Institute for Earth and Surface Dynamics,
Université de Lausanne

Internship Report

Data-driven Equation Discovery of the Relationship Between Tropical Precipitation and its Large-Scale Environment

---

Intern :
Jo LECUYER

Tutor :
Tom BEUCLER

September 4, 2023

Contents

Definitions and Notations **2**

1 Tropical Precipitation parametrization **5**
1.1 Motivation . . . 5
1.2 Past works and methods . . . 5
1.2.1 Symbolic Regression methods . . . 5
1.2.2 Analytical Baseline . . . 6
1.3 Data . . . 8
1.3.1 Data acquisition . . . 8
1.3.2 Variables used . . . 9

2 Methodology Proposed **11**
2.1 Framework, Metrics, Baselines . . . 11
2.1.1 Features and Target . . . 11
2.1.2 Metrics . . . 12
2.2 Neural Networks, Feature Selection and Kernel Discovery . . . 13
2.2.1 Neural Network baselines . . . 13
2.3 Kernel Learning . . . 14
2.3.1 Canonical Correlation Analysis . . . 15
2.3.2 Neural Network Kernel Layer . . . 16
2.3.3 Function Kernels . . . 16
2.4 Symbolic Regression . . . 17

3 Results **17**
3.1 Methodology and Training procedures . . . 18
3.2 RH-based NN example . . . 19
3.3 Feature Selection . . . 21
3.4 Kernel Learning . . . 21
3.4.1 Canonical Correlation Analysis . . . 21
3.4.2 Kernel Layer . . . 22
3.4.3 Function Kernels . . . 22
3.5 Symbolic Regression . . . 27
3.5.1 200 - 1000 hPa kernels . . . 27
3.5.2 500 - 1000 hPa kernels . . . 28
3.6 Pareto Plan . . . 29
3.7 Pareto optimal equations . . . 33

Definitions and Notations

CRH	Column Relative Humidity
CWV	Column Water Vapour
DSE	Dry Static Energy
$ERA5$	ECMWF Reanalysis v5
$FFNN$	Feed Forward Neural Network
L_v	Latent heat of water vapourisation
MSE	Mean squared Error
P	Total ERA5 precipitation
RH	Relative humidity
R^2	Coefficient of determination
R_d	Specific gas constant for dry air
R_v	Specific gas constant for water vapour
T	Temperature of air in Kelvins
ϕ	Any ERA5 variable, other than precipipation
σ_o	ERA5 standard deviation of subgrid-scale orography
θ	Potential temperature
θ_e	Potential equivalent temperature
θ_e^*	Saturated potential equivalent temperature
θ_e^+	Mesure of subsaturation, as defined by $\theta_e^+ = \theta_e^* - \theta_e$
g	Gravity constant
p	atmospheric pressure
p_0	Reference atmospheric pressure, taken as 1000 hPa
q	Specific humidity in kg.kg^{-1}
q_i	Ice water content in kg.kg^{-1}
q_l	Liquid water content in kg.kg^{-1}
q_s	Saturation Specific Humidity in kg.kg^{-1}
t	time

Acknowledgments

Working on this project at '∂AWN has been exciting and fulfilling. I learned invaluable lessons on atmospheric sciences and machine learning, but also on how to properly conduct a research project, with all the difficulties that this entails.

To begin with, I would like to thank my Internship tutor Dr. Tom Beucler. for his presence and continuous help throughout the internship, and for his impressive interdisciplinary knowledge of atmospheric sciences and machine learning. I'd also like to thank him for all his invaluable advice on research in general, whether in terms of working methods or communicating results.

I would also like to thank Pr. David Neelin and Dr. Fiaz Ahmed at UCLA's Department of Atmospheric and Oceanic Sciences, for their in-depth knowledge of tropical convection mechanisms and their scientific rigor, as well as for their availability and the always fascinating advice and ideas they gave me during our discussions throughout this internship.

I'm also grateful to French professors Hervé le Treut, Alexis Tanted and especially to Thomas Dubos, who introduced me to meteorological sciences at Polytechnique, and was the one that made this internship possible, for which I am grateful.

I would also like to express my thanks all the Ph.D. students, post-docs, and interns from the ∂AWN laboratory, Saranya Ganesh Sudheesh, Milton Gomez, Frederic Iat-Hin Tam, Marine Berthier, Aser Atawya, For the great help, availability and kindness they gave me when I joined the team and throughout the rest of the internship. I would also like to thank the Administrative Officer of our department, Sabrina Damiani, who greatly helped me to settle into excellent conditions at UNIL to pursue my laboratory research in the best possible way.

Introduction

The tropics, with their intricate climate dynamics and profound societal implications, have long intrigued climate scientists. Central to the understanding of tropical climate systems is the accurate representation of precipitation in climate models. Tropical precipitation is not merely a meteorological curiosity; it holds immense practical significance. These regions are home to a substantial portion of the global population and are particularly vulnerable to the impacts of climate change. The ability to predict and model tropical rainfall patterns is crucial for various applications, including agriculture, water resource management, and disaster preparedness.

In recent years, the intersection of scientific advancement and computational capabilities coupled with data availability has reinvigorated the pursuit of precise cloud systems parametrization ([7],[15]), notably through the use of Machine Learning (ML) based methods. However, these methods, while sometimes producing impressive results, often depend on tens of thousands of parameters, making for models that are difficult to read, and remain difficult to understand or explain physically. To alleviate this problem, in this study, we will seek to combine the performance of neural networks with the physical readability of our models, in particular by using Symbolic Regression methods, which allow us to derive analytical equations in a data-driven manner.

1 Tropical Precipitation parametrization

1.1 Motivation

Due to computational constraints, climate models used to make future projections spanning multiple decades typically have horizontal resolutions of 50 – 100 km ([14]). Even with our modern computation capability, climate models typically run at a horizontal resolution of 25 km: that is their 'grid-scale'. This coarse resolution means that a lot of sub-grid scale processes, that occur at smaller characteristic lengths like cloud microphysics are not directly computed by the models. However, they need still to be taken into account somehow due to their relevance in climate dynamics and are therefore parametrized: using grid-scale information, a parametrization tries to estimate the results (e.g., precipitation) of sub-grid scale phenomenon (clouds formations and microphysics), with various methods (empirical functions, neural networks..). These parametrizations can vary a lot between climate models and therefore cause divergences between them for future climate forecasting. Especially, clouds parametrization is believed to be "the greatest source of uncertainty in climate projections".([15]).

With the recent rise of Machine Learning (ML), due to larger and larger data availability and refined algorithms, ML-based parametrization for various problems have been developed, and also consequently for clouds and precipitation ([16],[7],[13], [9]). Using very little domain knowledge or physical hypothesis, ML-based models can develop highly nonlinear parametrization, that is not limited to a specific functional form like standard regression, and with notable performance increase. However, these models usually act like 'black boxes', relying on millions of parameters, and can prohibit the understanding of the physical meaning underlying their parametrization. Moreover, the plethora of parameters used can downgrade the generalization capability of these models, when presented with inputs that were out of the training datasets distribution: i.e, a warmer climate [3].

The questions that this study will try to answer are as follow : Can we develop a data-driven precipitation parametrization, that approaches the performance of NN-based models while keeping the formulation sparse and physically interpretable?

1.2 Past works and methods

1.2.1 Symbolic Regression methods

To answer the main question that we just outlined, we will deploy a Pareto-plan framework on the parametrization that we will develop. Hence, the evaluation of our models centers on their Pareto optimality, which gauges their status as the most performing models relative to their complexity. We can frame their 'performance' as some score on metrics like R^2 that we'll describe in more detail later in the study, and their 'complexity' as the number of trainable parameters of each model.

On one end of the complexity spectrum, one could find heavy Deep-Learning models that require $\mathcal{O}(10^8)$ parameters, like [4], and on the other hand some analytical parametriza-

tions with under 10 trainable parameters (e.g [2]).To strike a balance between these analytical mathematical formulations and high-performing neural networks, the primary aim of this study was to apply was to deploy data-driven equation discovery through cutting-edge Symbolic Regression techniques.

In symbolic regression, in contrast to conventional regression, we begin by defining a suite of mathematical operators rather than a set of basic functions. For instance, the introduction of division as a mathematical operator permits us to introduce rational non-linearities to our models; coupled with other traditional mathematical operators and analytical functions, the symbolic regression library (in our case `PySR`) generates an initial pool of equations randomly. Taking inspiration from the concept of natural selection in evolutionary theory, symbolic regression is typically executed as a genetic algorithm. This algorithm iteratively applies operations driven by genetic principles, such as selection, crossover, and mutation, to the collection of candidate equations. At each iteration, the equations are ranked based on their performance and simplicity. The most high-performing equations are chosen to form the succeeding generation of equations

An inherent advantage of training or discovering analytical models, as opposed to employing neural networks, is the instantaneous comprehension of the model's content. This encompasses the assessment of whether the model adheres to physical constraints. Furthermore, analytical models allow for the direct analysis of their structure using robust mathematical tools like perturbation theory and numerical stability analysis. Additionally, analytical models are highly communicable within the scientific community, amenable to numerical implementation, and exhibit efficient execution, especially when optimized implementations of well-known functions are available.

Significantly, [17] marked the pioneering use of automated, data-driven equation discovery in climate-centric applications. They harnessed sparse regression, particularly a relevance vector machine, to unearth an analytical model characterizing ocean eddies based on highly idealized data. Sparse regression, in this context, involves the user defining a library of terms, with the algorithm subsequently discerning a linear combination of these terms that optimally fits the data while minimizing the term count used. Their data-driven equations managed to outperform Deep-Learning algorithm (Convolutional Neural Networks in this case) while having better generalization capability. [8] developed a Data-Driven cloud cover Parametrization with Symbolic Regression using `PySR`. They trained their parametrization on high-resolution data from storm-resolving models and managed to develop sparse analytical relationships that have generalization capability to other models (like ERA5) and that beat the existing Cloud Cover parametrization like the Sunqdvist schemes.

1.2.2 Analytical Baseline

The main model that we will try to build upon is a buoyancy-based tropical precipitation parametrization introduced in [2] and further developed in [1]. One key aspect it explores is the role of water vapor in the atmosphere. It's well-established that increased water vapor typically leads to more intense tropical convection. For example, when the atmo-

sphere is loaded with moisture, you often see intense tropical storms and heavy rainfall. However, other factors like temperature, air pressure, and wind patterns also influence convection. The challenge is that these factors interact in intricate ways, making it difficult to pinpoint precisely how the environment affects convection.

The article tackles this complexity by employing statistical approaches to examine broader patterns, unraveling mean relationships between moisture and temperature variables in the form of the integral of potential equivalent temperatures in different height regions.

One key finding highlighted in the article is the concept of "precipitation onset." This refers to the point at which precipitation rapidly increases as atmospheric moisture content goes up. To put it simply, as the air gets moist, you tend to see a sudden uptick in rainfall. During this study, we will try also evaluate our models on their capability to capture this onset.

The article suggests that the increase in precipitation with moisture is primarily due to buoyancy. Imagine a rising air parcel—often associated with convection. When this parcel reaches a certain moisture level, it becomes positively buoyant near the freezing level. This means it rises more vigorously, leading to stronger precipitation. The article then develop a buoyancy measure called B_L

$$B_L = g\left[w_B\frac{(\theta_{eB}-\theta^*_{eL})}{\theta^*_{eL}} - w_L\frac{\theta^+_{eL}}{\theta^*_{eL}}\right] \tag{1}$$

with w_B,w_L being trainable parameters, θ_{eB} the integral of θ_e, the potential equivalent temperature, in the boundary layer,θ_{eL} the integral on the lower troposphere. (see section 1.3.2 for details on the θ-like variable). In [1] they link this buoyancy measure to precipitation, with a simple linear model

$$P = \alpha H(B_L - B_c)(B_L - B_c) \tag{2}$$

with P the precipitation,α, B_c trainable parameters and H the Heaviside function. This precipitation parametrization will act as our baseline in this study (called the 'B_L' or 'AN18' baseline) which we will try to build upon.

This buoyancy-based explanation doesn't directly account for frontal precipitation or stratiform rain, even though these are related to buoyant convective rain. This is another potential improvement that we will try to develop with our SR models.

