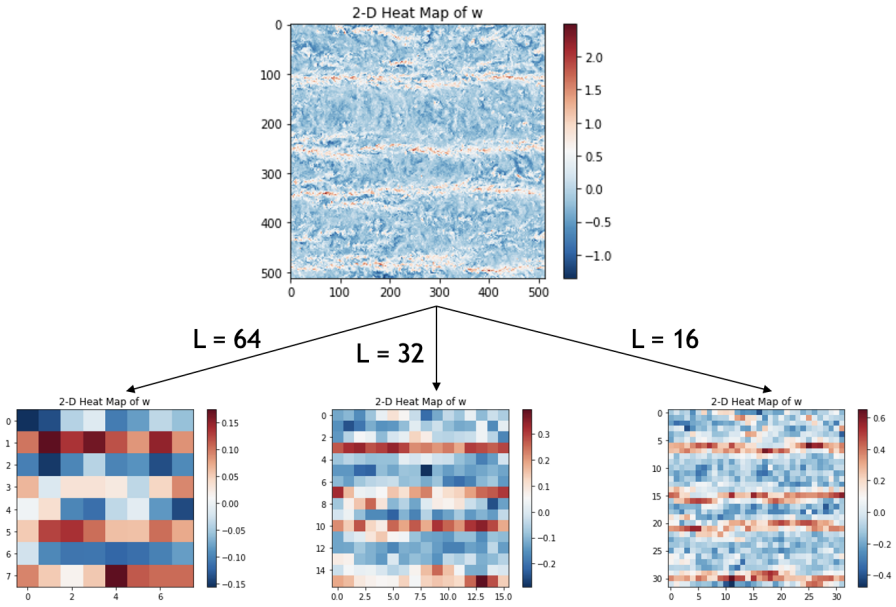


Figure 9: Horizontal coarse graining of the simulation over different scales. L is the coarse graining factor.

Finally, since the whole boundary layer is linked vertically it is important for our model to take a whole vertical vector as input. Indeed, motion at the bottom of the boundary layer can impact what happens at the top. We thus end up with the following data :

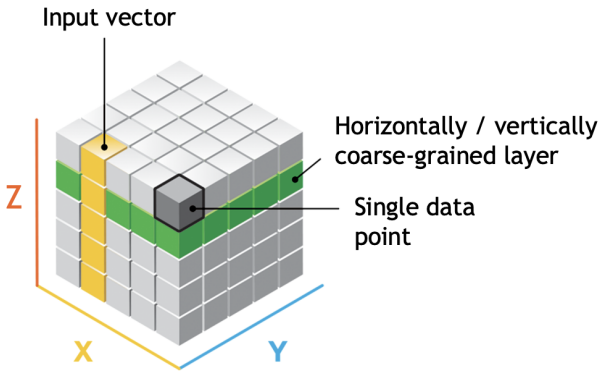


Figure 10: Input data

2.3 Devising a predictive model

We used deep neural networks to predict sub-grid scale heat fluxes, based on the resolved variables u, v, w, θ, s, as well as the turbulent kinetic energy tke which can be computed from the other resolved variables. Over the course of the internship, I started with simple networks and then complexified them in order to gain interpretability and to be more efficient in memory size. We used the Cheyenne [18] and Casper [19] clusters from the National Center for Atmospheric Research to train or models.

The models took as input the variables u, v, w, θ, s and tke. The output of the model is the heat flux $\overline{w'\theta'}$ over the whole boundary layer and at the corresponding (x,y) coordinates in the horizontal plane. Thus, each input sample had a size of $6 * 376$: 6 variables taken over the 376 vertical layers of our simulation. The total number of samples depended on the coarse graining factor : for a coarse graining factor of 32 (meaning that the resolution is 32 times coarser) for example, we had horizontal grids of $16 * 16$ which means that we had $16 * 16 = 256$ samples for each time-step. In total, we had 62 time-steps and thus 15872 samples for a coarse graining factor of 32. We used 80% of the data for the training and 20% for testing.

2.3.1 • Different architectures for different uses

The first model we designed was a simple feed-forward network with fully connected layers (Figure 11). We set manually the depth of the network as well as the number of neurons of each layer and went with a network with three hidden layers with 1024, 512 and 256 neurons respectively. We optimised the hyper-parameters of the model with the library Optuna : the batch size, the number of epochs, the learning rate. In a second step, we also used the library to also optimize the depth and number of neurons on each layer.

