Building confidence in data-driven surrogate transport models for turbulent plasmas

Robin Varennes, Zhisong S Qu, Youngwoo W Cho, Chenguang Wan, Kunpeng Li, Robin A Heinonen, Virginie Grandgirard

Building confidence in data-driven surrogate transport models for turbulent plasmas

R. Varennes[1], Z. S. Qu[1], Y. W. Cho[1], C. Wan[1], K. Li[1], R. A. Heinonen[2], V. Grandgirard[3]

[1]School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore 637371, Singapore

[2]Department of Physics and INFN, University of Rome, "Tor Vergata", 1 Via della Ricerca Scientifica, 00133 Roma, RM, Italy

[3]CEA, IRFM, F-13108 Saint-Paul-Lez-Durance, France

E-mail: `robin.varennes@gmail.com`

Abstract. Getting fast and reliable predictions of turbulent transport properties is an important challenge in magnetic fusion. Previous research [R. A. Heinonen & P. H. Diamond, Phys. Rev. E **101** 061201 (2020)] proposed a data-driven approach using neural networks to predict the particle flux and Reynolds stress in a minimal model of drift-wave turbulence. The present work extends this approach to the interchange instability driven by the magnetic curvature. An assessment of the limits and caveats associated with a data-driven approach based on machine learning regression algorithms is performed — an essential step for scalability toward more complex high-fidelity codes. In particular, a figure of merit is introduced to indicate regions within the parameter space where the neural network outputs can be trusted. Some applications of the data-driven surrogate model are presented. Specifically, predictions are used to gain insight into the vorticity gradient's contribution to the turbulent flux and the antiviscous nature of the Reynolds stress.

1. Introduction

One of the challenges to operate commercial nuclear fusion reactors is to confine a hot magnetized plasma. Indeed, extreme thermodynamical gradients in the reactor vessel lead to heat and particle transport, carried mainly by turbulence. To this day, a comprehensive description of turbulence remains elusive.

The present study addresses the complex issue of turbulent transport and its interaction with flows through data-driven surrogate modeling. Surrogate — or reduced — models provide approximations of outcomes that are difficult to measure or compute. They represent an important activity in the fusion community. These models are typically based on exact equations that are simplified in order to be quickly solved computationally. Some advanced reduced models offer remarkable predictive capabilities [1–5]. However, these models have limitations, particularly in scenarios where the

underlying approximations are not valid. With the increasing computing resources and development of large databases, a data-driven approach appears as a promising alternative. Such an approach relies on learning the patterns present in trustworthy data, typically from first-principle simulations or experimental data. These models are powerful tools to get a fast approximation of an outcome of interest and can be used to explore the underlying physics of the system. An impressive example of a data-driven surrogate model applied to a turbulent system is *Graphcast* for weather forecasting [6], which is typically as precise as an advanced numerical simulation for a fraction of the computational cost.

The goal of the present study is to apply such a data-driven approach to turbulent transport in tokamak plasmas, first with simple machine learning and low-fidelity simulation data. This preliminary work is meant to pave the way for more complex and high-fidelity simulations and to identify the challenges and constraints of such an approach. In this study, such a model is built using a machine learning optimization algorithm based on neural networks [7]. It allows linking some *features* — e.g. density gradient, vorticity... — to some quantities of interest that one wants to predict, here the turbulent particle flux and Reynolds stress. Building confidence in such a data-driven model is a key aspect of the study. Indeed, neural networks are often referred to as "*black boxes*" as each neuron applies nonlinear transformations to the input data, resulting in highly nonlinear and complex relationships between input features and output. In addition, the outcome of neural network models depends on a certain number of hyperparameters, i.e. tunable parameters specific to the training process chosen relatively arbitrarily and adding to the feeling of distrust of such an approach. Moreover, such models are trained on a limited amount of data ranging in a specific parameter space. The model's predictions of an outcome for a set of parameters not included in the training set range are generally not reliable. For multi-dimensional problems, it can be challenging for a user to know if a prescribed set of inputs is within the range of the training set. For a model trained on physical data, if the dataset used to train the model is representative of the whole range accessible by the physical system, asking the model for a prediction in a region of the parameter space where the model has not been trained would not only be unreliable but also not physically meaningful. For this reason, a measure of the confidence level of the set of features provided to the model is proposed in this study.

Applying such techniques to data coming from first principle codes is a formidable task, as it involves a large number of parameters to scan, and necessitates a consequent computing power and management of large amounts of data. Consequently, as a preliminary step, the data in this study is generated using the TOKAM2D code [8, 9] that describes simplified turbulence in a 2D plane transverse to the magnetic field in the low field side edge of a tokamak. This framework is chosen for its simplicity regarding the data generation, the associated storage and post-treating procedure, as well as the limited number of features for training. Moreover, despite being referred to as "simplified", the turbulence in this model is not simple. The non-linear interactions

governing this model are still not fully understood, making it relevant to the study of edge turbulence in tokamaks. Thus, this preliminary study is also used to gain insight into the underlying physics of the system. While the bulk of the physical analysis is left for future work, some preliminary results are presented. In particular, the vorticity gradient contribution to the turbulent flux and the anti-viscous behavior of the Reynolds stress are discussed.

Note that there is a renewed interest in simple plasma turbulence models, as they appear as a logical primary approach for proof-of-principle studies in machine learning applied to plasma physics. These applications target a wide range of topics: surrogate modeling [10–12], acceleration of simulations [13, 14], recovering dynamics from partial observations [15] or determining the governing equations of a system [16]. More generally, the new applications offered by machine learning in turbulent systems are developing at a fast pace. While the literature is too vast to cover, some recent reviews and perspectives on the topic can be found in [17–19].

The remainder of the paper is organized as follows. A brief overview of the TOKAM2D code and associated physics is addressed in Section 2. Some specific challenges associated with such data-driven methods for surrogate modeling of turbulence are discussed in Section 3. The making of the training datasets is addressed in Section 4. The machine learning model structure is described in Section 5. Two methods — previously unreported — to gain confidence in the data-driven model are presented in Section 6. Some results of the trained model, with an emphasis on the vorticity gradient contribution to the flux and the anti-viscous behavior of the Reynolds stress, are presented in Section 7. The Section 8 closes the paper.

2. Description of the solved equations

This paper presents a robust numerical approach to obtain surrogate models based on simulation describing intricate non-linear physics. Edge tokamak plasmas turbulence is chosen in this study, and the data is generated by solving a simplified set of fluid equations that couples two fields, i.e. density fluctuations and the vorticity, while restricting the phase space to the directions transverse to the magnetic field, typically r a radial and θ an angle coordinate.