The study primarily focuses on tropical oceanic regions but also looks at tropical land regions. It suggests that the relationship between moisture and precipitation onset can differ between land and ocean. For instance, precipitation onset over land can occur at smaller moisture levels compared to the ocean. This variation is attributed to differences in vertical moisture distribution. In this study, we'll try to provide a land and ocean parametrization, notably through the use of orography-based variables to account for these differences.

1.3 Data

As said earlier, we will follow in this study a data-driven approach to find new precipitation parametrization. To continue on the path opened up by the articles [2] and [1], we will use real-world data, not idealized models or high-resolving ones. An argument could be made to use a high-resolution model to first learn the structure of the parametrization equations(e.g. [8]), and then retrain this model on real-world data. However, we chose on this study to see whether or not we could develop a satisfying model starting directly from real-world data. One of the advantages of this approach is that it avoids learning the biases inherent in the results of an idealized model.

1.3.1 Data acquisition

Our research hinges on the ERA5 Reanalysis dataset, a powerful tool for advancing tropical precipitation prediction. Developed by the ECMWF, this dataset merges diverse observational data using advanced assimilation techniques and backs up these observational data with meteorological models such as the International Forecast Systems (IFS) to interpolate them on a complete spatiotemporal grid [10]. ERA5 provides fine-grained spatial (grid) and temporal (dating back to 1979) resolution, making it an ideal resource for investigating short-term weather events and long-term climate trends in tropical regions.

The ERA5 dataset is functioning at a spatial resolution of 0.25°on both longitude and latitude and an hourly resolution on time. We chose to investigate the tropical regions, thus restraining the global grid to 25°north and south on latitudes, ending with a grid composed of 1440 longitudinal points times 200 latitude points. For the 3D variables, we used pressure level coordinates and selected 21 pressure levels from 1000 hPa to 10 hPa.

The total precipitation variable, i.e. our prediction goal, was initially taken from the TRMM-3B42 dataset, which operates at a 3-hourly time resolution. We made this choice following [2], which was a study we were trying to build upon. However, after further investigation, the ERA5 total precipitation was found much easier to predict, with R^2 jumping from around 0.2 (when trying to predict the TRMM precipitation with ERA5 variables) to 0.5 when trying to predict the ERA5 precipitation. This major discrepancy could stem from multiple causes, for example, the fact that the horizontal grids between TRMM and ERA5 do not align, thus needing to interpolate one dataset into another, therefore adding noise and uncertainty. This discrepancy could also arise from the fact that all ERA5 products are not observational data but rather a reanalysis, computed with climate models introducing their biases. One could argue that since the ERA5 total precipitation is computed using other ERA5 state variables, the biases go "in the same direction", leading to an easier prediction of precipitation using ERA5 to ERA5 rather than ERA5 to TRMM.

Using the ERA5 total precipitation, we nonetheless kept the 3-hourly resolution used in TRMM and therefore coarsened our hourly ERA5 data. The 3-hourly total precipitation was simply computed as a sum of the three hourly values in ERA5. For all the

other time-dependent variables used in this study, we opted for an instantaneous choice, meaning that we take the ERA5 value of the first of the three 1-hourly values, and take it as our 3 hourly value, resulting in a 3-hourly "snapshot" of the atmosphere rather than an accumulated view (e.g we could have took the mean of the three hours).

1.3.2 Variables used

Relative Humidity RH The inclusion of RH from ERA5 is pivotal. It furnishes us with essential insights into atmospheric moisture, a key determinant in predicting precipitation, particularly in tropical environments. The knowledge of the RH values on the full pressure column is crucial to predicting convection and precipitation phenomena. This variable is dimensional and expressed in percentage

Temperature T ERA5's temperature data is another crucial variable, in our analysis, helping us comprehend temperature patterns and their significant role in initiating precipitation events within tropical regions ([18]). It will be expressed in K throughout the study

Specific Humidity q**,** The specific humidity variable q (expressed in $kg.kg^{-}1$)from ERA5 is another valuable addition. It enriches our grasp of how moisture content influences tropical precipitation processes. Even if it's supposed to bear the same information as Relative Humidity, minus the temperature information (RH depending on q and T only), this variable is still very useful if we aim to create sparse relationships: having RH and q saves us some transformations that would otherwise need to be done in the $\mathcal{G}_\chi$ formulation.

Ice Water Content q_i **and Liquid Water Content** q_l Incorporating q_i and q_l data from ERA5 broadens our perspective. These variables shed light on phase transitions involved in tropical precipitation, including freezing, melting, and condensation phenomena. They complement the humidity-based variables and will help us derive a precipitation parametrization formulation. One of their drawbacks is that they are in a lot of models not prognostic. Prognostic means that they are not integrated in times via Partial Differential equations, as opposed to T, q and $\mathbf{U}$ that are integrated using the Navier-Stokes equations in a hydrostatic model. For example in the International Forecast System, the models that back up ERA5, q_i and q_l are not prognostic ([5]). The opposite of prognostic is diagnostic: the variables are obtained through a parametrization, at each time step. For example, in our case, precipitation is a diagnostic variables: using ideally prognostic variables, we aim to propose a diagnostic (a parametrization) for precipitation at each time step and grid points.

Equivalent Potential Temperature θ_e**,** θ_e^* **and** θ_e^+ θ_e (expressed in K) is a fundamental thermodynamic parameter.It quantifies the temperature a parcel of air would attain if lifted adiabatically to a reference pressure level while being saturated with water vapor. As such, its value it's more conserved than the traditional potential temperature

θ, during a convection event which leads to condensation. These convection events potentially lead to precipitation, thus explaining the use of these variables in our study [6]. It is expressed as

$$\theta_e = T\left(\frac{p_0}{p}\right)^{R_d/(c_d+r_t c)} RH^{-r_v R_v/(c_d+r_t c)} \exp\left[\frac{L_v r_v}{(c_d + r_t c)\, T}\right] \tag{3}$$

With r_t and r_v the total water content and water vapor mixing ratios, R_d,R_v the specific gas constants for dry air and of water vapor, c_d and c_l the specific heat capacities of dry air and of liquid water and L_v is the latent heat of vapourization of water. In the context of precipitation prediction, a higher θ_e signifies atmospheric instability, which promotes convection and precipitation. By integrating θ_e data from ERA5, our research gains deeper insights into the conditions conducive to convective activity and precipitation initiation within the complex tropical climate system. We will also use θ_e^*, which is the value of θ_e if the air was at saturation; and $\theta_e^+ = \theta_e^*$ - θ_e, as a measure of subsaturation.

Moist and Dry Static Energy Moist static energy (MSE) and dry static energy (DSE) are essential thermodynamic quantities used to evaluate the potential for atmospheric motion and convection. MSE is defined as follows:

$$MSE = cT + gz + L_v q \tag{4}$$

Where g is the acceleration due to gravity,z is the altitude. Dry Static Energy is defined as

$$MSE = cT + gz \tag{5}$$

These variables are similar to θ_e and θ, respectively. As with q and RH, we use them even if they seem redundant with the θ because they might help in formulating a sparse relationship. Moist static energy accounts for the latent heat associated with water vapor, making it a key parameter in assessing atmospheric instability and the potential for convection. Dry static energy, on the other hand, represents the energy available for lifting air parcels without considering moisture effects. Both of these energy variables are integral to our analysis, aiding in the evaluation of atmospheric stability and convection potential within tropical regions.

Orography based-variables To account for the land-sea separation between precipitation regimes, we include orography-based variables. First, we use the land-sea mask (LSM), which ranges on a scale of 0 to 1 and indicates the proportion of land over the sea on a given grid point. We will also go further in the land description in the form of four measures of the sub-grid scale orography. The standard deviation of orography, denoted σ_o thereafter, is a scalar representing the standard deviation of height; we also investigated the mean-slope of the sub-grid scale orography, and its anisotropy. We introduce a form of normalization for the standard deviation of orography: it is initially on a scale with values ranging from 0 to around 900, and the land and ocean are not well split up because flat land ends up with nearly the same σ_o as the ocean, very close to 0. As such we normalise such that

$$\sigma_o = \begin{cases} 0 & \text{if LSM} < 0.5 \\ 1 + \log(1 + \sigma_o) & \text{if LSM} > 0.5 \end{cases} \tag{6}$$

Hence, we have a scale that goes from 0 to around 7, and that is > 1 on land and equals to 0 on ocean

2 Methodology Proposed

To try to tackle our sparse precipitation parameterization problem, we will make use of multiple techniques, a lot of them rooted in the broad field of Machine Learning (ML) Methods. We will therefore devote the first section to defining the general framework in which we will use these methods. In this section, we'll describe the datasets we'll be using, as well as the loss functions and metrics used in our neural networks, which form the general framework within which we'll be performing ML.

In a second section, we describe in more detail the type of neural networks used, which in our case will be variations on the classic Feed-Forward Neural Network (FFNN). These networks will first be used to clear the field, enabling us to quantify which variables are the most important for the problem (feature selection), using the metrics previously discussed. They will also be used throughout the study as our chosen 'common ground' to compare different inputs: if we want to assess the quality of two integration methods, we integrate with them a given variable and give it as input to the same FFNN architecture: the FFNN will act as a reflector of how much information we managed to keep with the integration.

In the third section, we will then look at the methods used to develop new integration schemes for 3D variables. Most of these methods will be based on the FFNN structure already developed, which we will use to train the weights of different integration kernels

Finally, in a fourth section, we'll look at how these methods lead to optimal use of Symbolic Regression (SR). We will describe the SR methods used, and the framework for training them, which is also similar to what we developed in the first section.

2.1 Framework, Metrics, Baselines

To perform a data-driven approach, we'll need the traditional building blocks of it: First, a dataset, with features and target, to train and validate our NN and Symbolic Regression. Second, metrics: in the form of a loss function to define objectives for our NN, and performance metrics, that will also us to benchmark all of our methods (NN-based or not) on a unified ground. All of our Neural Networks will use the `PyTorch` Python library [12].

2.1.1 Features and Target

To create our datasets, we take into inputs N_{3D} 3D variables each with N_p pressure levels, and N_{2D} 2D or static variables, and the total precipitation P. They are all spanning on a horizontal grid of $N_{lat} \times N_{lon}$ points, during N_t timesteps. As such, a dataset $\mathcal{D}$ will always mean in this work an ensemble of two matrices, a features matrix F composed of the ERA5

inputs other than precipitation, and a target matrix T composed of precipitation inputs, such that

$$\mathcal{D} = (F, T) \quad \text{with} \quad \begin{cases} F & \text{of size} \quad (N_{3D} \times N_p + N_{2D}, N_{lat} \times N_{lon} \times N_t) \\ T & \text{of size} \quad N_{lat} \times N_{lon} \times N_t \end{cases} \tag{7}$$

That can be considered two matrices of samples from the random variables $\boldsymbol{\phi}(\mathbf{x}, t) \in \mathbb{R}^{N_{3D} \times N_p + N_{2D}}$, and $P \in \mathbb{R}$. These datasets $\mathcal{D}$ will be the form used to train our Neural Network or Symbolic Regression equations and to evaluate the performance of different candidates $\mathcal{G}_\chi$ functions, precipitation parametrization. Our broad problem is then framed as follows : Find a simple semi-empirical, local in time and horizontal space, deterministic relationship between total ERA5 precipitation and grid-scale variables, such that

$$P(\mathbf{x}, t) = T = \mathcal{G}_{\boldsymbol{\chi}}(F(\mathbf{x}, t)) \tag{8}$$

Hence, the $\mathcal{G}_{\boldsymbol{\chi}}$ function that we are looking for is such that $\mathcal{G}_\chi : \mathbb{R}^{N_{3D} \times N_p + N_{2D}} \to \mathbb{R}$

2.1.2 Metrics

Mean Squared Error

Coefficient of Determination The main performance metric we will be using during this study is the coefficient of determination, R^2, which can be defined for our case as

$$\begin{aligned} R^2 &= 1 - \frac{\sum_{i=1}^{N_{lon} N_{lat} N_t} (T_i - \mathcal{G}_\chi(F_i))^2}{\sum_{i=1}^{N_{lon} N_{lat} N_t} (T_i - \bar{T})^2} \\ &= 1 - \frac{MSE}{\sigma_T^2} \end{aligned} \tag{9}$$

with T and F the target and features matrices defined earlier, and $\bar{T}$ denoting the mean of T. This measure is related to the Mean Squared Error (MSE), the Loss function that will be used in the majority of our NN throughout this work.