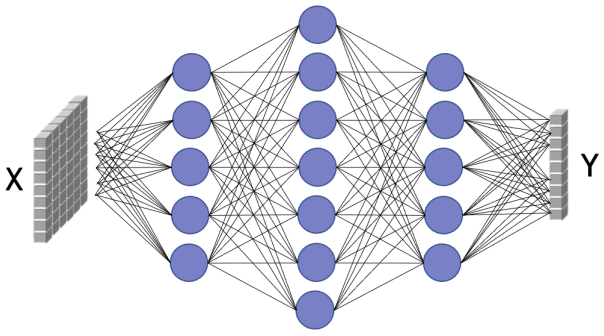


Figure 11: First model

Using this model as a baseline, we tried to improve the model on different aspects :

- The model is relatively large and since the whole boundary layer is linked vertically, a lot of information is redundant and we could achieve similar results with a smaller network.

- It is hard to understand what drives the model and reducing the input dimension could lead to a better understanding of the important variables.
- We have no idea how confident the model is on each prediction. There is no reason to believe that the extreme values of the distribution are as well predicted as the values around the mean.

To address the first two points, we compared different methods : Principal Component Analysis (PCA), Auto-Encoders (AE), Variational Auto-Encoders (VAE) which used different types of convolutional architectures. [20] showed that VAEs can accurately represent convective processes in climate models, while enabling interpretability and better understanding of sub-grid-scale physical processes. Indeed, our models that use AEs or VAEs as a way to reduce the input dimension still feature good performances.

To add an uncertainty measure to our prediction, we decided to add stochasticity to the parametrization [21]. Instead of predicting only one value, we assume that each point is distributed along a normal distribution and we predict a mean and a standard deviation. Therefore, for each point we get an interval in which we have a strong probability to find the true value.

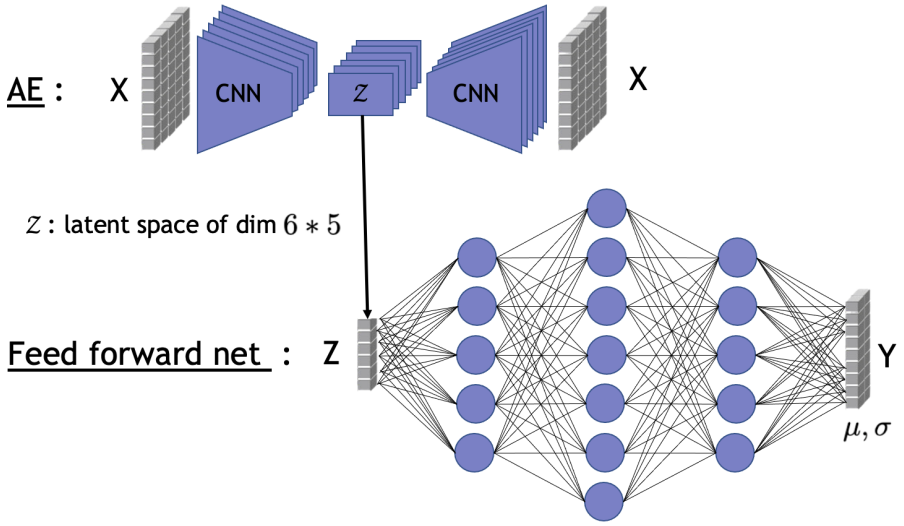


Figure 12: Final model

Our final model (Figure 12) is thus made of an Auto-Encoder and a feed-forward network with fully connected layers which takes as input the encoded values of the original input data. The encoder is made of convolutional layers with one channel for each input variable. It is as if we had one Auto-Encoder for each variable. The latent space is therefore made of 6 latent spaces : one for each variable, of size 5. Most the hyper-parameters of the network (size of the latent spaces, number of hidden layers in the feed forward network, number of neurons,...) were optimized. Even though we often ended up in local minima, the results are better than if we had taken arbitrary values. One noteworthy result is that

the global dimension of the latent space is 30. When performing PCA, we also found that keeping the first 30 most important vectors was enough to represent the original input of dimension $6 * 376$ with great fidelity. Finally, the training was performed using the following loss :

$$\text{loss} = \text{reconstruction loss} + \text{log-likelihood}$$

- the reconstruction loss is the mean squared error between the decoded value of the encoded original input and the original input,
- the log-likelihood corresponds to the likelihood that the output of the network follows the predicted $\mathcal{N}(\mu, \sigma)$ distribution.