These equations are namely the modified [20] Hasegawa-Wakatani equations [21] that describe turbulence originating from the drift waves instability and its interactions with zonal flows. In this study, this set of equations is extended with the addition of the interchange instability [22, 23] and read

$$\partial_t n + [\phi, n] + \kappa \partial_y \phi + D\nabla^4 n = C(\tilde{\phi} - \tilde{n}) \tag{1a}$$

$$\partial_t \mathcal{W} + [\phi, \mathcal{W}] + g\partial_y n + D\nabla^4 \mathcal{W} - \nu \mathcal{W} = C(\tilde{\phi} - \tilde{n}) \tag{1b}$$

where

- t is the time coordinate in unit of c_0/L with c_0 the sound speed of ion at electron temperature and L a reference length;
- $x = (r-a)/\rho_0$ and $y = a\theta/\rho_0$ are the normalized radial and poloidal coordinates with a the minor radius of the tokamak and ρ_0 a reference ion Larmor radius;
- n is the density fluctuation in unit of $n_0\rho_0/L$ with n_0 a reference density;
- ϕ is the electric potential in unit of $\rho_0 T_e/Le$ with T_e the electron temperature and e the electron charge;
- $\mathcal{W} = \nabla^2\phi$ is the vorticity with $\nabla^2 = \partial_x^2 + \partial_y^2$;
- $\kappa = -(L/\rho_0)(\nabla_x \ln n_0)$ is the inverse background density gradient length which is supposed homogeneous in the x direction;
- C is the *adiabatic parameter* responsible for the drift waves instability in unit of $c_0 n_0 e^2 \eta_e / L T_e k_z^2$ with η the resistivity and k_z the parallel wave number;
- g is the magnetic curvature amplitude responsible for the interchange instability, proportional to the inverse major radius $1/R_0$;
- D is the hyperdiffusion coefficient and ν a friction coefficient.
- The Poisson bracket is defined as $[f, g] = \partial_x f \partial_y g - \partial_y f \partial_x g$.
- The non-zonal part of the density and potential is defined as $\tilde{f} \equiv f - \langle f \rangle_y$ where $\langle . \rangle_y = (1/L_y) \int_0^{L_y} \{.\}\, dy$ is the flux-surface average.

In this set of equations, the drift waves instability stems from the phase shift between the electric potential and the density fluctuations, for which the amplitude depends on the adiabatic parameter C. In this model, this phase shift is mainly due to the effect of collision and two asymptotic regimes can be defined: *adiabatic* when $C \to \infty$ — i.e. the collisionless case that reduces Eqs(1) to the Hasegawa-Mima equations [24] describing stable drift waves — and *hydrodynamic* when $C \to 0$ — i.e. the highly collisional case describing the resistive drift wave instability that impacts mostly large radial scale $k_x \sim 0$ and is always unstable in the presence of a finite density gradient and the absence of dissipation. The interchange instability stems from the magnetic curvature which amplitude is set through the parameter of control g. This term is sometimes referred to as an *effective gravity* as it bears an analogy with the Rayleigh-Bénard instability [25] for a neutral fluid with a temperature gradient directed opposite to gravity. Here, the density gradient has the role of the temperature gradient and the magnetic curvature the role of the gravity. The main drive of these instabilities is the background density gradient described by κ. Note that the Kelvin–Helmholtz instability can also be destabilized in this system for some specific conditions. The reader interested in the linear stability of such a system can have a look in [26]. Furthermore, only the non-zonal part is kept in the adiabatic term $C(\tilde{\phi} - \tilde{n})$. This modification [20] of the standard Hasegawa-Wakatani equations incorporates the fact that electrons have a vanishing response to zonal flows. In practice, this leads the system to generate finite and steady zonal flows.

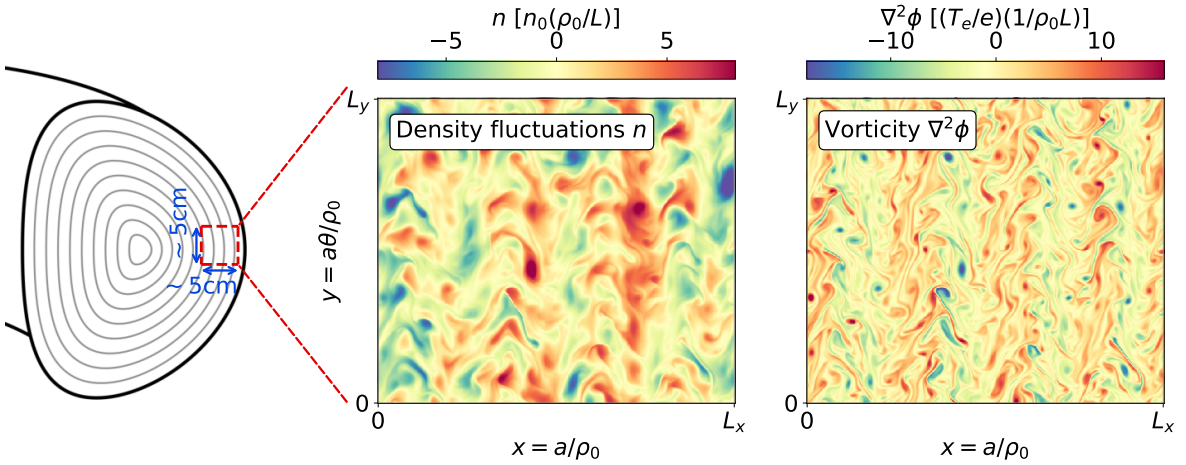

Figure 1. Overview of the domain in a typical simulation of the Hasegawa-Wakatani equations with the code TOKAM2D and snapshot of density fluctuation n and vorticity W. The order of magnitude of the duration of a simulation is $\sim 25\mu s$. Details of the simulations mesh and parameters are given in Section 4.

The equations Eqs(1) are solved using the code TOKAM2D [8, 9] in a 2D domain periodic in both directions. The code is based on a pseudo-spectral method with a 2/3 de-aliasing rule and a 4th-order Runge-Kutta time integration. With the Fast Fourier Transform being the main operation, the GPU acceleration of the code is highly efficient. An overview of the simulation domain and typical snapshots of the outputs are shown in Fig.1.

The parameters of control of TOKAM2D are the adiabatic parameter C, the curvature term g, the hyperdiffusion coefficient D, the friction ν, the background density gradient κ, the size of the domain L, the initial density fluctuation $n(x, y, t = 0)$ and vorticity $\mathcal{W}(x, y, t = 0)$, and the resolution of the simulation.

3. Beyond the physics: the challenges of turbulence modeling from data

The machine learning algorithm and associated models used in this study make use of conventional neural networks for regression. Yet, multiple challenges specific to the turbulence modeling from data have to be addressed. To illustrate these challenges, Fig.2 displays typical profiles of the turbulent particle flux and Reynolds stress — i.e. the quantities to predict — obtained with a TOKAM2D simulation.

The first remark is that these profiles are heavily fluctuating. A complete model would ideally provide a probability distribution of these quantities. The present work targets a minimal statistical approach to predict a single mean-field value of flux or Reynolds stress which is in line with the mean-field theory of turbulence [27, 28]. On the numerical side, this makes the evaluation of the model performance challenging. Indeed, the metrics used to assess the model are usually functions of the difference