Onset description: Binary classification metrics While the R^2 metric describes the overall performance of our model, one can develop metrics to specifically assess the performance of certain regimes. For example, if we want to quantify how well our model describes the onset of precipitation. We can consider that a point T_i exceeds a certain threshold T_c, for example, $T_c = 0.3mm/h$, the corresponding column is precipitating at that time. We can frame our regression problem as a binary classification problem, being "Is it raining or not ?". The results of this problem can be summed up in a standard confusion matrix

		Predicted	
		$\mathcal{G}_\chi(F_i) > T_c$	$\mathcal{G}_\chi(F_i) \leq T_c$
Actual	$T_i > T_c$	TP	FN
	$T_i \leq T_c$	FP	TN

From there, FP,FN,TP and TN can be combined in various ways to produce metrics. We chose to go with the Mathew Correlation Coefficient, which can be written as

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{10}$$

The MCC provides a balanced measure of classification accuracy, particularly useful when dealing with imbalanced datasets like ours, with around 90 percent of non-raining samples. MCC considers both the sensitivity (true positive rate) and specificity (true negative rate) of the model, making it well-suited for evaluating the model's overall effectiveness in capturing precipitation events while minimizing false alarms, thus providing a robust assessment of its binary classification performance.

2.2 Neural Networks, Feature Selection and Kernel Discovery

One of the main focuses of this study, sparsity, means that want to reduce both the dimensionality of our features vector $\phi(\mathbf{x}, t)$ which equals to $N_{3D}N_p + N_{2D}$ and trainable parameters vector χ; all of that while keeping performance metrics as good as possible. To tackle the first issue, one can first select the most informative variables between all of our ERA5 inputs, therefore reducing N_{3D} and N_{2D}. To discriminate the "informativeness" of each variable, we will be comparing them using the same NN model, a

2.2.1 Neural Network baselines

Given a Batch Size N_b, an input size N_i and an output size N_o, we can describe the layers as follow

Linear Layer Given an input tensor $\mathbf{x_i}$ of size (N_i) and an output tensor $\mathbf{x_o}$ of size (N_o)a linear layer, denoted in this study Linear(N_i,N_o), will perform the following operation

$$\mathbf{x_o} = \mathbf{A}\mathbf{x_i} + \mathbf{b} \tag{11}$$

With $\mathbf{A}$ the weight matrix, of size (N_o, N_i), and $\mathbf{b}$ the bias vector of size (N_o). $\mathbf{A}$ and $\mathbf{b}$ are entirely composed of trainable parameters. The Linear layer is the main building block of the FFNN.

Batch Normalization layer We describe a Batch Normalization layer, denoted in this study BatchNorm(N_i), given an input tensor $\mathbf{x_i}$ of size (N_i) and an output normalized tensor $\tilde{\mathbf{x_i}}$ of size (N_i)

$$\tilde{\mathbf{x_i}} = \frac{\mathbf{x_i} - \mathbb{E}(\mathbf{x_i})}{\sqrt{\text{Var}(\mathbf{x_i}) + \epsilon}} * \boldsymbol{\gamma} + \boldsymbol{\beta} \tag{12}$$

With ϵ a small fixed parameter ($\epsilon = 10^{-6}$ in this study) to avoid division by zero, and $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$ trainable vectors of size N_i.The pointwise multiplication is denoted $*$. This layer normalizes the inputs to fixed means $\boldsymbol{\gamma}$ and standard deviation $\boldsymbol{\beta}$. This provides a faster convergence of our networks and better results [11].

Kernel Layer We describe a Kernel layer, denoted in this study KernelLayer(N_i,1),given an input tensor $\mathbf{x_i}$ of size (N_i) and an output tensor $\mathbf{x_o}$ of size (1) as the operation

$$\mathbf{x_o} = \mathbf{A}\mathbf{x_i} \tag{13}$$

With $\mathbf{A}$ the weight matrix, of size $(1, N_i)$, and no bias vector. Therefore, the N_i trainables parameters of $\mathbf{A}$ compose our integration kernel, so the layer act as an integration layer.

Functionnal Kernel Layer We describe a Functional Kernel layer, denoted in this study FuncKernelLayer((N_i,1),k_ϕ),given an input tensor $\mathbf{x_i}$ of size (N_i) and an output tensor $\mathbf{x_o}$ of size (1) as the operation

$$\mathbf{x_o} = k_\phi^{\boldsymbol{\eta}}(\boldsymbol{p}) \cdot \mathbf{x_i} \tag{14}$$

With $\mathbf{p}$ the pressure level vector of size (N_i), $k_\phi^{\boldsymbol{\eta}}$ a weight function depending on p and parametrized by the trainable parameters of the vector $\boldsymbol{\eta}$. $\cdot$ denotes the scalar product. As such, we perform with this layer an integration of the 3D input variables $\mathbf{x_i}$, but the trainable parameters act as parameters in a function that then outputs a weight vector $k_\phi^{\boldsymbol{\eta}}(\boldsymbol{p})$, rather than being directly themselves the vector.

Activation functions We use two activation functions. Given an input tensor $\mathbf{x_i}$ of size (N_i), we use the Rectified Linear Unit (ReLU) function, written as

$$\mathrm{ReLU}(\mathbf{x_i}) = \frac{\mathbf{x_i} + |\mathbf{x_i}|}{2} \tag{15}$$

With $|\cdot|$ the absolute values of the elements of the vector. As such, every output of ReLU is non-negative. We use this activation function in the end of all our network : since our regression target, precipitation, is positive by definition, we enforce this physical constraint to be respected with this activation function.

We also use the Gaussian-error Linear unit (GeLU), written as

$$\mathrm{GeLU}(\mathbf{x_i}) = \mathbf{x_i} * \Phi(\mathbf{x_i}) \tag{16}$$

With $\Phi(x)$ the cumulative distribution function of a normal distribution $\mathcal{N}(0, 1)$. This activation function is a smoothed version of the ReLU activation function. One of its important features is that it does have a small non-zero output for negative inputs, rather than just 0 for ReLU. It prevents the "death" of certain neurons, meaning that a neuron that outputs a lot of negative values could be never activated, hence his weight is never modified by the backpropagation, effectively "killing it", which negatively impacts the NN performances by reducing his number of effectively working weights.

2.3 Kernel Learning

To reduce further the dimensionality of our inputs, one way that comes to mind is to integrate our 3D inputs, transforming a N discrete values on the vertical into one. The

naive approach could be to just perform a column integral, written for

$$\begin{aligned}\bar{\phi} &= \int_0^{p_0} \phi(p)dp \\ &\sim \sum_{i=1}^{N} \phi(p_i)\Delta p_i\end{aligned} \tag{17}$$

With $\Delta p_i = (p_{i+1} - p_i)$, $p_{N+1} = 0$. However, in the same manner as with feature selection, we want to retain as much information from this integral as we can. One can think that certain atmospheric portions are more important to convective processes than others and that these portions are different for each state variable: we could typically suppose that the relative humidity is really important relative to clouds in the lower troposphere, where as the ice water content is typically correlated with precipitation at higher altitudes. In this manner, we can then rewrite the last integral associated with a weighting function $k_\phi(p)$ such that

$$\begin{aligned}\hat{\phi} &= \int_0^{p_0} k_\phi(p)\phi(p)dp \\ &\sim \sum_{i=1}^{N} k_\phi(p_i)\phi(p_i)\Delta p_i\end{aligned} \tag{18}$$

With k_ϕ being a deterministic function. The problem is then rephrased as such that we need to find k_ϕ such that R^2 is maximized when using $\hat{\phi}$ as a predictor. There are several ways to tackle such a problem an we are going to go through them in the next paragraphs

First off, it is clear that our kernels k_ϕ depend on a certain number of trainable parameters, that are supposed to be fitted when we resolve the problem. While we reduce the dimension of our variables vector $\boldsymbol{\phi}$, we increase the dimension of our trainable parameters vector $\boldsymbol{\chi}$, and we want this increase to be ideally small. Since we are performing a discrete integration during our computation, one can want to search for N trainable parameters each corresponding to the N values taken by $k_\phi(p_{i=1,..,N})$. There are multiple ways to find such a kernel. We opted for two different way: Canonical Correlation Analysis and learning with a Kernel Layer.

On a second time, we can go further and seek to reduce the number of trainable parameters of the kernel in itself. We chose to develop a kernel based on a family function depending on a few parameters that will be trained with an FFNN and the Functional Kernel Layer described earlier.

2.3.1 Canonical Correlation Analysis

Given a state variable defined on N pressure levels $\phi_{1,..,N}$, and the corresponding precipitation P, we can treat each sample these pair in our data as sampled from two column vectors of random variables, with finite second moments. Canonical Correlation Analysis (CCA) seeks an optimal vector k_ϕ to perform a linear combination of the random variables in the vector ϕ into a single scalar, that we can write in the form $k_\phi^T \cdot \phi$, (with $\cdot$ being the

scalar product). The objective is then defined as maximizing the correlation between the precipitation and the linear combination of the state variable vector, i.e.

$$k_\phi = \underset{k'_\phi}{\operatorname{argmax}} \operatorname{corr}\left(k_\phi'^T \cdot \phi, P\right) \tag{19}$$

With $\operatorname{corr}\left(k_\phi'^T \cdot \phi, P\right)$ representing the Pearson's correlation coefficient ρ, such that for two independent random variables X,Y with second finite moments,

$$\rho_{X,Y} = \operatorname{corr}(\mathrm{X}, \mathrm{Y}) = \frac{\mathrm{E}(XY) - \mathrm{E}(X)\mathrm{E}(Y)}{\sigma_X \sigma_Y} \tag{20}$$

. With $\sigma_X = \sqrt{\mathrm{E}\left(X^2\right) - \mathrm{E}(X)^2}$ the standard deviation of the random variable X. In our discrete case, we would compute the sample correlation using the matrices T and F such that

$$\rho = \sum_{i=1}^{N_{lon}N_{lat}N_t} \hat{T}_i P_i - \sum_{i=1}^{N_{lon}N_{lat}N_t} \hat{T}_i \sum_{i=1}^{N_{lon}N_{lat}N_t} P_i \tag{21}$$

Where $\hat{T}_i = \sum_{j=1}^{N_p} (T)_{ij} (k_\phi)_j$We can then numerically compute our optimal k_ϕ and we obtain a kernel for each 3D variable a kernel k_ϕ with N trainable weights parameters, one for each pressure level.

However, it's important to recognize the potential limitations of relying solely on linear correlation when applying CCA. The intricate dynamics of tropical precipitation are often characterized by complex non-linear interactions between subgrid scale variables and precipitation patterns. Linear correlation measures might fail to capture these intricate non-linear associations, leading to an incomplete understanding of the underlying processes. By exclusively relying on linear correlations, there is a risk of overlooking significant relationships that may be better captured by more sophisticated techniques, such as NN-based ones which can unveil complex non-linear relationships. To tackle CCA's shortcomings, we introduce in the next paragraph a NN-based kernel discovery method.

2.3.2 Neural Network Kernel Layer

Another way to discover kernels with one trainable parameter per pressure level for each variable is to use Neural Networks, quite similar to the one proposed for feature selection. We are taking the same network, but adding a first neuron layer without biases, ensuring we got a multiplicative kernel with N parameters like in our definition. As such we're optimizing for an objective (lowest MSE) by twisting the weights of the multiplicative kernel. Meaning that the single input on the second layer is the most informative combination of all the pressure levels with respect to this neural Network architecture.

2.3.3 Function Kernels

If we want to reduce the dimension of our parameters $\boldsymbol{\chi}$, we can reduce the size of the aforementioned kernels. Indeed N parameters for single pressure level variables seem high for empirical baselines which are supposed to be easily readable and reproducible.

In this manner, we need a way to parametrize our kernels $k_\phi(p)$ with as few parameters as possible. One way to attain this objective is to assign to $k_\phi(p)$ a fixed function form, for example, $k_\phi(p) = ap + b$, with $a, b \in \mathbb{R}$ being trainable parameters. We then reduced for each variable the dimension of the parameters from N to 2. We could think of a wide range of functional families that could be suited for these kernels. In this study, we tried Heaviside and Ramp functions, polynomials of varying orders, sine and cosines, and sigmoids, and ended up using a 3-parameters function composed of a constant weight and a Gaussian function, expressed as

$$k_\phi^{\alpha,\sigma,\mu}(p) = \alpha + \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right) \tag{22}$$

With $\alpha, \mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^{+*}$. This functional form allows us to accentuate (in terms of weighting) different atmospheric layers, centered around specific pressure levels controlled by the parameter μ and with a specific height, controlled by σ. The α parameter allows us to still have smaller but non-zero weights outside of the region of interest.