2.3.2 • Results

We used two metrics to rate the different models and compare their performances :

- the pixel-wise Mean Squared Error (MSE) between the predicted heat flux and the true heat flux (which is computed based on the high resolution LES),
- the coefficient of determination R^2 :

We note n the number of grid points available (on the coarse resolution grid). For each point we note y_i the true heat flux values and $\hat{y}_i$ the predicted value. The mean of the observed true data (data from the LES simulation) is defined by $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.
R^2 is defined by :

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}$$

In the best case, the modeled values exactly match the observed values, which results in $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = 0$ and $R^2 = 1$. A baseline model, which always predicts $\overline{y}$, will have $R^2 = 0$. Models that have worse predictions than this baseline will have a negative R^2.

The different models that we compare below are :

- The first simple feed-forward network with fully connected layers,
- A convolutional network which reduces the dimension to 6*30 before using fully connected layers to predict the output,
- A model similar to Figure 12 that uses an Auto-Encoder's latent space as input to predict the heat flux,
- The final model depicted on Figure 12 which in addition to the previous model predicts a standard deviation for each point as a measure of its confidence.

Criterion	FF-net	CNN	Conv AE + FF-net	Conv AE + FF-net + gaussian distrib
MSE	0.043	0.045	0.058	0.093
R^2	0.957	0.955	0.942	0.907

Table 1: Model comparison

By using a convolutional network to reduce the input dim by a factor of over 10, and thus reducing further the network size, we manage to keep similar performances in terms of MSE and R^2. The introduction of an Auto-Encoder enables us to reduce the dimension even further with a latent space of $6*5$ instead of $6*30$. We also have a better understanding of the important features inside the input since we can reconstruct it from its encoding. This method yields worse but still accurate results. Finally, introducing the Gaussian distribution yields much worse results but enables us to quantify the uncertainty of the model for each prediction. We take a closer look at the difference induced by the Gaussian distribution in Figure 13.