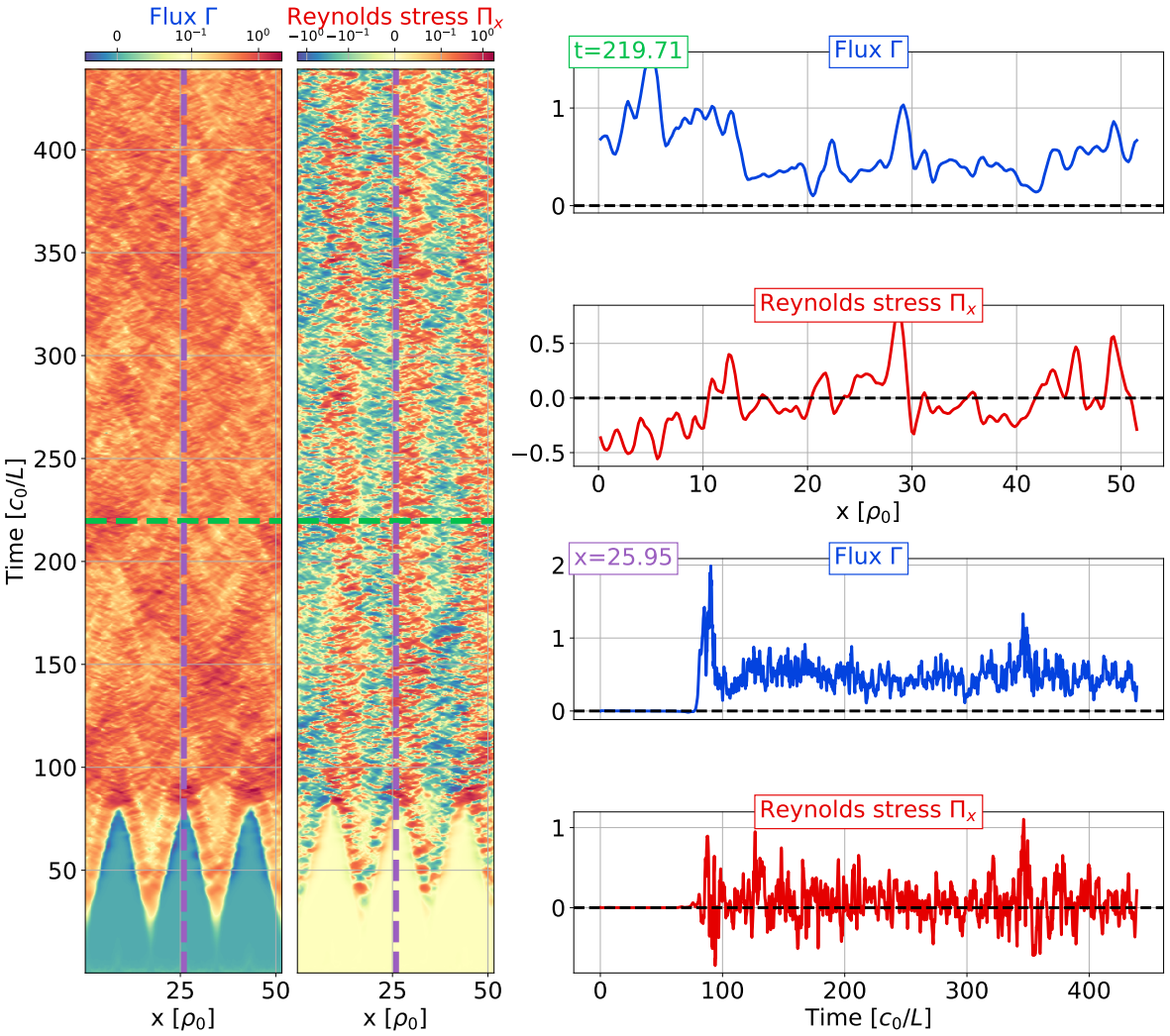

Figure 2. Typical profiles of the turbulent particle flux and Reynolds stress obtained by numerically solving the modified Hasegawa-Wakatani equations Eqs(1) with the TOKAM2D code, here initialized with a radial perturbation of the vorticity. Color maps on the left show the spatiotemporal evolution of these quantities in a typical simulation. The profiles on the right show their radial profile at a given time (top) and their time evolution at a given radial position (bottom).

between the predicted value — here intended to represent the mean — and the true value that includes fluctuations — obtained through direct numerical simulations. This difference then carries a combination of aleatoric uncertainty, which arises from the inherent chaotic variability associated with turbulence that leads to fluctuations, and epistemic uncertainty, which stems from the limitations of the model in accurately predicting the mean behavior. In other words, compared to typical machine learning regression problems where one's goal is to make this difference as small as possible, here this difference should tend toward a measure of the fluctuations' level. As the average level of fluctuation is not known *a priori*, this makes the evaluation of the

model performance more difficult, especially for the Reynolds stress which has a very low signal-to-noise ratio — generally below 10%.

The different nature of these two outputs also raises the question of the model's structure. Indeed, as developed in Section 5, a neural network is composed of layers of neurons that shuffle the information of some inputs to predict the outputs. It is then tempting to use a single neural network to predict both outputs, with the idea that the network will capture some general patterns from the physics and be able to predict both outputs simultaneously. However, it could happen — and it is the case in this study — that the outputs are so different that one can dominate the other in the learning process. This particular issue is addressed in Appendix A.

The other challenges are the generation of the training datasets, the choice of the inputs — called *features* — and evaluating the reliability of the model's predictions. These challenges are addressed in the following sections.

4. Making of the dataset

While the Hasegawa Wakatani equations Eqs(1) describe the time evolution of 2D fields, the goal of this study is to obtain a surrogate model predicting flux-surface averaged quantities at equilibrium. The quantities of interest to predict are the flux-averaged radial turbulent particle flux $\Gamma = \langle n v_{Ex} \rangle_y$, and Reynolds stress $\Pi = \langle v_{Ex} v_{Ey} \rangle_y$ with v_{Ex} and v_{Ey} the radial and poloidal component of the electric drift respectively. The latter can be understood as the radial flux of poloidal momentum.

The features, i.e. the quantities chosen to predict the outputs, are here chosen in regard to the physical problem at hand. Here, one wants to gain insight into the flux and the Reynolds stress. Based on previous studies on reduced modeling [29–31], the chosen features are the flux-surface averaged total density gradient $N' = -\kappa + \partial_x \langle n \rangle_y$, vorticity $W = \langle \mathcal{W} \rangle_y$, its gradient $W' = \partial_x W$, and potential enstrophy $\varepsilon = \langle (\tilde{n} - \nabla^2 \tilde{\phi})^2 \rangle_y$. The latter can be understood as a proxy for the turbulent intensity.

Choosing such features limits the scope of the model to theoretical studies as the vorticity and potential enstrophy are not controllable nor directly measurable in experiments. For the long-term applicative goal of such work, i.e. building a model on high-fidelity simulation data and/or experimental data, the nature of features should be constrained by engineering parameters and/or experimentally accessible quantities.

The choice of these quantities is justified as there is still ongoing research on linear and quasilinear [32] models of turbulence to understand their role in the resulting flux (of particle or momentum, i.e. the Reynolds stress). In particular, the choice of the potential vorticity — which is, in essence, a turbulent intensity — as an output and not an input is justified by the fact that the physics that governs its evolution is not fully understood. Multiple works [30, 31, 33, 34] indicate that the flux of potential vorticity, which can be related to the very active topic of turbulence spreading, is not purely diffusive and more modeling is needed to understand its behavior. This study only scratches the surface of this physics in Section 7 and the bulk of the physical analysis

is left for future work, the focus being on the framework of the data-driven approach.

In addition to the physical relevance of the chosen features, the choice of these quantities is also pragmatic as they evolve in time such a substantial amount of data can be obtained from a single simulation.