2.4 Symbolic Regression

After reducing the number of features and integrating the pressure-level variables, we end up with a few, typically under 6, distinct scalar inputs to feed into Symbolic Regression (SR). All the techniques we developed earlier sought to allow the use of SR, which does not perform well with a number of variables bigger than ~ 10. Unlike traditional regression methods that focus on numerical coefficients, Symbolic Regression aims to identify mathematical formulas that best represent the interactions between input variables. No predefined model serves as a starting point for symbolic regression; instead, initial expressions emerge through random combinations of mathematical components like operators, functions, constants, and variables, that are part of subsets that we can specify in the start.

The strength of symbolic regression lies in its potential to uncover intrinsic dataset relationships, untethered by human assumptions or gaps in domain knowledge. This promotes interpretability, enabling insights into the data-generating system, enhancing generalizability, and mitigating overfitting tendencies.

After fitting the training datasets, SR (using the library `PySR`) outputs a family of growing complexity equations: Balancing accuracy and simplicity between these equations will be done using a complexity-accuracy Pareto Plan, resulting in solutions along a Pareto front: the front where it's impossible to increase accuracy without increasing the complexity, and vice versa .

3 Results

Now that we have precisely described the methods we are going to use, we present and discuss here the results obtained with it. We will first see the methodology adopted to train the neural networks and the associated technical details: we will briefly describe the structure of the different networks used, the hyperparameters used for their training as well as cross-validation methods.

We will then see a simple example of parameterization by neural networks using only relative humidity as a predictor: This will allow the reader to better understand the physical meaning of the different metrics used, and the following results. Then, we will see which variables appear to have the most predictive power with our FFNN architecture and will comment on these results.

In the third part, it will be necessary to analyze the results obtained by the different methods of "Kernel Learning", as well as to quickly analyze the meaning of the Kernels obtained. This will finally allow us to approach the results of Symbolic Regression, where we will present several candidate equations and study their different terms and meanings. We will conclude the study by summarizing all the results obtained in Pareto precision-complexity plans, which will allow us to draw a Pareto frontier and thus to have a global vision of the work accomplished and to come

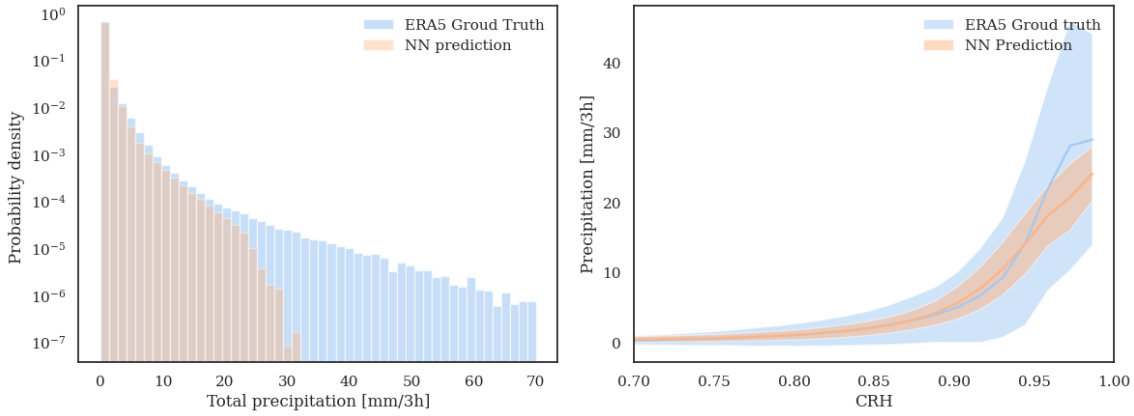
3.1 Methodology and Training procedures

The first part of our study was devoted to developing a neural network architecture

Parameters	**Main Network**	**Kernel Layer**
Layer 1	-	KernelLayer(N,1)
Layer 2	Linear(N,256),GeLU()	Linear(1,256),GeLU()
Layer 3	BatchNorm(256),Linear(256,128),GeLU()	Same as Main Network
Layer 4	BatchNorm(128),Linear(128,64),GeLU()	Same
Layer 5	BatchNorm(64),Linear(64,64),GeLU()	Same
Layer 6	BatchNorm(64),Linear(64,1),RELu()	Same
Optimizer	Adam	Same
Learning rate	10^{-4}	Same
Batch size	1024	Same
Epochs	5	Same

To train these models we ran some tests on the optimal number of samples to use. A single time step of our ERA5 data gives us $1440 * 200 = 288000$ sample, and we dispose of $365 * 8 = 2920$ time steps per year, and multiple years. This means we have quite a lot of data, in fact a largely sufficient amount for the small, sparse model that we are trying to train. Our NN models have a number of trainables parameters typically of order $\mathcal{O}(10^4)$, while deep-learning forecasting models using $3D$ inputs might have around $(O)(10^8)$ (e.g. [4]), therefore needing dozen of ERA5 complete years to train. In our case, we found it sufficient to use around 15 time steps on the training sets, leading to around 5.10^6 samples. This number of samples allows us to reach a plateau of minimum loss, while being sufficiently big to prevent overfitting, all of that with a training times of around 5 minutes with a single 24-core AMD Epyc2 7402 CPU.

We perform an early stopping procedure, meaning that we stop the training when the validation loss starts to rise, and we keep the model associated with the best validation loss.



(a) (b)

Figure 1: (a): Univariate distribution density of ERA5 precipitation and the associated RH-only NN-based predicted precipitation, log scale on the y- axis. (b) : ERA5 precipitation and corresponding RH-Only NN-based predicted precipitation, conditional on CRH. The line represents the mean on each CRH bins (every 0.01 CRH), and the top and bottom of the shadow respectively the 84th and 16th quantile of each bin.

k-folds-like cross-validation We will use in this study cross-validation methods that resemble the k-fold method, with $k = 3$ in our case, but with minor modifications. We divide the training dataset and the validation datasets in 3 blocks, which all have the same number of samples. We then train a first network with the training blocks 1-2, and the validations blocks 1-2 ; a second network with training and validations blocks 2-3, and a third with blocks 1-3. Therefore we have trained three networks on different datasets, giving us a minimal, maximal, and mean performance, allowing us to quantify the variance in terms of accuracy of our networks, which we will express in the form $R^2 = \text{mean} \pm(\text{max} - \text{mean})$.

3.2 RH-based NN example

Using R^2 as a unified measure to quantify informativeness between models may be useful and necessary. However, a global R^2 does not really allow one to get a grasp on the difference in forecast quality between different geographies (e.g ocean and land), or precipitation regimes. We train a model of the 'Main Network type', using as input the full column of Relative humidity, so an input size of 21. We follow the training procedures described in the previous section. The R^2 obtained by this simple network is $R^2 = 0.43 \pm 0.02$. We present on figure 1 the univariate distribution of ERA5 precipitation on the left and its conditional distribution with Column Relative Humidity (CRH) on the right. We plot it against this same distribution but with the NN-predicted precipitation, to get a clearer view of what really lies behind "$R^2 = 0.43 \pm 0.02$". The Column Relative Humidity humidity is defined as

$$CRH = \frac{CWV}{CWV_s} = \frac{\frac{1}{g}\int_0^{p_0} q dp}{\frac{1}{g}\int_0^{p_0} q_s dp} \tag{23}$$

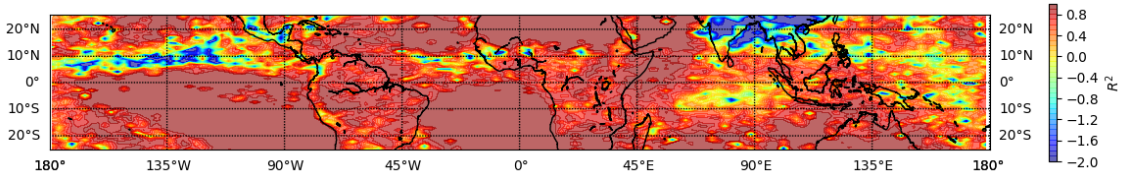


Figure 2: R^2 world map on the tropical latitudes and full longitudes, results obtained by the RH-only NN-based predicted precipitation model. R^2 is computed on each grid points over 100 samples in time.

and serve as a way to highlight different precipitation regimes: as seen on 1, the mean precipitation scale exponentially with CRH. As we see on the precipitation univariate plot on the left, our RH-based NN captures quite well the distribution for low to moderate raining events, typically under 20 mm/3h, but misses out completely the more extreme events, typically with precipitation over 30mm/3h. This underrepresentation of extreme events is a corollary of an overrepresentation of the really low-rain events. It is specifically visible in the second bin (1 to 2 mm/3h, on 1), with the overrepresentation on this single bin compensating for the underrepresentation in every subsequent bin, due to the really high number of low-precipitating bins. This a typical manifestation of the so-called drizzle effect: our model is raining 'too ofen, and too few'. We have too many precipitating bins, and thus they are precipitating an abnormally small amount of rain.

On the same figure, 1(b), we represent the true and predicted precipitation distribution, conditional on CRH. We can observe that the conditional mean of precipitation is predicted quite well by our NN for moderate to high rain events (below 20mm/3hr), but the variance is not captured as well. Our NN tends to stick to values near the mean, another predictable effect from the fact that we are using a MSE loss function In figure 2, we represent the R^2 on every grid point of our studies, obtained over 100 samples (i.e 100 times point) for every grid point. As such, we obtain a geographical breakdown of our NN parametrization quality. As discussed earlier, we see that the problem is much easier on oceans. However, we see a lot of huge drops in R^2 on land, typically on the African and South America East Coasts. This drop is also worsened in areas with varying topography, such as the Andes. This indicated that a model relying solely on thermodynamical variables fails to generalize on our case from land to ocean. We will introduce further in this study orographical variables that will help with a better generalization

With the above discussion, we have in mind what typically lies behind a specific averaged score R^2 in terms of differences of geography or precipitation regimes. In the following paragraphs which will be presenting our experiments and results regarding the choice in variables and integration structures, we will be comparing their performance solely based on their R^2 scores.

3.3 Feature Selection

As proposed we start by shortening the number of input variables for our by comparing their performances using the a satndard feed forward neural Network. We build one network for each variables as input, meaning that the 2D variables NN will have an input size of 1 and the 3D ones an input size of 21. One could argue that comparing them one by one could bias our scale of informativeness for some variables, given that some might not be informative on their own but only when coupled with others. Howewer it would have been too computationnaly expensive to train networks on every combination.

On fig. 5 are provided the R^2 scores obtained by our 'Main Network' using single 3D variables with 5 differents methods. To have a first glimpse at how much information each variable carrys, we made tests with the Full Vertical Profile, mass-weighted or not. Here, 'mass-weighted' means to that we multiply each inputs on a pressure level p_i by the corresponding δp_i. This gives a rough weighting to the network and gives it information on pressure levels repartition. Altough one could suppose that a Neural Network could recreate these weights, given the number of trainable parameters (40.10^3) it has, adding mass-weighting proved to increase our overall R^2 performance on nearly all variables.

We see that solely temperature based variables, θ,T and DSE appears to have quite low predictive power on their own, around $R^2 = 0.2$ with full levels. The wind speed magnitude appears to carry nearly no information on it's own, with R^2 close to 0. Variables that concatenate temperature and humidity tends to carry the most information : θ_e^+,RH are the best predictive variables with R^2 around 0.45, while MSE and θ_e are very close at 0.35. The ice and liquid water content, q_i and q_l, also carry a lot of information (R^2 respectively of 0.37 and 0.45) but their final use in our parametrizations will be debated later in this study. Overall, our best candidates for a sparse analytical realtionship seems to be variable to includes humididty and temperature in one mesure : RH and θ_e^+. We will investigate these first results further to see if high scoring variables here convert into high scores with kernel integration.