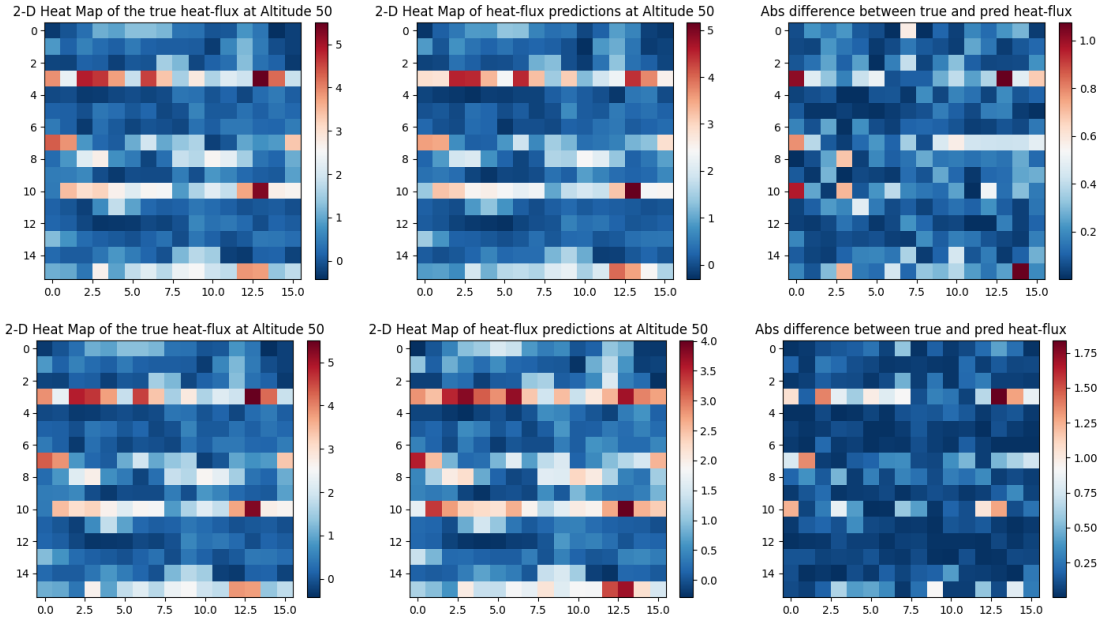


Figure 13: Comparison of the last two models (when adding the Gaussian distribution). First row is without the Gaussian distribution. Second row is with the Gaussian distribution. First column is the ground truth. Second column is the predicted values. Third column is the pixel-wise absolute difference between ground truth and prediction.

What appears here is that adding the Gaussian distribution for each prediction induces a higher error for extreme values predictions. The model tends to under-estimate the high values in the turbulent areas but we still keep the global aspect of the turbulent motions. Depending on the usage, one may prefer a model that has a low error to a model that can predict a confidence interval or on the contrary one may want to have a confidence interval even if it means being less precise (when studying extreme values for example). In some cases, the uncertainty might also come from the original data, in which case having a low-error model but no confidence interval does not make much sense.

3
MESH AGNOSTIC MODELS FOR SUB-GRID FLUXES PREDICTION

3.1 The Vertical problem : vertical interpolation

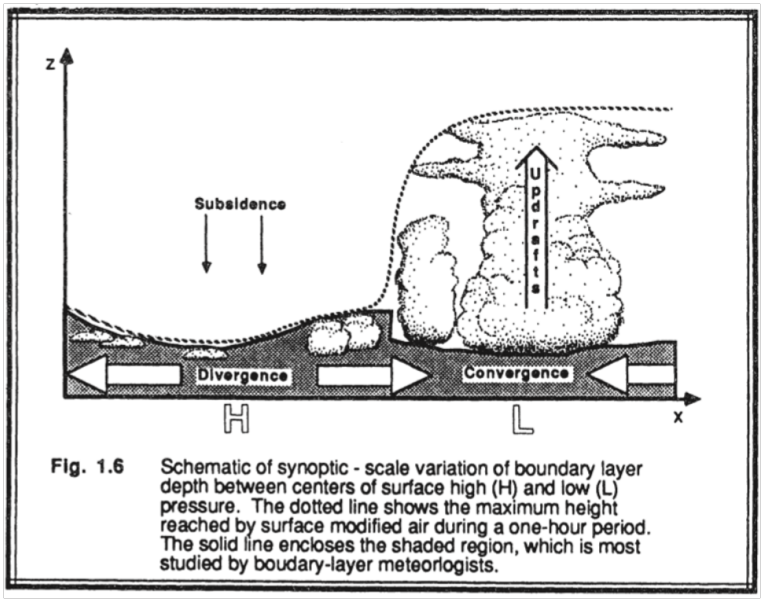


Fig. 1.6 Schematic of synoptic - scale variation of boundary layer depth between centers of surface high (H) and low (L) pressure. The dotted line shows the maximum height reached by surface modified air during a one-hour period. The solid line encloses the shaded region, which is most studied by boudary-layer meteorlogists.

Figure 14: Boundary layer height variations [2].

From our LES simulations, we only have fixed settings which result in training our models on a fixed boundary layer height. Therefore, since the whole boundary layer is linked and still dependent on the altitude, we have two problems when generalizing our model :

- How do we cope with simulations with a different boundary layer height (and therefore simulation points at different altitude) ?
- How do we adapt our model which takes a fixed number of vertical point as input (376 in our case) to a coarser simulation ?

Inspired by the work in [22], we looked at interpolation and extrapolation techniques. Since we had trained models on a fixed height with a fixed number of points at altitudes z_i, the goal was to take any simulation which resolved points at altitudes z_i' and predict the values of the resolved variables at the altitudes z_i. From there, we can pass the predicted values as input to our model. This framework allows us to predict fluxes for points up to the height of the boundary layer we trained our model on. This is enough for simulations with a lower or equal boundary layer height.