With the features chosen, one now has to deal with the exploration of the parameter space. Indeed, one is interested in equilibrium states that cover a parameter space whose bounds are *a priori* unknown and can only be explored through the control parameters of the simulation. One must then choose reasonable intervals of these control parameters and choose a sampling technique to ensure good coverage of the parameter space. Typical sampling methods include grid search, random search, Latin Hypercube Sampling [35] or Bayesian optimization [36]. In the present study, one wants to cover the widest range of possible density, electric potential and their associated spatial derivatives since all quantities of interest can be expressed as combinations of these parameters. Three control parameters are then considered: the background density gradient κ, the magnitude of the initial electric potential ϕ_0 and its radial harmonic k_{x0} such that $\phi(t = 0, x, y) = \phi_0 \cos(2\pi k_{x0} x)$. A random search of 50 simulations is performed in the interval $-4 < \kappa < 4$, $50 < \phi_0 < 50$ and $k_{x0} = [1, 2, 3]$ with a bias toward sets of parameters which yield a positive linear growth rate Γ_{lin}. Some simulations with a negative linear growth rate are also included in order to have data to train the model to recover sets of features that should return vanishing flux and Reynolds stress. The resulting scan is displayed in Fig.3.

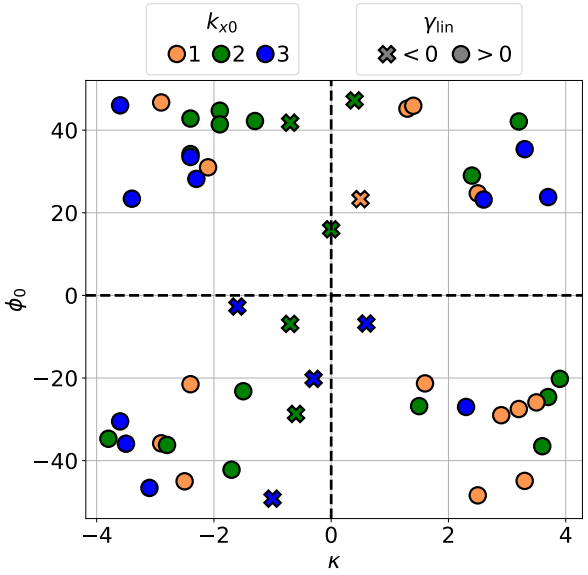

Figure 3. Scan of the control parameters — the background density gradient κ, the magnitude of the initial electric potential ϕ_0 and it's radial harmonic k_{x0} — for the making of the database. The circles/crosses indicate a positive/negative linear growth rate γ_{lin}.

Other control parameters are set to fixed values for each simulation. The adiabatic

	Different Parameters	**Fixed parameters**			**Simulation mesh**				
	Interchange parameter	Viscosity	Hyper-diffusion	Adiabatic parameter	Space domain	Space mesh	Sim. time	Data save step	RK4 step
	$\boldsymbol{g}$ $[\rho_0/R_0]$	$\boldsymbol{\nu}$ $\left[\frac{L\rho_0}{c_0}\right]$	$\boldsymbol{D}$ $\left[\frac{L\rho_0^4}{c_0}\right]$	$\boldsymbol{C}$ $\left[\frac{c_0 n_0 e^2 \eta_e}{LT_e k_z^2}\right]$	$\boldsymbol{L_x}\|\boldsymbol{L_y}$ $[\rho_0]$	$\boldsymbol{N_x}\|\boldsymbol{N_y}$ $[-]$	$\boldsymbol{T_{\text{sim}}}$ $[\gamma_{\text{lin}}^{-1}]$	$\boldsymbol{\Delta t_{\text{diag}}}$ $[L/c_0]$	$\boldsymbol{\Delta t}$ $[L/c_0]$
DW only	0	0.01	1.5×10^{-4}	2.0	51.5	256	50	$\frac{T_{\text{sim}}}{2000}$	$\frac{\Delta t_{\text{diag}}}{512}$
DW+ Interchange	1								

Table 1. Fixed simulation control parameter used for the generation of both dataset: one with only the drift-waves instability and another with both drift-waves and interchange instabilities. Scanned parameters are displayed in Fig.3. Note that γ_{lin} is an estimation of the linear growth rate.

parameter is set to $C = 2$, the hyperdiffusion coefficient to $D = 10^{-4}$, the friction to $\nu = 10^{-2}$, the size of the domain to $L = L_x = L_y = 51.5\rho_0$ and the space resolution to 256×256. The duration T_{sim} of each simulation is set to 50 inverse linear growth rate Γ_{lin}^{-1} and the data is saved every $\Delta t_{\text{diag}} = T_{\text{sim}}/2000$ while the time step of the 4th order Runge–Kutta scheme is $h = \Delta t_{\text{diag}}/512$. Two training datasets are made, one without interchange, i.e. $g = 0$ and directly comparable to [10], and one with interchange $g = 1$. A summary of the simulation control parameters for both datasets is given in Table 1.

The resulting distribution of each feature and output for each dataset is displayed in Fig.4. It gives an idea of the range of values of the features and outputs, and shows that the distributions are quite different for both datasets. In addition, one can also observe that no particular effort has been made to ensure fairly distributed features, as will probably be higher-fidelity datasets in the future. For example, the distribution of the density gradient N' for the dataset with drift waves only would have been symmetric around 0 if the same number of simulations with positive and negative κ had been performed, which is not the case with the random search performed here.

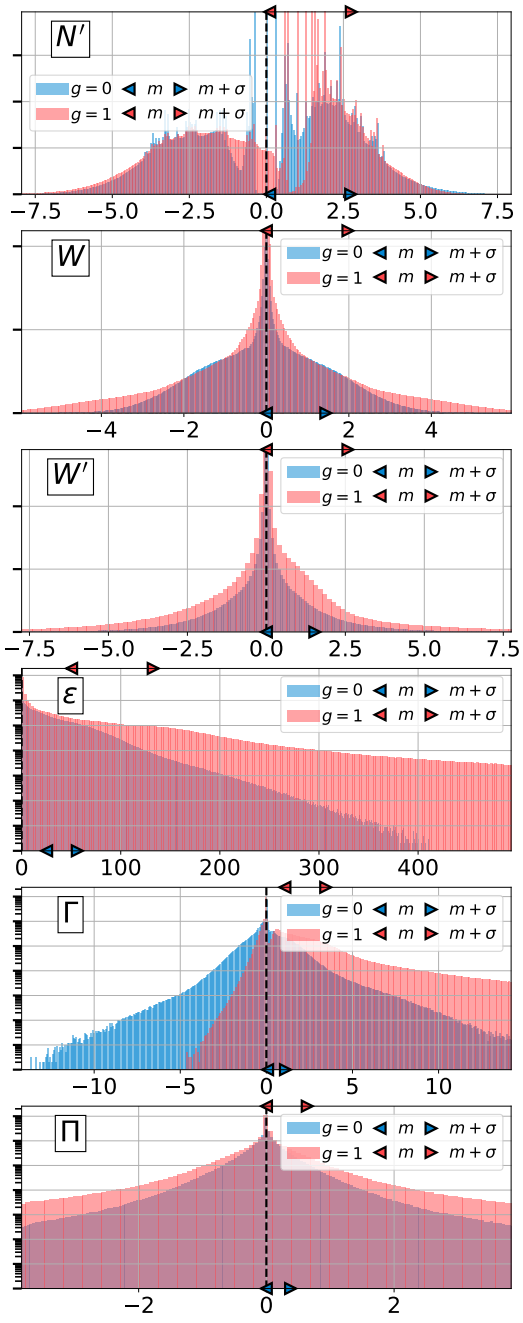

Figure 4. Distribution of each feature/output for each dataset. Blue: with drift waves only $g = 0$, red: with both drift waves and interchange $g = 1$. m and σ indicate the mean and standard deviation of the distributions. Note that the y-axis of the three last figures is in log scale.