3.4 Kernel Learning

3.4.1 Canonical Correlation Analysis

To start our kernel finding journey, we first put in place a framework utilizing CCA. As described earlier, we first find a kernel using the CCA methods. Once we found this kernel, we run the CCA-Integrated variables trough 'main network' FFNN and compare their score to a regular integral with no weights, to see if we manage to scavenge more informativennes from the full levels with our CCA than with a regular integration. We see that the quality of the CCA is quite ambivalent: for the three variables shown in fig. 3, θ_e^+, RH and q_l we see some really noisy kernels; if we compare their scores to the results provided in fig. 5, we see a small increase in R^2 on q_l, and a worse performance on θ_e^+, RH compared to a regular mass weighted integral. The limitations of the CCA, relying on linear correlation analysis, already highlighted in the methodology description lead us to non-linear methods of full kernel discovery.

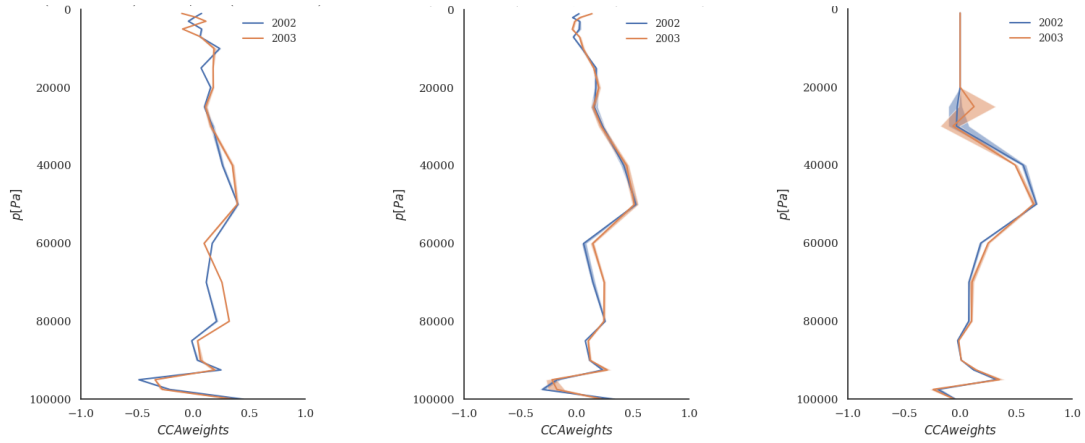


Figure 3: Kernels obtained with the CCA method, for (a) : θ_e^+, $R^2 = 0.31 \pm 0.01$ (b) : RH, $R^2 = 0.29 \pm 0.02$ and (c) q_l, $R^2 = 0.21 \pm 0.01$. Parameters trained on the 100 to 1 hPa levels. The full line represent the mean weights on the 3 folds, and the pale filled zone borders the max and min. The two differents lines on each graph represent results on two different years, 2002 and 2003

3.4.2 Kernel Layer

Using the Full Kernel Layer method discussed in the Methodology section, we compute the kernels for twelve 3D variables, presented in fig. 4. We use the 'Kernel Layer' NN architecture to discover the weights of the kernel, and then give as input the integrated 3D variables with the the weights we found to a 'Main Network' NN architecture, giving us a R^2 score, allowing us to quantify how much information we managed to keep with this integration scheme.
On fig. 5, we resume the results obtained so far in terms of R^2 and compare the Kernel Layer integration, standard integration and full-level inputs. The obtained results with Kernel Layers surpass those achieved through standard integration, offering clear advantages. Moreover, we managed to obtain R^2 comparable to the full 21 levels, indicating that we are on a promising path. However, as seen in fig. 4 it is evident that certain noisy kernels are associated with less informative variables, particularly those related to temperature. This noise issue is further compounded by the complexity of the model, which involves 21 parameters. The model's lack of readability and interpretability is notable, as the weights appear scattered and noisy, impairing our ability to discern meaningful patterns. To address these shortcomings, there is a pressing need for a refined version that strikes a balance between preserving prediction quality and smoothing out functions, ultimately reducing the number of parameters while enhancing the model's usability.

3.4.3 Function Kernels

As we've seen, the use of a NN-based Kernel layers provides ways to retain more information than a non-weighted integration. However, we are still adding a lot of trainable parameters, and these kernels are not easily expressed nor interpretable, with spurious jumps in weight values for example. As discussed earlier, we will now try to learn kernel using a kernel family function with 3 parameters. We represent on fig 6 the same 12

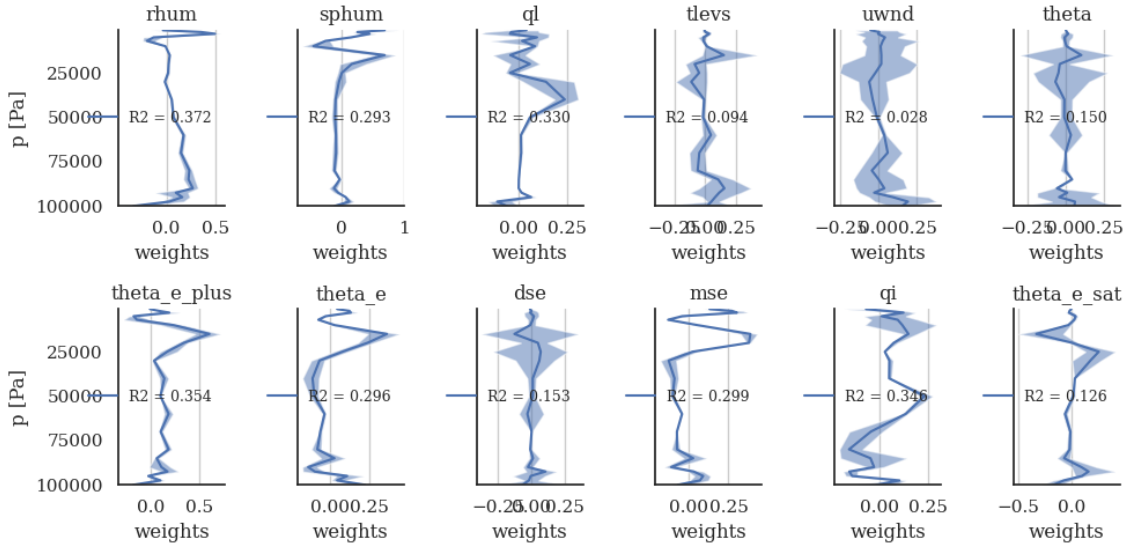


Figure 4: Kernels obtained with the kernel layer methods, with parameters trained on the 1000 to 10 hPa (full levels). The R^2 score is obtained by a 'Main Network'-type NN wich only use as an input feature the variable integrated with the displayed kernels, and this kernel is fixed, i.e found and optimized using another network

kernels as earlier. We chose to restrict the pressure levels to under 200 hPa because it actually makes the learning more stable. If we take the full levels, the functions are not as smooth and do not converge as well, sometimes making strangely huge peaks on the levels above 200 hPa. Since it could be labeled spurious that these levels are good causal predictors of convection, we discard them.

We see that the resultant R^2 is really comparable with the full kernel learning of the networks. That is reassuring since it proves that we managed to lower the parameter's space dimension without compromising much. The decision to use 3D variables on pressure levels below 500 hPa from ERA5 data is a prudent one, rooted in the avoidance of causality problems. Atmospheric variables at higher pressure levels, typically above 500 hPa, can indeed exhibit causality challenges when attempting to relate them to precipitation processes. This is due to the fact that variables at higher altitudes are often influenced by, rather than causing, precipitation and convection. For example, the tropopause, the boundary between the troposphere and the stratosphere, typically occurs at varying altitudes but is often well above 500 hPa. Processes occurring near the tropopause are driven by complex interactions between the stratosphere and troposphere and are not directly tied to surface-level weather events like precipitation. [2] Therefore, variables at or above the tropopause level are not suitable causal indicators for precipitation.

By focusing on pressure levels below 500 hPa, we mitigate the risk of including variables that are more likely to be influenced by precipitation and convection rather than causing them. This approach ensures that the selected variables are more likely to capture the causal relationships between atmospheric conditions and precipitation, leading to more accurate and physically meaningful subgrid parameterization models.

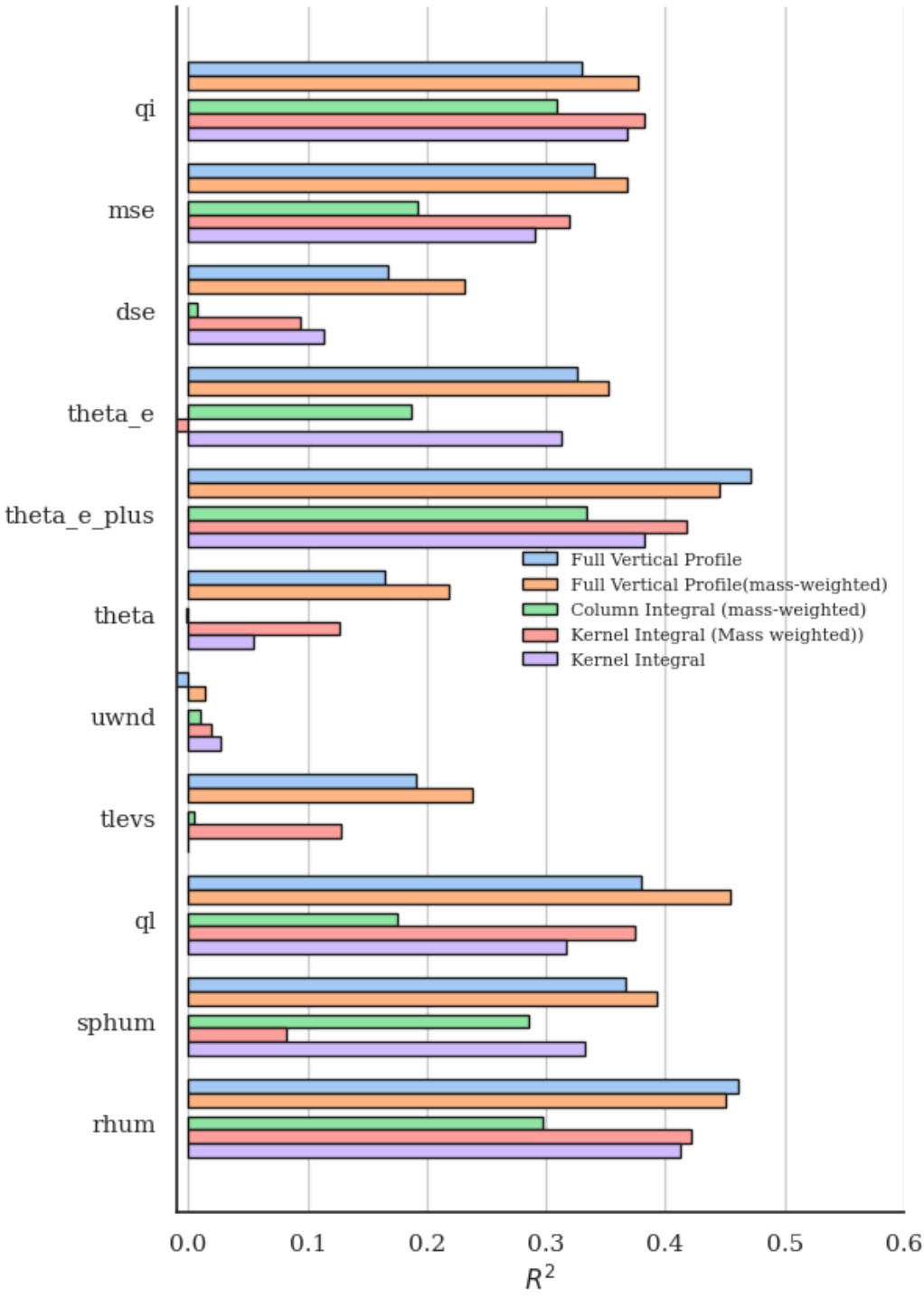


Figure 5: R^2 scores obtained using different input variables for the same NN architecture, 'Main Network'. For each variable, we use 5 different types of inputs. Two of them are composed of the full pressure levels column (21 values), Full Vertical Profile, and Full vertical Profile (mass-weighted). The 3 others are composed of 1 value, being the result of different integration methods of the 21 values of the Column. One is the Column Integral Mass weighted (eq. 17), and the two others are Kernel Integrals (eq. 18), mass-weighted or not.