For interpolation, we compared linear interpolation, cubic interpolation and optimized Convolutional Neural Networks (CNN) :

Criterion	Linear	Cubic	CNN
MSE	0.059	0.060	0.187
R^2	0.940	0.940	0.813

Table 2: Interpolation comparison with a coarse-graining factor of 2.

It appears that for interpolation tasks, linear or cubic interpolation is enough. The problem arises when we consider simulations with a smaller boundary layer and we thus have to extrapolate the points above. In this case, linear and cubic methods did not fare well. Therefore we have restricted our following work to simulations with a similar boundary layer height where those techniques are good enough.

3.2 The horizontal problem

In section 2, we trained our model only on one coarse graining level. Here are the performances when we try to predict sub-grid heat fluxes with this model on different grid scales :

Training coarse factor	Testing coarse factor	performance : MSE (R^2)
32	16	0.319 (0.681)
32	32	0.093 (0.907)
32	64	0.279 (0.721)

Table 3: Generalization from training on single coarse-graining level

The model cannot generalize to different scales. We now look at different coarse-graining levels for training. The goal is to train a model that learns across the different coarse-graining scales. In other words, we want a model that can predict sub-grid scale processes at any given resolution.

When coarse-graining on different scales, we find that we have a problem of data imbalance between different classes. Indeed, just by looking at Figure 9, we see that we have far more samples (ie : vectors in the horizontal plane) for low coarse graining factors than for higher ones. Thus, we have 2^2 times as much data for a coarse-graining factor $L = 16$ than for a coarse-graining factor $L = 32$ and again we have we have 2^2 times as much data for a coarse-graining factor $L = 32$ than for a coarse-graining factor $L = 64$. How can we devise a model that manages to generalize given this imbalance ?

We explored different ideas such as importance sampling for deep learning framework [23], or meta learning to solve the problem of imbalance-sensitive networks [24].

3.2.1 • Mixed training dataset with data imbalance

The first approach was simply to take all the data available and train the model on it without differentiating the input on the coarse-graining level. This led to poor results with the model learning way too much on the most recurring data ($L = 16$) and too little on the rare data ($L = 64$).

We then decided to add a weight on the loss depending on the coarse-graining factor in order to give more importance to data that is not often seen. This technique however did not work as well as intended, even if it was better than the first approach.

Finally, what seemed to work best was to take as much data from each coarse graining factor at each epoch combined with the weight system. This means that for each epoch, all samples for the coarse-graining factor $L = 64$ is seen, a quarter of the samples for $L = 32$ is seen and $\frac{1}{16}$ of the data for $L = 16$ is seen. For each epoch, we select the data samples and their order at random.

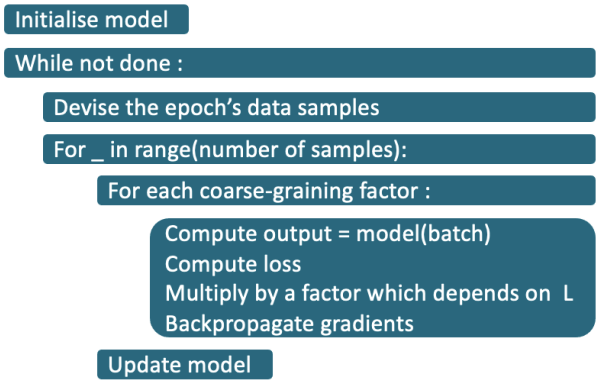


Figure 15: Training algorithm to address data imbalance.

3.2.2 • Simulation resolution as an input variable

The idea of adding the horizontal resolution as an input to the model the was explored briefly but did not yield great results and we wanted to explore more part 3.2.3 so we decided to leave it for later work. The idea was to add the resolution directly as input to the network with fully connected layers in Figure 12.

3.2.3 • Meta Learning for grid-agnostic models

Meta-learning, or the idea of learning how to learn is a systematic approach to efficiently learn generalizable meta-features and apply them to new prediction tasks given very limited data. This framework of learning is particularly important in the climate sciences where data in many data-driven parametrization and/or forecasting model tends to be imbalanced, incomplete, and noisy. The idea behind meta-learning is to train a meta model and local models simultaneously [25].