In the next section, these datasets are used to train neural networks to predict the turbulent particle flux and Reynolds stress.

5. Description of the model

With the generated datasets, a neural network is trained to predict the (Γ, Π) values from the inputs (N', W, W', ε). A brief description of regression neural networks is given here, complemented by a sketch in Fig.5.

A neural network is a machine learning model inspired by the human brain, composed of layers of neurons that process information. The typical structure involves an input layer, one or more hidden layers, and an output layer. The input layer is composed of one neuron per input variable — in our case (N', W, W', ε) — that returns identity. These inputs' values are then multiplied by a tunable weight and passed to each neuron of the next layer, i.e. the first hidden layer. In hidden layers, each neuron is a mathematical function that performs two operations: a linear combination of the neuron's input to which a bias is added, and the application of a non-linear activation function. This shuffles the data in a non-linear way, allowing the network to learn complex patterns. The number of hidden layers and neurons per layer is chosen by the user and depends on the complexity of the problem.

The output layer is composed of one neuron per output variable. In a regression problem, these neurons return only a linear combination — i.e. without the activation function — of the last hidden layer output to predict the output variables. The predictions of the neural network are then compared to the true values present in the training dataset through a loss function. Using this error measure, the weights and biases can be updated through the backpropagation algorithm that propagates the error gradient from the output layer to the input layer in order to minimize the loss function. This process, also called *gradient descent*, estimates the direction in which the weights and biases should be updated to reduce the error, and their value updated using a step called the *learning rate* η, i.e. a scalar characterizing the step size to reach a minimum of the loss function. Too small learning rates can lead to slow convergence, while too large learning rates can actually prevent convergence. The process is repeated through each iteration — called *epochs* — up to a satisfactory level of precision for the output that is characterizable through different metrics.

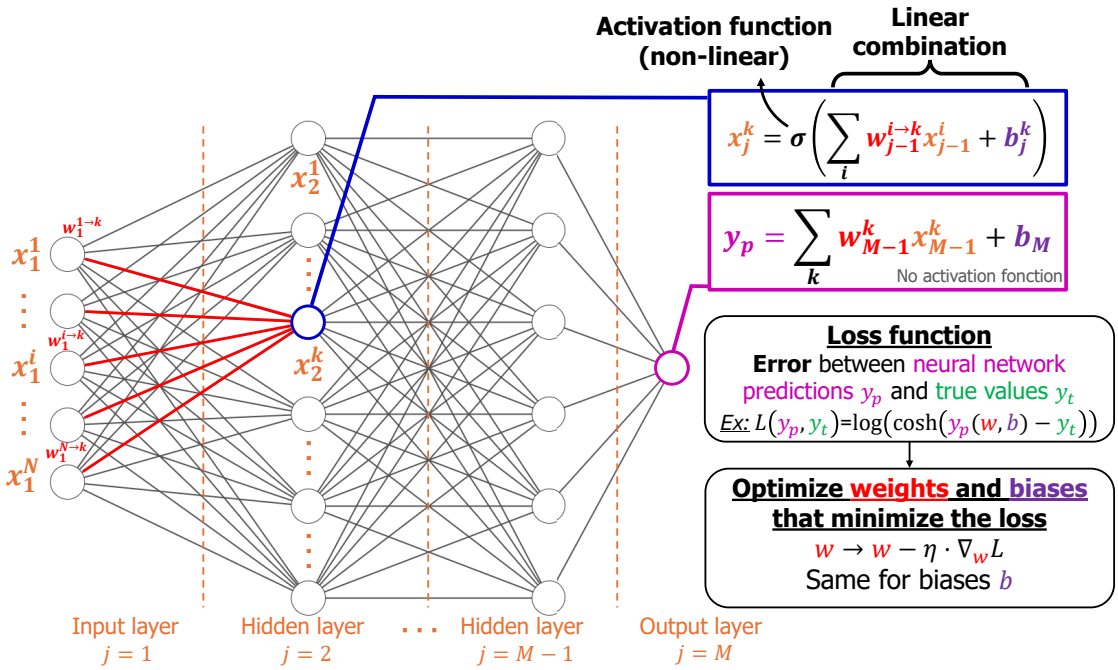

Figure 5. Sketch of a basic regression neural network. The input layer is composed of one neuron per input variable, i.e. (N', W, W', ε). The hidden layers are composed of a number of neurons chosen by the user that apply a linear combination of the previous layer output — with tunable weights and biases — followed by a non-linear activation function. On the last layer, the neurons return just a linear combination of the last hidden layer output to predict the output variables, i.e. (Γ, Π). The predictions are compared to the true values through a loss function, and the weights and biases are updated using backpropagation through each epoch up to a satisfactory level.

In this study, two outputs are to be predicted, the turbulent particle flux Γ and the Reynolds stress Π. This raises the question of the model's structure, which is discussed in Appendix A. Ultimately, the choice is to train one neural network per output.

The activation function in the hidden layers is the *exponential linear unit* (ELU) function [37], reading

$$\sigma(x) = \begin{cases} e^x - 1 & \text{if } x \leq 0 \\ x & \text{if } x \geq 0 \end{cases} .$$

The chosen loss function is the *logcosh* function with a L^2 regularization, reading

$$\mathcal{L} = \ln(\cosh(y_t - y_p)) + \lambda ||W||^2 \tag{2}$$

where y_t is the true values of (Γ, Π) and y_p the predicted ones. The regularization term $\lambda||W||^2$, with W the weights matrix and $||.||$ the Frobenius norm and $\lambda = 10^{-5}$, is added to the loss function to prevent overfitting, i.e. learn the noise in the data.

The learning rate η is adaptively set by the *Adam* optimizer, a variant of the stochastic gradient descent algorithm [38]. In addition, a batch normalization [39]

is applied after each hidden layer. Batch normalization is a technique used to make learning more stable and faster by normalizing the inputs to each layer, ensuring they have a consistent scale and distribution.

The dataset is split into a training, validation and test sets, with respectively 70%, 15% and 15% of the data. The training set is used to train the model and adjust its weights and biases, the validation set is used to tune the hyperparameters of the model and assess the model performance during the training, and the test set is used to evaluate the final model's performance without any bias.

The hyperparameters that are kept tunable are:

- the number of epochs;
- the batch size, i.e. the number of samples used to estimate the error gradient at each iteration;
- the number of hidden layers and neurons per layer, related to the maximum level non-linearity of the model;
- the random seed, that is involved in multiple parts of the training process, such as the initialization of the weights and biases, the sampling of the data in different subsets and the shuffling of the dataset.

Given the relatively low dimensionality of the dataset and the simplicity of the model, it is possible to test a large set of hyperparameters. As detailed in the next section, it turns out that the choice of hyperparameters does not significantly impact the model's performance. The surrogate model is then relatively independent on the training parameters, which is reassuring regarding the actual presence of patterns in the data as well as the model's robustness.

In the next section, the metrics used to evaluate the models' performance and their range of validity are presented.

6. Gaining confidence in the trained model

One important task when using machine learning for surrogate modeling is to assess the model's confidence in its predictions. There are multiple elements to discuss.