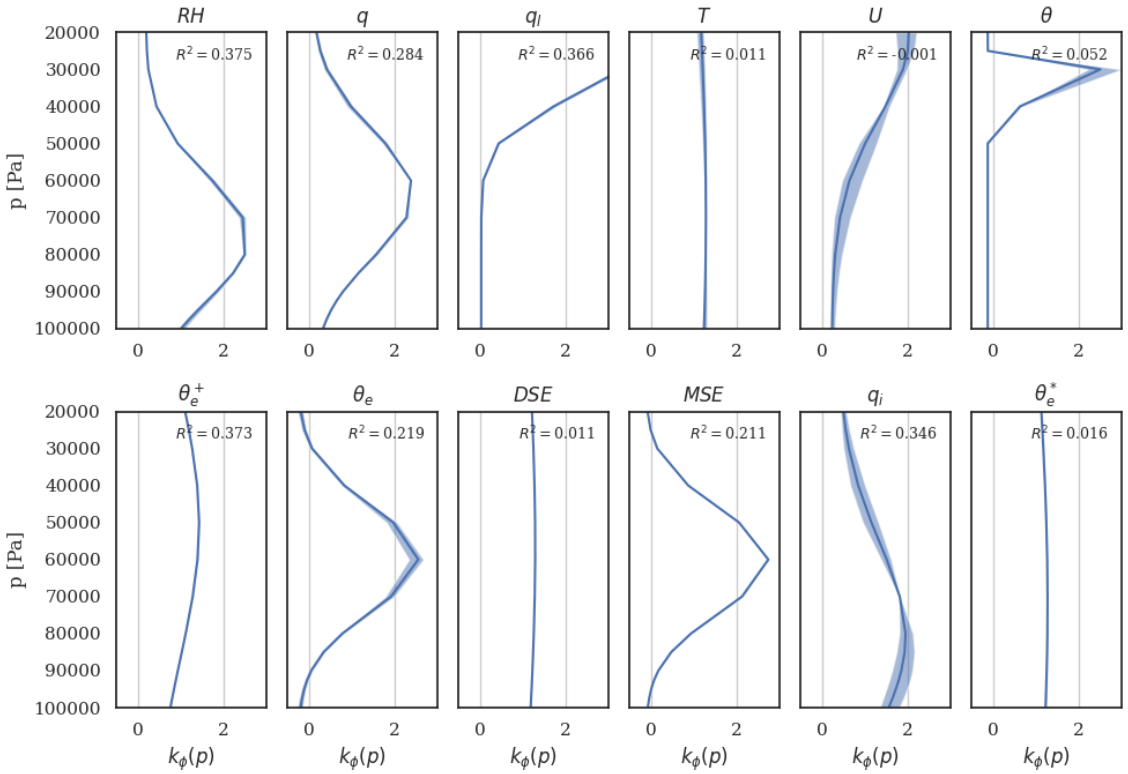


Figure 6: Kernels obtained with the functional kernel layer methods, with parameters trained on the 1000 to 200 hPa levels. The R^2 score is obtained by a 'Main Network'-type NN which only uses as an input feature the variable integrated with the displayed kernels, and this kernel being fixed (because it was found and optimized using another network)

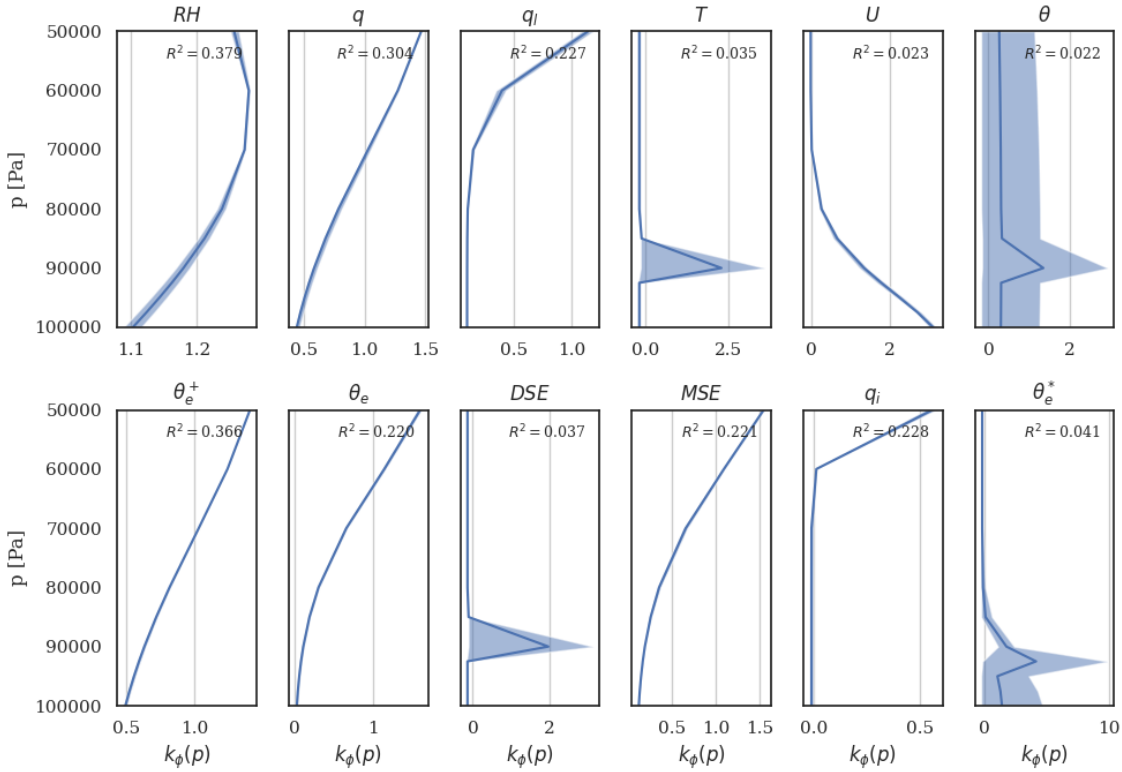


Figure 7: Kernels obtained with the functional kernel layer methods, with parameters trained on the 1000 to 500 hPa levels. The R^2 score is obtained by a 'Main Network'-type NN which only uses as an input feature the variable integrated with the displayed kernels, and this kernel being fixed (because it was found and optimized using another network)

We see that these new kernels provide nearly similar results to their 200 hPa counterparts for the most informative humidity-based variables RH,q,θ_e^+. However, we see way worse results for q_i and q_l. Indeed, the information carried by these variables seems to be concentrated above the troposphere (500hPa) looking at the 1000-200 hPa kernels (fig. 6). However, this is not that damaging to our research since we are trying to privilege prognostic variables, i.e. $T/q/U$ based variables.

3.5 Symbolic Regression

The `PySR` library that we're using provides a framework allowing us to manually tune a variety of hyperparameters. Here, we are using a custom loss function, a slightly modified version of MSE to enforce the positive precipitation physical constraint, expressed as

$$\mathcal{L}(\mathcal{G}_\chi(F_i), T_i) = (\mathrm{ReLU}(\mathcal{G}_\chi(F_i)) - T_i)^2 \tag{24}$$

With F and T being as always the features and target matrices. We provide the `PySR` frameworks with a few operators: here, the main ones will be $\exp(), \sin(), \cos(), \log(), \mathrm{RElu}(), \tan()$. And we assign a specific complexity score to them. Each equation will have a higher complexity score the higher the number of operators, constants, and variables there. Our framework finds equations of increasing complexity, and the basic complexity quantification in `PySR` is not suited for interpretable or readable equations.

Typically, since we aim to define a "readable" functional form, we need to avoid formulations like $\sin\arctan(x)$, which are almost never used in any analytical formulations of physical laws or processes. However, from a computing point of view, $\sin\arctan(x)$ is not that "complex". There is a kind of gap between what us humans might consider a complex equation and what might be actually a complex computation. Using a 2 variables formulation, e.g $f(x, y) = \alpha x + \beta y$ might seem much less complex than $f(x) = e^{\alpha x} - \sin(\arctan((x))$, however the second equation as a lower dimensionality, so a lower variance when training, etc. and is as straightforward to compute as the first ones on a modern computer. In this manner, we assign a low complexity of 1 to each constant and '+,*' operators in the equations, a slightly higher complexity, 2, for variables, and the '/' and 'power' operators. And finally, a complexity of 4 for each analytic function is described at the start of the paragraph ($\sin(), \exp()$...). After obtaining a series of equations, we validate their performance on a validation set that we fold 3 times using the same 3-fold methods as before, to obtain an estimation of its variance.

3.5.1 200 - 1000 hPa kernels

To develop our first family of equations, we will use RH,q_i,q_l,σ_o,θ_e,θ_e^+,θ_e^* as input variables. We will all integrate them with the 20-100 hPa Gaussian function kernels we developed earlier, and will add to RH,θ_e,θ_e^+,θ_e^* a normalization in the form of (for example for RH).

$$\bar{RH} = \frac{1}{D} \int_{2.10^4}^{10^5} k_{RH}^{\alpha,\mu,\sigma}(p_i) RH(p_i) \Delta p_i \tag{25}$$

With D a normalization coefficient such that $D = 10^5 Pa$, the same for the four variables. This transforms our integral to a weighted mean along the column and adds a parameter, but more importantly, it normalizes the values of RH and θ's based variables to values around $\mathcal{O}(10^1)$. This helps the Symbolic Regression to find good equations because a great disparity in the values of the different input variables makes it harder for ML-based models to converge to a satisfying solution. In all the following equations, variables are kernel integrated and normalized with D for RH,θ_e,θ_e^+,θ_e^*, but we dropped the overbar for the sake of readability. After running our Symbolic Regression model, we ended up with a family of increasing complexity functions, which will be summarized in the complexity-performance Pareto Plan in the next paragraph. To get a first grasp of the type of equations we obtained, we present one Pareto optimal equations (i.e you can't lower its complexity without lowering R^2) using RH, q_i, and q_l, and expressed as

$$P = \mathrm{ReLU}(0.415q_i + q_l + 10^{-4}e^{8.96RH}) \tag{26}$$

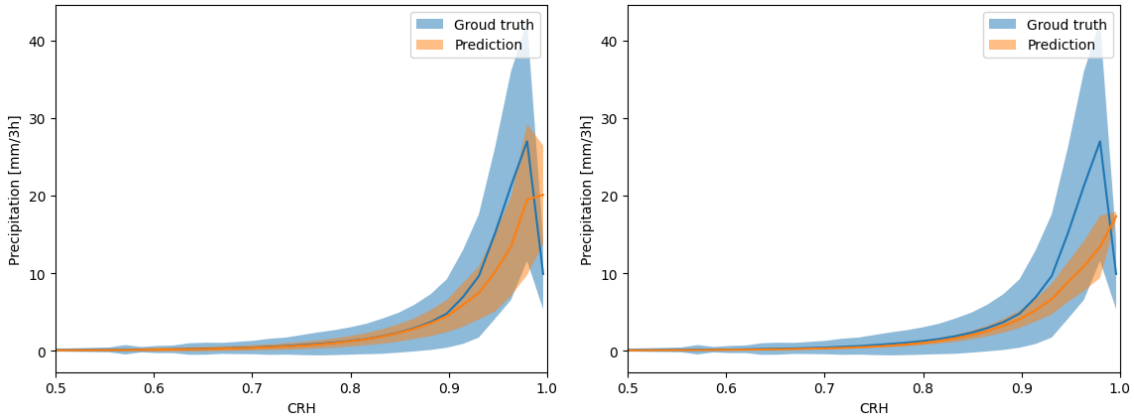
Here, we are using dimensional inputs, since q_i,q_l are expressed in $Pakg.kg^{-1}$, due to the multiplication by δp in the integral and the fact unlike RH they are not normalized by a dimensional parameter. As such, this equation would need to include for example a multiplying factor in $mm.h^{-1}$ and a dividing factor in Pa for it to be dimensional. However, the role of symbolic Regression is to potentially unravel new non-linear functional forms of precipitation parametrization; we can easily twist these forms *a posteriori* to make them obey at least dimensionality. The model's predicted precipitation conditional distribution on CRH is displayed on fig. 8 (a). We obtained with this model $R^2 = 0.48 \pm 0.02$, meaning that we outperform our 'Main Network' NN using the full 21 levels of Relative Humidity and tens of thousands of parameters, which obtained $R^2 = 0.43 \pm 0.02$. Considering that the equation obtained does not use especially complex operators and intricate mathematical formulation, relying only on exponential and simple additions, it's a good proof of concept for the parametrization quality of Symbolic Regression. However, we obtain this performance through the use of possibly non-causal pressure levels above 50 hPa that we discussed earlier, and non-prognostic (diagnostic) variables q_i and q_l. These results serve more as baselines to compare them to causal and prognostic models.