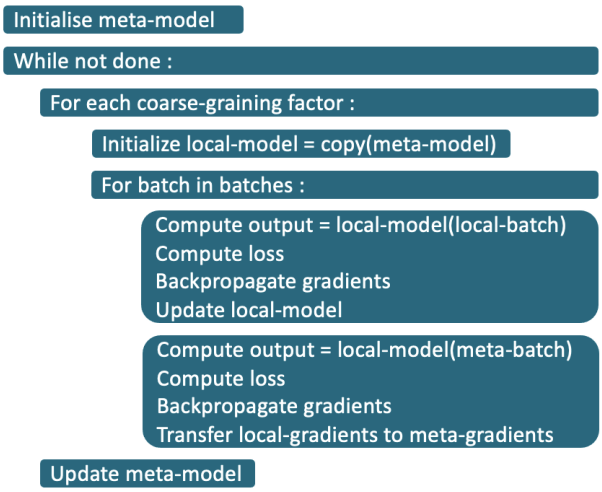


Figure 16: Training algorithm for meta-learning.

For each epoch, we use the meta model to initialize the local models. We then update the local models independently. We can then update the meta-model by back-propagating the loss through the local models and onto the meta-model. This method enables local models to use the the data in the other classes as well for the training. In our case, each local model corresponds to a coarse-graining factor. Mathematically, we have :

The local parameters θ_i are updated in a regular way : $\theta_i = \theta_i - \alpha_i \nabla_{\theta_i} \mathcal{L}_{T_i}(f_{\theta_i})$

We then update the meta parameters θ with $\theta = \theta - \beta \nabla_\theta \sum_i \mathcal{L}_{T_i}(f_{\theta_i})$

The problem with this method is that it requires second order gradients (θ_i depends on θ and the loss regarding the i^{th} coarse-graining factor depends on θ_i). Therefore, to simplify computations, we approximate the update by : $\theta \sim \theta - \beta \sum_i \nabla_{\theta_i} \mathcal{L}_{T_i}(f_{\theta_i})$

We can also add weights that depend on the coarse-graining factor. Therefore we get :

$$\theta \sim \theta - \sum_i \beta_i \nabla_{\theta_i} \mathcal{L}_{T_i}(f_{\theta_i})$$

An important note is that such methods in sections 3.2.1 and 3.2.3 are theoretically better than what we did in section 2. Indeed, if we take weights of 0 for every coarse-graining factor, except for $L = 32$ for which the weight is set to 1, we are again in the configuration of section 2 with a single coarse-graining factor for the input. Therefore if we optimize the weights, we should get performances that are at least as good as in section 2.

3.2.4 • Results

After trying to optimize all the hyper-parameters of the different models, we get the following performances :

Training	Testing	Classic data imbalance	Meta-Learning
16,32,64	16	0.303 (0.697)	0.315 (0.685)
16,32,64	32	0.380 (0.620)	0.772 (0.228)
16,32,64	64	0.618 (0.382)	0.653 (0.347)

Table 4: Comparison between the frameworks in sections 3.2.1 and 3.2.3. Results are formatted as : [MSE (R^2)]

The results are way worse than what we initially had with section 2's framework (cf Figure 3). Such results can only mean that we have not optimized our hyper-parameters well enough. Indeed, there are a lot more hyper-parameters to tune : we have multiple local models to optimize as well as multiple weights that depend on the coarse-graining factors. Therefore, when optimizing, we often ended up in local optima and have not yet managed to find the right parameters that end up in better performances. This work is still left to be finished.

We can then also evaluate our models by training them only on some coarse-graining factors and testing them on a different one to see how the models can generalize through interpolation or extrapolation of coarse-graining levels.

CONCLUSION

Data-driven modeling for climate science is an emerging scientific field where a lot can be done to improve current models that only rely on the exact resolution of the governing equations. In particular, deep learning opens new prospects in this field to automatically learn the dependencies between variables.

During this internship, I was able to grasp the different challenges at hand in this field even though I had never studied geosciences before. I was able to attend multiple conferences and weekly working groups with researchers in the field working from San Diego in California to Grenoble in France. Such discussions helped me greatly understand the challenges mentioned above. It also helped me regarding my future orientation and the type of subject that I would like to work on for a Ph.D.