First, the choice of the features is an essential step. Indeed, when selecting a set of features, one exposes itself to the risk of model misspecification and/or multicollinearity. The former indicates that the base formed by the features is not rich enough to explain the outputs, while the latter indicates that some features are highly correlated and do not add enough independent information to the model results. Here the misspecification assessment is tricky for reasons discussed below and is mostly approached through an *a posteriori* analysis of the model predictions in Section 7. The multicollinearity is assessed in Appendix B. In addition, it is good practice to perform a sensitivity analysis of the features to assess their weight in the model predictions. Indeed, it could happen that some features have a very low contribution to the outputs, and could be removed

from the model without affecting the predictions. This is estimated thanks to a feature importance algorithm, as detailed in Appendix C.

Two particular concerns specifically related to this work are further discussed below. First, as the evaluation of the model performance is challenging due to the fluctuating nature of the outputs, the sensitivity of the model to the hyperparameters is assessed. Second, a measure of the range of validity of the model is introduced to assess the reliability of the model predictions.

6.1. Invariance of the model to hyperparameters

It is usual to use scalar values, called *metrics*, as a figure of merit of a neural network model performance. These metrics are usually functions of the residuals, i.e. the difference between the predicted and true values.

The most common metrics for regression problems are

- Loss function: as defined in the previous section, the loss function quantifies the error between the predicted and actual values.
- Mean Squared Error (MSE): the average of the squared differences between the predicted and actual values.
- Mean Absolute Error (MAE): the average of the absolute differences between the predicted and actual values.

These values can be calculated for each output of the model, i.e. (Γ, Π), and for each subdataset, i.e. training, validation, and test sets.

The most unbiased metrics are the ones evaluated on the test set, i.e. a particular subset of the dataset which is not involved in any part of the training process, thus ensuring that the model generalizes well to new data. As discussed in the previous section, multiple hyperparameters are involved in the training process. As the trained datasets are relatively small and easy to handle numerically, multiple sets of hyperparameters have been tested. This has been done not so much to find the most optimal set of hyperparameters, but rather to ensure that the choice of hyperparameters does not significantly affect the model predictions. Indeed, our claim is that, if the data can indeed be explained by relatively simple surrogate models, then the choice of hyperparameters should not significantly affect the model predictions. This invariance to the hyperparameters is, in our opinion, a good indicator of the model performance and robustness. Fig.6 shows the MAE (in TOKAM2D unit) of multiple models obtained on the test subdataset for a wide range of hyperparameters, for each output and dataset. For each output and dataset, the variance in the test MAE is small and there is no clear trend with any hyperparameters. Note that the actual value of this MAE is not indicative of the model performance. Instead, they should be understood as some measure of the output fluctuations. For both outputs, it appears that the test MAE is significantly larger for the dataset with both drift waves and interchange than for the

dataset with drift waves only. This is expected, as the interchange is superimposed on the drift waves, meaning that, on average, the fluctuations in this dataset are larger.

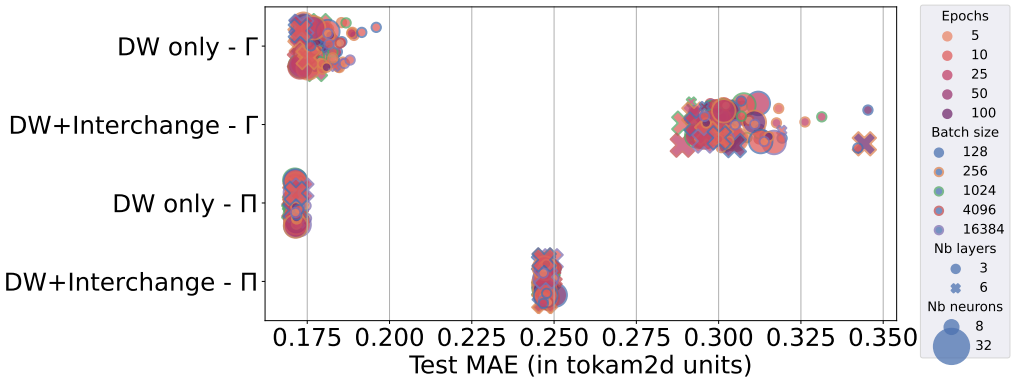

Figure 6. Mean average error evaluated for the test dataset for models trained with a wide range of hyperparameters for the flux Γ and the Reynolds stress Π and for both datasets: with drift waves only and with both drift waves and interchange. The random seed is also scanned here (with values 2, 4, and 6) but not shown in the legend.

This compact view of the model performance hides multiple information, as a single scalar value does not give any insight into the distribution of the residuals along the whole range of values for each output. In other words, the model could be very accurate for some values of the output and very inaccurate for others.

In addition, Fig.6 does not clearly inform on the overfitting (even if strong overfitting would lead to larger MAE values). By analyzing the performance of a randomly chosen handful of models more closely, it seems that very few epochs are needed to reach a satisfactory level of precision for the outputs (typically 5 or 10), and overfitting can start to occur below 100 epochs but does not significantly affect the model predictions for the values tested here.

In the following, the model chosen for all the predictions has a batch size of 256, 3 hidden layers with 8 neurons each, and is trained for 10 epochs.

6.2. Range of validity of the model

A last step, specific to the making of surrogate models on turbulent systems, is to get some evaluation of the model range of validity. Indeed, the trained model predictions result from the operation done in the neural network, which is in essence a function that concatenates linear and non-linear operations. As such, the model does not have any constraint on the user inputs provided to it to get predictions. Consequently, the model gives a prediction for any inputs, even if they are very far from the inputs of the training data. As the dataset comes from a turbulent system, knowing if a user input (N', W, W', ε) is accessible by the system is challenging without looking in detail at the training data. One solution to get an idea of the validity of the user inputs is to look

at a measure of distance from the user input to the nearest neighbor in the training dataset. Such a measure is useful to tell if the user input fed to the model is far from the training data, in which case the model prediction should be taken with caution as it is either extrapolating or interpolating between faraway points in a region of the parameter space which may not be accessible by the system at all.

The Euclidean distance Δ_{min} from a user input to the nearest neighbor in the training data is calculated as:

$$\Delta_{\text{min}} = 4 \times \min \left(\sqrt{\sum_{x=(N',W,W',\varepsilon)} (\hat{x}_{\text{train}} - \hat{x}_{\text{user}})^2} \right) \tag{3}$$

where the subscripts "*train*" indicates the training data and "*user*" the user input. The hat symbol indicates normalized data. The prefactor 4 is arbitrary and chosen to have a value of $\Delta_{\text{min}} = 1$ for a user input that is $\sigma/4$ away from the training data with σ the standard deviation a given feature. A value of $\Delta_{\text{min}} = 0$ means that the user input coincides with a point in the training data. A sketch explaining the concept of this measure is shown in Fig.7 in a 2D space for the sake of simplicity. It exhibits the training data inputs as blue points and a scan of the user inputs with one fixed input as pink points. A user input crossing the bulk of the training data is considered "reliable" — in the sense that the model has been trained on similar data — while a user input far from the training data is likely to give unreliable predictions.