3.5.2 500 - 1000 hPa kernels

We are using in this section prognostic input variables namely RH,σ_o,θ_e,θ_e^+,θ_e^*, with the same normalization and training procedures as the last section with 200-1000 hPa ; but we are using 50-100 hPa pressure levels with their associated kernels displayed on fig. 7. We obtain another family of increasing complexity equations, that we will sum up in our next section about the Pareto plan, but right now we will focus on one equation using only RH and σ_o

$$P = \mathrm{ReLU}((1.16RH)^{8.21RH} e^{(-\sigma_o/(11.2RH+\sigma_o)))} \tag{27}$$

Here, we are using non-dimensional inputs only. As such, this equation would need to include for example a multiplying factor in $mm.h^{-1}$ for it to be dimensional.We displayed the results obtained by this model in fig. 8 (b). We obtained $R^2 = 0.38 \pm 0.01$. As we can see, this model is noticeably worse than the one displayed on 8 (b). The mean is not as well captured for $CRH > 0.9$, and more importantly the variance of the ground truth



(a) (b)

Figure 8: ERA5 precipitation and corresponding predicted precipitation, conditional on CRH. The line represents the mean on each CRH bin (every 0.01 CRH), and the top and bottom of the shadow respectively the 84th and 16th quantile of each bin. (a) : Results obtained with the model using 20-100 hPa, 26. $R^2 = 0.48 \pm 0.02$. (b) : Results obtained with the model using 50-100 hPa, 27. $R^2 = 0.38 \pm 0.01$

distribution is badly reproduced into account compared to (a). However, we still perform as well as 'Main Network' NN taking as input the Kernel Integrated RH, which is the input we are using in this equation (with σ_o). So our goal is still reached, since this model is Pareto-Optimal compared to a NN based one, using only 7 trainable parameters : 3 for the RH kernel, 1 for the normalization terms, and 3 parameters in the RH equation. The 'Main Network' with a kernel integrated RH has 42.10^3 trainable parameters. In the final section, we will show 3 different Pareto-Optimal equations with these Kernels, using prognostic variables that outperform this one.
An improvement over the above discussed could be to try to split the problem in two: on one hand, a binary classification problem trained to develop a functional form f_O that predicts the onset of rain, i.e. 'Is it raining or not'. This functional form would be a Heaviside function. On the other hand, we could train a functional form f_P that only accounts for precipitating bins only. This would mean that it would only be trained on precipitating bins, alleviating dataset size problems in symbolic Regression ($\sim$ 5000 samples). However, we implemented this method which seem to be promising but did not manage to develop models performing as well as the all-inclusive models.

3.6 Pareto Plan

the Pareto Plan presents a compelling case for the utility of Symbolic Regression. Although Symbolic Regression exhibits a slightly lower accuracy compared to Neural Network (NN), as we've seen with the two models cited in exemple above, its advantage lies in its capacity for parsimonious model representation. The significance of this resides in two key considerations.

Firstly, the simplicity of the Symbolic Regression models, as reflected by their lower parameter count, lends itself to increased interpretability. In atmospheric science, un-

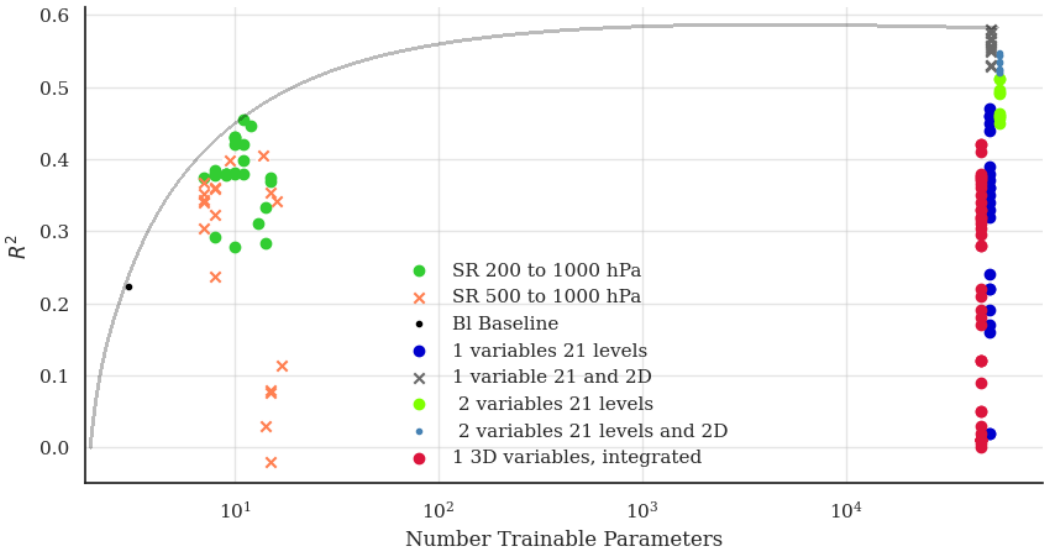


Figure 9: Complexity-Performance Pareto Plane, here Number of trainable parameters - R^2. The Pareto Frontier is the continuous line in pale gray. The points labeled 'SR 200 to 1000 hPa' and '500 to 1000 hPa' represents the family of functions described in section 3.5.1 and 3.5.2, respectively. The 'B_L Baseline' represents the AN18 model (eq.)described at the start of our study. All the other groups are computed with the 'Main Network' NN type, with different inputs. '1 variable 21 levels' represent 21 inputs values corresponding to each pressure levels of a single variable. '2 variable 21 levels' represent 42 inputs, the full pressure levels of two different variables. '1 variable 21 and 2D' represent 22 input values, the full pressure levels of a single 3D variable and a single 2D variable e.g orography or enthalpy fluxes. '2 variables 21 levels and 2D' represent 43 inputs in the same manner. '1 3D variables integrated' represent a single input being the result a Function Kernel Integration of the 3D variables.

derstanding the physical processes governing precipitation is of paramount importance. Symbolic Regression generates analytical expressions that directly relate model input variables to precipitation, thus providing valuable insights into the underlying physical mechanisms. The parameters used in the formulations need to be dimensionnal, therefore can be linked to certain characteristic dimensions of our problem; given that we are using the same SR model, we can train these few parameters on a different part of the globe, use these parameters to be informed on certain characteristics dimensions of the precipitating systems in each part were we trained them.

Secondly, the reduced model complexity afforded by Symbolic Regression aligns with the principle of Occam's razor, favoring simpler models when performance differences are not substantial. In cases where accuracy differences between Symbolic Regression and NN models are marginal, the preference for simpler models is not only computationally efficient but also aligns with the scientific principle of preferring simpler explanations when they are equally effective. Moreover, models with fewer parameters tend to generalize better to out-of-distribution or unseen data (Parsimony Rule); in a changing climate, this generalization capability could be worth attention. To prove these claims, we represented on 9 the sum up of all experiments carried this far. This figure is a complexity-performace Pareto Plan, which forms a Pareto Frontier, shown in pale gray. As we can see, our best SR models are obviously outperformed by NN models with full pressure levels inputs;

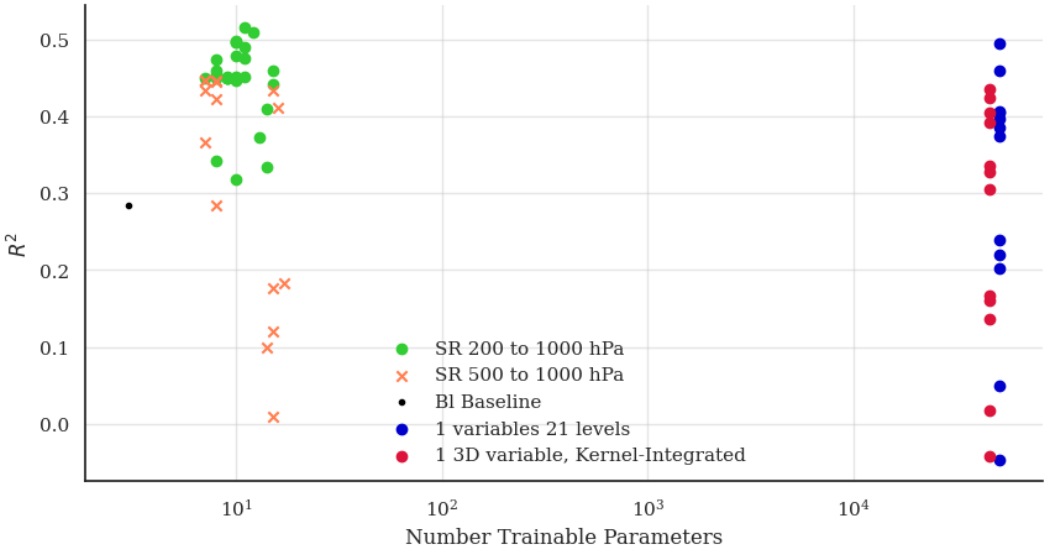


Figure 10: Same Complexity-performance Pareto plane as in 9, but with R^2 computed only on Ocean (LSM < 0.5)

however, this outperformance is not that huge. Using around $\prime(10^1)$ parameters compared to $\mathcal{O}(10^4)$ with NN, we reach R^2 near 0.48 with SR from 200 to 1000 hPa, and 0.45 with SR from 500 to 1000 hPa, these last models interesting us the most. This means that while using fewer pressure levels, and even further integrating them, our models managed to retain some key information, working with a really sparse number of parameters.

We also manage to outperform the linear AN18 B_L Baseline that we try to build upon in this study, by a noticeable margin, going from $R^2 = 0.21$ to $R^2 = 0.41$ for our best model. However, this baseline was purposefully designed to parametrize ocean precipitation. To compare our models on the same grounds we represent the figure 11 and 10, we present on the Pareto plan the R^2 estimated respectively on land and on ocean, to assess the generalization capability of the equations developed. On the land Pareto plan, 11, we can see that while still being outperformed by NN models, our SR equations managed to obtain R^2 around 0.2 for the best models, while the B_L baselines perform at around $R^2 = 0.03$, which is expected considering that it was suited for an ocean prediction. The use of orographical variables to enhance our land generalization capability helps our model to attain this performance; we will go into more detail on this topic on the next section. On the ocean comparison, fig. 10, we see that our SR equations still outperform the B_L baselines, and perform even better than the NN; however, this plot does not show the results on all the NN we developed, only the 'simplest' one, due to time limitations and some computing issues on the Internship. If we had included the best-performing NN showed on fog. 9 (e.g 21 levels + 2D), they would have beat our SR equations.

Finally, in fig. 12, we showed the onset prediction performance using the Mathews Correlation Coefficient as our Performance Metric.