Back to the internship subject, we have built a framework for predicting sub-grid scale processes across different resolutions. The final goal of this work is to apply such predictions to parameterize unresolved processes in simulations at a scale ranging from the scale of Large-Eddy Simulations to the scale of Regional Climate Models. The model is not yet tuned to yield good performances but the theoretical guarantees it provides make the subject worth investigating further.

REFERENCES

[1] Milton Van Dyke. *An album of fluid motion*. International series of monographs on physics. The Parabolic Press, 1982. ISBN: 09157600209.

[2] Roland B. Stull. *An introduction to boundary layer meteorology*. International series of monographs on physics. Kluwer Academic Publishers, 1988. ISBN: 9789027727695.

[3] A Arakawa, J-H Jung, and C-M Wu. "Toward unification of the multiscale modeling of the atmosphere". en. In: *Atmos. Chem. Phys.* 11.8 (Apr. 2011), pp. 3731–3742.

[4] Masaki Satoh et al. "Global cloud-resolving models". en. In: *Curr. Clim. Change Rep.* 5.3 (Sept. 2019), pp. 172–184.

[5] Rachel Honnert et al. "The atmospheric boundary layer and the "gray zone" of turbulence: A critical review". en. In: *J. Geophys. Res.* 125.13 (July 2020).

[6] Seung-Bu Park et al. "Coherent structures in the boundary and cloud layers: Role of updrafts, subsiding shells, and environmental subsidence". en. In: *J. Atmos. Sci.* 73.4 (Apr. 2016), pp. 1789–1814.

[7] Markus Reichstein et al. "Deep learning and process understanding for data-driven Earth system science". en. In: *Nature* 566.7743 (Feb. 2019), pp. 195–204.

[8] Rui Wang, Robin Walters, and Rose Yu. "Incorporating symmetry into deep dynamics models for improved generalization". In: (Feb. 2020). arXiv: 2002.03061 [cs.LG].

[9] Rui Wang, Robin Walters, and Rose Yu. "Approximately equivariant networks for imperfectly symmetric dynamics". In: (Jan. 2022). arXiv: 2201.11969 [cs.LG].

[10] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance". In: *Journal of Fluid Mechanics* 807 (2016), pp. 155–166. DOI: 10.1017/jfm.2016.615.

[11] Christopher J Anders et al. "Software for dataset-wide XAI: From local explanations to global insights with Zennit, CoRelAy, and ViRelAy". In: (June 2021). arXiv: 2106.13200 [cs.LG].

[12] Nikolaos D Katopodes. *Turbulent Flow*. Elsevier, 2019, pp. 639–640.

[13] Luigi C. Berselli, Traian Iliescu, and William J. Layton. *Scale similarity models*. Springer-Verlag, 2006, pp. 195–224.

[14] Janni Yuval and Paul A O'Gorman. "Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions". en. In: *Nat. Commun.* 11.1 (July 2020), p. 3295.

[15] P Gentine et al. "Could machine learning break the convection parameterization deadlock?" en. In: *Geophys. Res. Lett.* 45.11 (June 2018), pp. 5742–5751.

[16] Yu Cheng et al. "Deep learning for subgrid-scale turbulence modeling in large-eddy simulations of the convective atmospheric boundary layer". en. In: *J. Adv. Model. Earth Syst.* 14.5 (May 2022).

[17] Rui Wang et al. "Towards physics-informed deep learning for turbulent flow prediction". In: (Nov. 2019). arXiv: 1911.08655 [physics.comp-ph].

[18] Computational and Information Systems Laboratory. *Cheyenne: HPE/SGI ICE XA System*. Boulder, CO: National Center for Atmospheric Research. 2019. DOI: 10.5065/D6RX99HX.

[19] Computational and Information Systems Laboratory. *Casper : HPE/SGI ICE XA System*. Boulder, CO: National Center for Atmospheric Research. 2019. DOI: 10.5065/D6RX99HX.

[20] Gunnar Behrens et al. "Non-linear dimensionality reduction with a Variational Encoder Decoder to understand convective processes in climate models". In: (Apr. 2022). arXiv: 2204.08708 [physics.ao-ph].

[21] Arthur P Guillaumin and Laure Zanna. "Stochastic-deep learning parameterization of ocean momentum forcing". en. In: *J. Adv. Model. Earth Syst.* 13.9 (Sept. 2021).