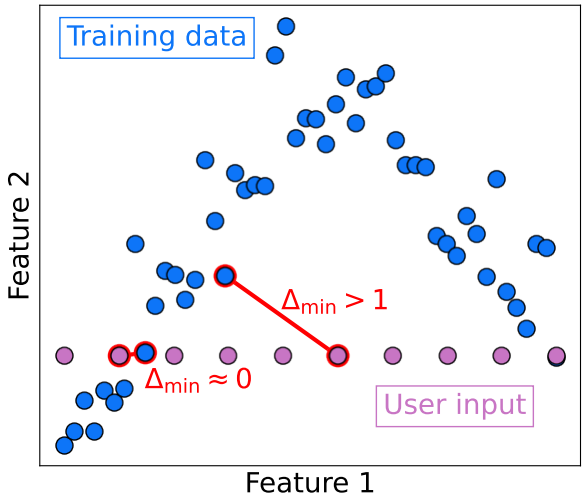

Figure 7. Visual representation of the nearest neighbor from user input to the training data as a measure of confidence of the model prediction.

To smooth the potential effect of rare outliers, the measure is taken as the average of k-nearest neighbors. The value of $k = 30$ is chosen, as it is the conventional number for a representative sample size as stated by the central limit theorem.

For the training datasets in this study — which typically have a number of individual data points $N \approx 10^7$ and a number of features $M = 4$ — it takes around

$\sim$ 1s to get the k-nearest neighbors for a single user input on a NVIDIA A100 GPU (i.e. fairly high-end at the time of writing). The number of operations — using the efficient NUMPY library — scales as $O(N \times M)$, which is probably not prohibitive for future studies with larger datasets. However, this overhead — combined with the requirement of a GPU to achieve fast computations — may be a limitation for some users.

In the next section, this measure is used to interpret the model predictions for the turbulent particle flux and Reynolds stress.

7. Results

In this section, some applications of the trained neural network are presented.

First, a safety check is performed to ensure that the neural network is able to predict the density gradient impact on the turbulent particle flux where the results are already known. Then, the focus is set on the vorticity gradient contribution to the turbulent particle flux and the anti-viscous behavior of the Reynolds stress.

The reason to suspect these two effects is based on the general concept in thermodynamics stating that fluxes are linked to forces [40, 41] such that

$$\boldsymbol{\Gamma_x} = f(\nabla_x \boldsymbol{F}) \tag{4}$$

where $\Gamma_x = (\Gamma, \Pi)$ is a radial flux vector — Π being a poloidal momentum flux — and $\boldsymbol{F}$ is a collection of field, e.g. density, electric potential, temperature, etc ... The function f is generally modeled as a polynomial or linear combination for reduced modeling. In the context of the Hasegawa-Wakatani framework, the fluxes are $\Gamma_x = (\Gamma, \Pi)$ as Π is a poloidal momentum flux and the forces are $\nabla_x \boldsymbol{F} = (N', N'', \dots, W, W', \dots)$ as W is a gradient of poloidal velocity. Numerically, all orders of derivative cannot be kept and, in this study, the choice has been made to focus on the effect of the vorticity and its derivatives (note that the vorticity curvature was initially kept but its impact on flux has been assessed as negligible, see Appendix C).

In particular, the vorticity gradient contribution to the turbulent particle flux is generally neglected. In fact, to the author's knowledge, very few studies [10, 42] report such an effect, although some hints can be found in [43–46] reporting relationships between density and vorticity, and between flux and vorticity.

In addition, the antiviscous behavior of turbulence can also be studied. Indeed, the mean poloidal velocity V_y conservation reads

$$\frac{\partial V_\mathrm{y}}{\partial t} = -\Pi' \tag{5}$$

with Π' the flux-averaged Reynolds stress divergence. Knowing that $W = \partial_x V_y$ and assuming a diffusive form of the right-hand side, i.e. $\partial_t V_y \propto W'$, the Reynolds stress acts as an anti-diffusion if $\Pi = CW$ with a constant $C > 0$. This would indicate that turbulence generates a mean poloidal flow, i.e. zonal flows. Note that most linear and

quasilinear models predict a vanishing C, indicating that the zonal flow generation is a non-linear effect.

The following examples are then a way to test the neural network capability to capture such effects and observe the relative impact of the interchange and drift waves instability. The interpretation of these results, along with a more comprehensive in-depth study of the physics accessible by this data-driven surrogate model is left for future work.

7.1. Density gradient contribution to the turbulent particle flux

This first example is meant to ensure that the neural network is able to predict a well-known relationship between the density gradient and the turbulent particle flux. It is expected that the turbulent particle flux Γ increases with the density gradient N' and the turbulent intensity, here represented by the proxy of the potential enstrophy ε. The conventional Hasegawa-Wakatani equations, i.e. without interchange, present a reflection symmetry ensuring that the turbulent particle flux is anti-symmetric about $N' = 0$. With the presence of interchange, this symmetry is broken and one expects a positive density gradient to be stabilizing, i.e. to reduce the turbulent particle flux. These properties are clearly visible in the raw data of each dataset, as shown in Fig.8 where each point is represented in the $(N', \Gamma, \varepsilon)$ space.

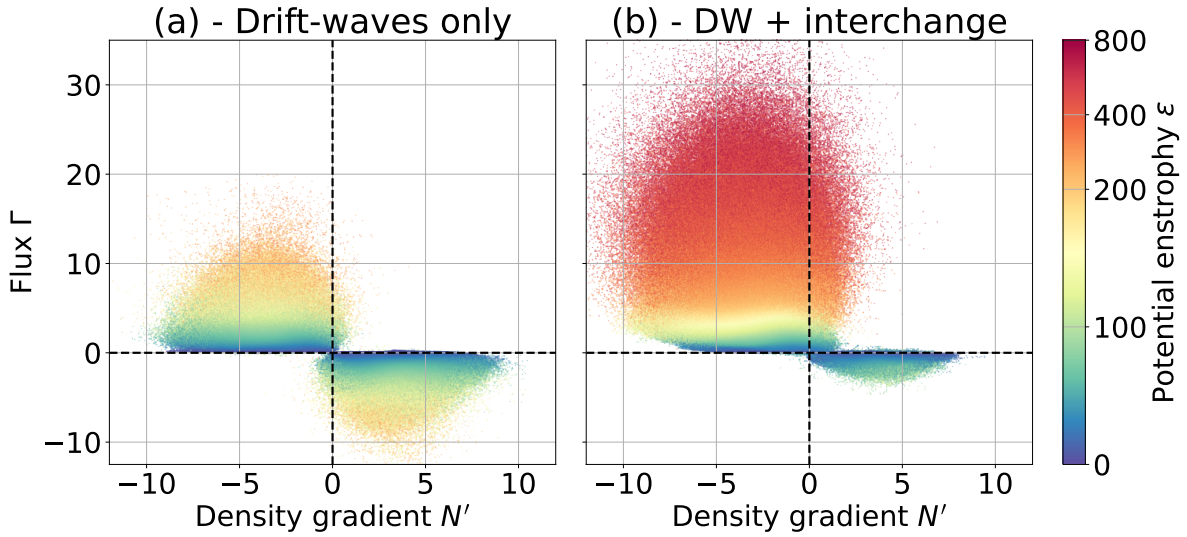

Figure 8. Visualisation of whole dataset distribution in the $(N', \Gamma, \varepsilon)$ space. Left plot: dataset with only drift waves. Right plot: dataset with both drift waves and interchange.