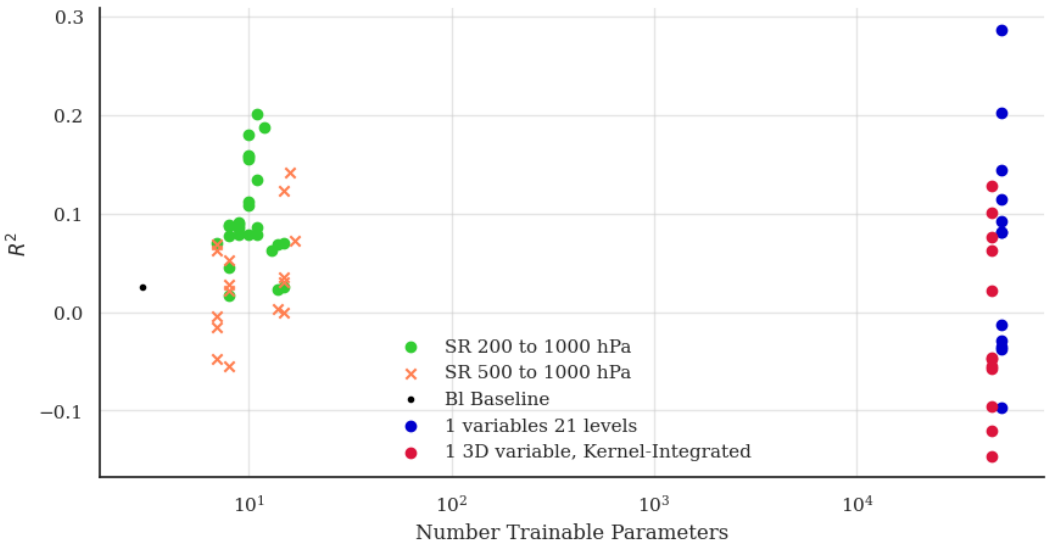


Figure 11: Same Complexity-performance Pareto plane as in 9, but with R^2 computed only on Land (LSM >= 0.5)

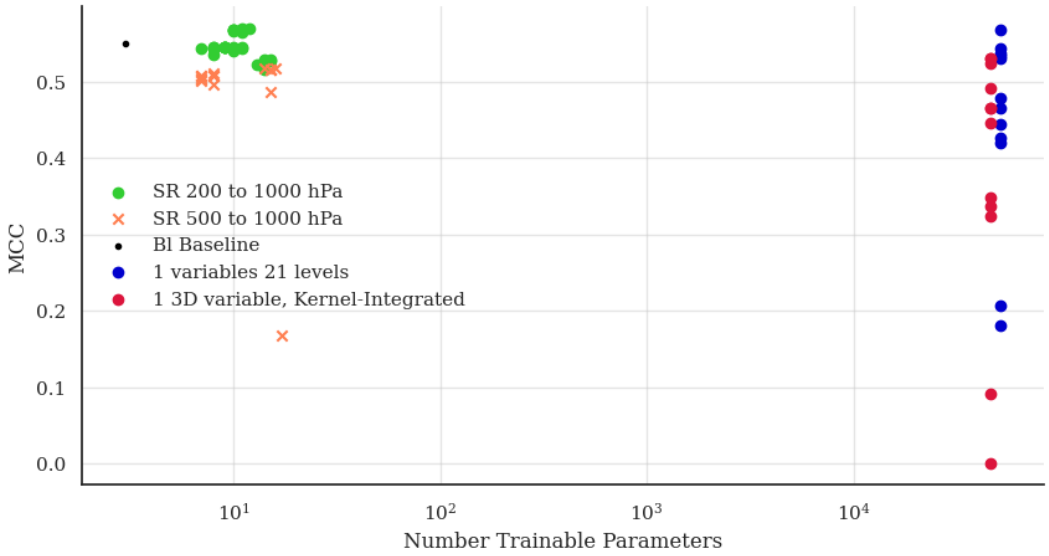


Figure 12: Complexity-Performance Pareto plane, but using the Mathews Correlation Coefficient as an indicator of performance. The models do not change compared to the other Pareto planes, we just transform the model's scalar outputs into boolean to make the problem a Binary Classification one

3.7 Pareto optimal equations

To get a better sense of what these equations We present three Pareto-Optimal equations, P_1, P_2 and P_3, expressed as

$$\begin{aligned} P_1 &= ReLU\left(\lambda_1 e^{\lambda_2 RH + \lambda_3 \sigma_o} + \lambda_4 e^{-\lambda_5 \theta_e^+ + \lambda_6 \sigma_o}\right) \\ P_2 &= ReLU\left(\lambda_1 e^{\lambda_2 RH} + \lambda_4 e^{-\lambda_5 \theta_e^+}\right) \\ P_3 &= ReLU\left(\lambda_1 \left(e^{\lambda_2 RH} + \lambda_3 \sigma_o\right)^2 + \lambda_4\right) \end{aligned} \tag{28}$$

With λ_i being trainable parameters. Their values are displayed on the annex. As we can see they all rely on exponential regarding T/q based variables, namely a positive exponential for RH and a negative one for θ_e^+, which was predictable due to their relation to mean precipitation. Using the exponential of relative humidity in parametrization equations for tropical precipitation is justified for several reasons. As relative humidity approaches 100 percent, or θ_e^+ 0 K, the air becomes increasingly saturated, which is a critical condition for the formation of clouds and subsequent precipitation. Exponential functions naturally capture this behavior, as they rise rapidly as their input approaches zero; in the same ways they create a kind of 'moisture Threshold': below a certain critical value, the exponential term is really small (near zero), indicating limited moisture availability and therefore no precipitation: this creates an onset-like behavior that is awaited in tropical precipitation [2].

Exponential functions also have a clear physical interpretation in the context of moisture-driven processes. The rate of increase in precipitation with increasing humidity is governed by the exponential coefficient, which can then be related to the sensitivity of precipitation to changes in moisture.

Table 1: Scores of 3 Pareto-Optimal equations, with retrained parameters, evaluated on 20 random days of 2003

Equations	R^2	R^2_{land}	R^2_{ocean}	MCC
P_1	0.405 ± 0.012	0.181 ± 0.005	0.473 ± 0.013	0.516 ± 0.007
P_2	0.374 ± 0.011	0.060 ± 0.003	0.469 ± 0.015	0.530 ± 0.009
P_3	0.386 ± 0.021	0.118 ± 0.010	0.468 ± 0.017	0.523 ± 0.013

The utilization of orographical variables within the Symbolic Regression models reveals a noteworthy performance discrepancy in predicting precipitation over land and specific mountainous regions, as seen on 1. This divergence in R^2 values underscores the importance of incorporating orographic factors into precipitation parametrization. P_1 and P_3, while having similar overall R^2, perform way better on land than P_2. That can also be seen graphically on 13 and 14. The results of the P_1 equations appear to be much better than the P_2 counterpart on really mountainous areas like the Andes or the East African Mountains. This justifies our inclusion of orographical variables to try to generalize our model to precipitation prediction on land.

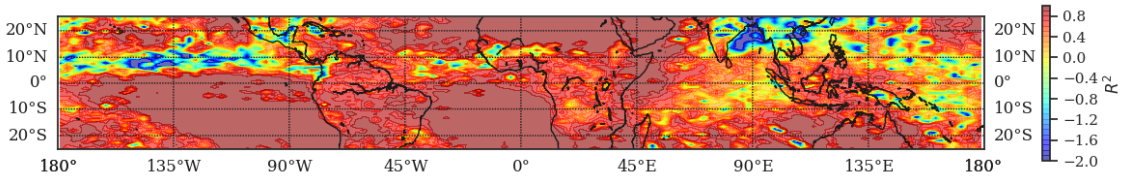


Figure 13: R^2 world map on the tropical latitudes and full longitudes, results obtained by the P_1 model with parameters fitted on 2003 samples. R^2 is computed on each grid point over 80 samples from the year 2002.

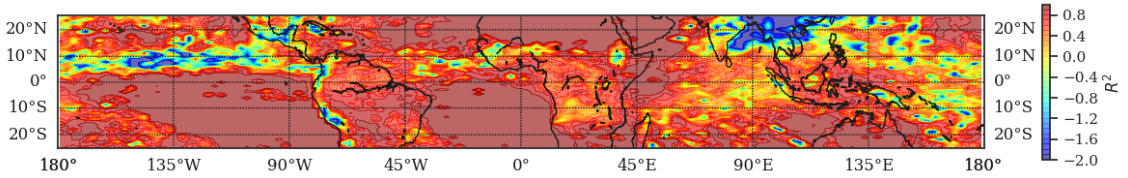


Figure 14: R^2 world map on the tropical latitudes and full longitudes, results obtained by the P_2 model with parameters fitted on 2003 samples. R^2 is computed on each grid point over 80 samples from the year 2002.

The observed lower R^2 scores on land areas can be attributed to the complex interplay between topography and atmospheric dynamics. Mountains disrupt prevailing air masses, leading to orographic lifting, which often results in enhanced precipitation on windward slopes and rain shadows on leeward sides. By including orographical variables in the modeling process, we acknowledge and account for these intricate interactions. The models are thus better equipped to capture the localized variations in precipitation patterns associated with elevation changes.

Conclusion

In conclusion, this master's thesis has undertaken a comprehensive exploration of subgrid parametrization for tropical precipitation, leveraging advanced techniques such as Neural Networks (NN) and Symbolic Regression. Through extensive feature selection using NN, we successfully identified and harnessed key atmospheric variables. We then developed a series of models to discover integrating kernels to reduce these variables from a dozen pressure levels to a single predictor variable. Furthermore, our work extended to the incorporation of orographical variables, revealing promising prospects for improving model accuracy, particularly in the context of land and mountainous regions.

These first steps culminated in the discovery of semi-empirical models with Symbolic Regression, which managed to obtain nearly similar results as Neural Networks while having only around 10 trainable parameters. We also managed baseline models present in the literature on multiple metrics. Importantly, our approach has maintained model sparsity, contributing to model interpretability, with readable equations

Looking ahead, there remain avenues for further exploration. While Symbolic Regression has yielded valuable analytical expressions, future research should delve deeper to ascertain whether these equations represent the optimal representations of the underlying physical processes. Additionally, assessing the model's performance under changing climate conditions remains an essential step, offering critical insights into its generalizability and adaptability.

Furthermore, the potential for expanding this research is vast. Incorporating time series data or broader geographical inputs could enhance the robustness and applicability of the developed models. In this dynamic field of atmospheric science, ongoing investigations hold the potential to refine our understanding of precipitation processes and contribute to improved climate models and forecasting tools. In light of these prospects, this thesis represents a foundational step towards more accurate and versatile subgrid parametrization for tropical precipitation, with ample room for future exploration and advancement.

References

[1] Fiaz Ahmed, Ángel F Adames, and J David Neelin. Deep convective adjustment of temperature and moisture. *Journal of the Atmospheric Sciences*, 77(6):2163–2186, 2020.

[2] Fiaz Ahmed and J David Neelin. Reverse engineering the tropical precipitation–buoyancy relationship. *Journal of the Atmospheric Sciences*, 75(5):1587–1608, 2018.

[3] Tom Georges Beucler, Arthur Grundner, Ryan Lagerquist, and Sara Shamekh. Systematically generating hierarchies of machine-learning models, from equation discovery to deep neural networks (core science keynote). In *103rd AMS Annual Meeting*. AMS, 2023.

[4] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, pages 1–6, 2023.

[5] Michel Déqué, Christine Dreveton, Alain Braun, and Daniel Cariolle. The arpege/ifs atmosphere model: a contribution to the french community climate modelling. *Climate Dynamics*, 10:249–266, 1994.

[6] Kerry A Emanuel. *Atmospheric convection*. Oxford University Press, USA, 1994.

[7] Pierre Gentine, Mike Pritchard, Stephan Rasp, Gael Reinaudi, and Galen Yacalis. Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, 45(11):5742–5751, 2018.

[8] Arthur Grundner, Tom Beucler, Pierre Gentine, and Veronika Eyring. Data-driven equation discovery of a cloud cover parameterization. *arXiv preprint arXiv:2304.08063*, 2023.

[9] Arthur Grundner, Tom Beucler, Fernando Iglesias-Suarez, Pierre Gentine, Marco A Giorgetta, and Veronika Eyring. Deep learning based cloud cover parameterization for icon. *arXiv preprint arXiv:2112.11317*, 2021.

[10] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[12] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[13] Stephan Rasp, Michael S Pritchard, and Pierre Gentine. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, 115(39):9684–9689, 2018.

[14] Tapio Schneider, Shiwei Lan, Andrew Stuart, and Joao Teixeira. Earth system modeling 2.0: A blueprint for models that learn from observations and targeted high-resolution simulations. *Geophysical Research Letters*, 44(24):12–396, 2017.

[15] Tapio Schneider, João Teixeira, Christopher S Bretherton, Florent Brient, Kyle G Pressel, Christoph Schär, and A Pier Siebesma. Climate goals and computing the future of clouds. *Nature Climate Change*, 7(1):3–5, 2017.

[16] Sara Shamekh, Kara D Lamb, Yu Huang, and Pierre Gentine. Implicit learning of convective organization explains precipitation stochasticity. *Proceedings of the National Academy of Sciences*, 120(20):e2216158120, 2023.

[17] Francesco Zanetta, Daniele Nerini, Tom Beucler, and Mark A Liniger. Physics-constrained deep learning postprocessing of temperature and humidity. *arXiv preprint arXiv:2212.04487*, 2022.

[18] Weining Zhao and MAK Khalil. The relationship between precipitation and temperature over the contiguous united states. *Journal of climate*, 6(6):1232–1236, 1993.