[22] Oussama Boussif et al. "MAgNet: Mesh Agnostic Neural PDE Solver". In: *ICML 2022 2nd AI for Science Workshop*. 2022. URL: https://openreview.net/forum?id=tbIJmAdqYc8.

[23] Angelos Katharopoulos and François Fleuret. "Not all samples are created equal: Deep learning with importance sampling". In: (Mar. 2018). arXiv: 1803.00942 [cs.LG].

[24] Mateusz Ochal et al. "How Sensitive are Meta-Learners to Dataset Imbalance?" In: (Apr. 2021). arXiv: 2104.05344 [cs.LG].

[25] Pytorch Lightning. *"Tutorial 12: Meta-Learning - Learning to Learn"*. Accessed Aug 10, 2022, https://pytorch-lightning.readthedocs.io/en/stable/notebooks/course_UvA-DL/12-meta-learning.html.

APPENDIX

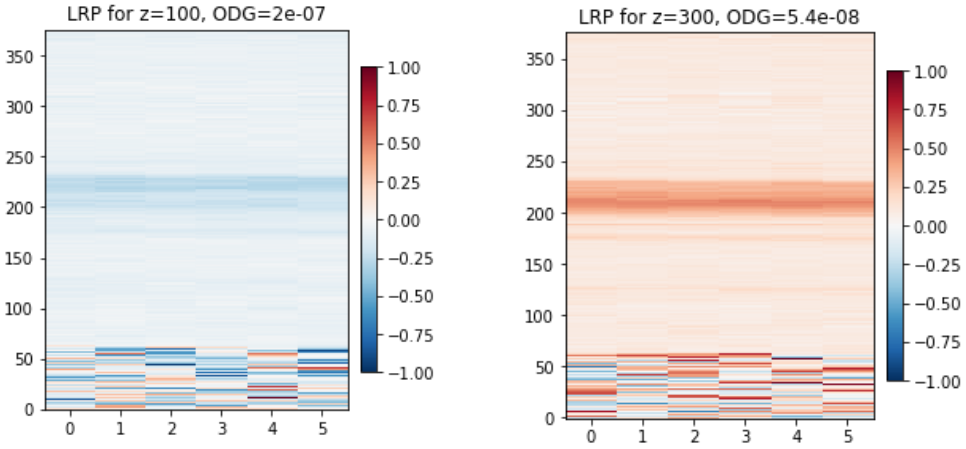


Figure 17: Layer-wise Relevant Propagation (LRP) for two different altitude heat flux predictions. Vertically we have each vertical layer. Horizontally we have each input variable.

Layer-wise Relevant Propagation shows that the whole boundary layer is important to predict a given altitude. Admittedly, the points around 200^{th} and 250^{th} as well as the bottom of the boundary layer seem more important than the others but they still all are important.

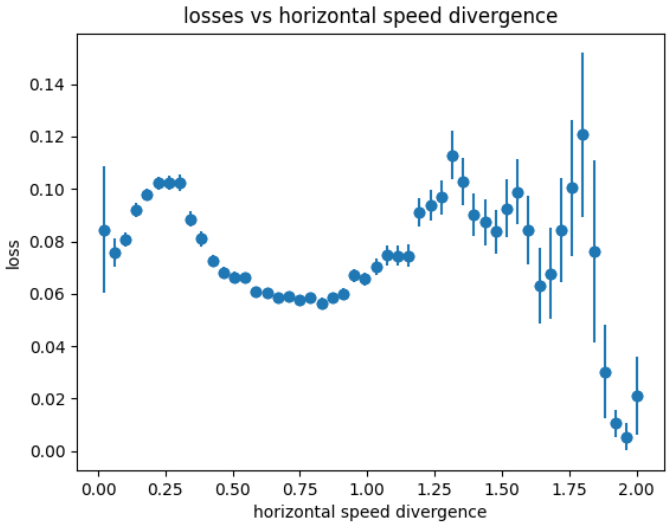


Figure 18: Study of the loss against the velocity divergence at a given point.

Figure 18 shows that there is no clear link between the horizontal variations of the wind velocity and error of our model. Therefore, there is no need to add horizontally adjacent points to the input of our model. Contrary to the vertical axis, there is no clear link between points in the horizontal plane.