The neural network is able to capture these properties as shown in Fig.9a where the predicted turbulent particle flux Γ is plotted against the density gradient N' for different values of the potential enstrophy ε and for fixed values of the vorticity and its derivatives $W = W' = 0$. The inner color of the curves indicates the minimum distance

between the input asked to the neural network and the training dataset, as defined in Section 6. Green indicates that the user input is close to the training dataset, while red indicates that the user input is probably inaccessible by the system and thus is not physically relevant.

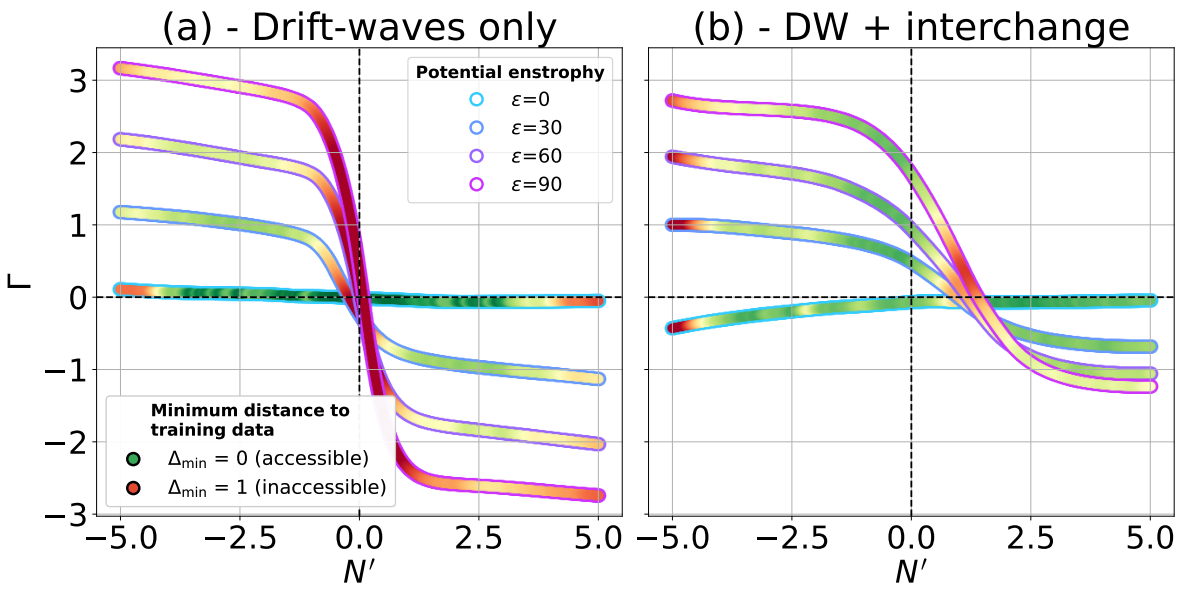

Figure 9. Predicted flux Γ from the neural network model vs. density gradient N' for different values of the potential enstrophy ε at $W = W' = 0$. Left plot: dataset with only drift waves. Right plot: dataset with both drift waves and interchange.

For the model trained on the dataset with only the drift waves instability, one then observes that regions of both high turbulent intensity ε and low-density gradient N' are not accessible by the system. This is expected as N' is a drive of turbulence, thus high-density gradients are associated with high turbulent intensity. The vanishing value of the flux at zero turbulent intensity is well recovered as the linear phase of the simulation — initialized with a wide range of density gradient N' — is included in the training dataset.

For the model trained on the dataset with both drift waves and interchange, where predictions are shown in Fig.9b, one can now observe an asymmetric turbulent particle flux about $N' = 0$. Compared with the model trained on the dataset with only drift waves, for a given potential enstrophy ε, the flux at a positive density gradient is lower than the flux at a negative density gradient. This is reminiscent of the stabilizing effect of positive density gradient with the interchange instability.

At low positive density gradient and high potential enstrophy, the surrogate model is not reliable as the user input is far from the training dataset and thus the predictions of positive flux are not physical. Also, regions of both high turbulent intensity and high positive density gradient are not accessible as the magnetic curvature is stabilizing.

Finally, for both datasets, the flux appears to bear a linear dependence on the potential enstrophy ε, i.e. with the turbulent intensity, which is in line with quasilinear

theory [32].

This example is a good omen for the capability of the neural network to capture physically relevant patterns of the system. The next examples focus on less-known physics and where such a data-driven approach can bring new insights.

7.2. Vorticity gradient contribution to the turbulent particle flux

One can now address the vorticity gradient contribution to the turbulent particle flux. Once again, the whole data associated with each dataset is shown in Fig.10 where each point is represented, here in the (N', Γ, W') space. One can still appreciate the density gradient impact on the flux in each dataset as in the previous example, but now a pattern of the vorticity gradient impact on the flux is also clearly visible.

The flux prediction of the data-driven model for the dataset with only drift waves turbulence is shown in Fig.10a for different values of the vorticity gradient W' at a fixed potential enstrophy $\varepsilon = 40$ and vorticity $W = 0$. As reported in [10], one can observe a shift of the turbulent particle flux proportional to the vorticity gradient regardless of the density gradient. When comparing this trend with the dataset with both drift waves and interchange, displayed in Fig.10b, the same pattern is observed and superimposed with the stabilization at positive density gradient. Consequently, the interchange instability does not seem to affect the linear vorticity gradient contribution to the flux.

Of course, the striking pattern in the data in Fig.10 is by itself a solid argument for the presence of a non-negligible effect of the vorticity gradient contribution to the turbulent particle flux. However, it is worth mentioning that this pattern is only apparent thanks to careful data visualization (with some fine-tuning of the symmetric logarithmic scale of the color bar). Counter-intuitively, it is actually the neural network prediction that first revealed the presence of such a pattern, and the data visualization was obtained afterward.

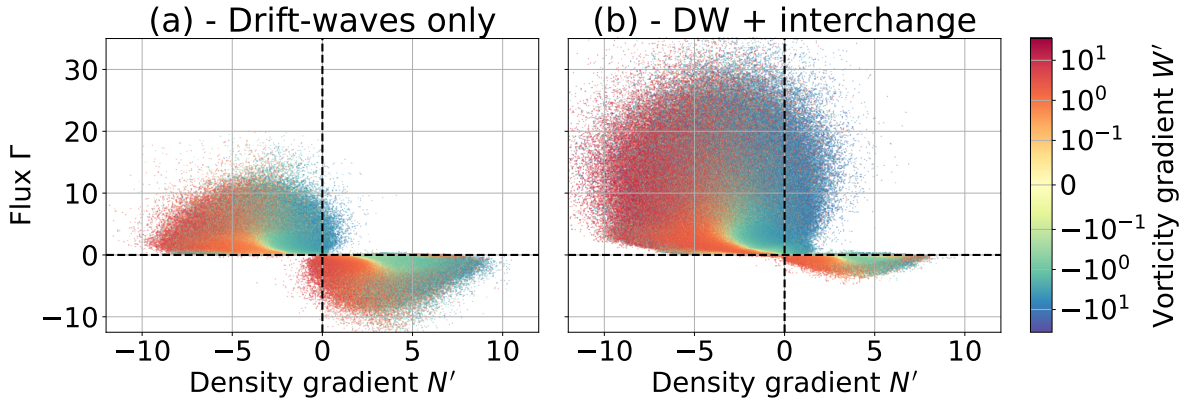

Figure 10. Visualization of whole dataset distribution in the (N', Γ, W') space. Left plot: dataset with only drift waves. Right plot: dataset with both drift waves and interchange.

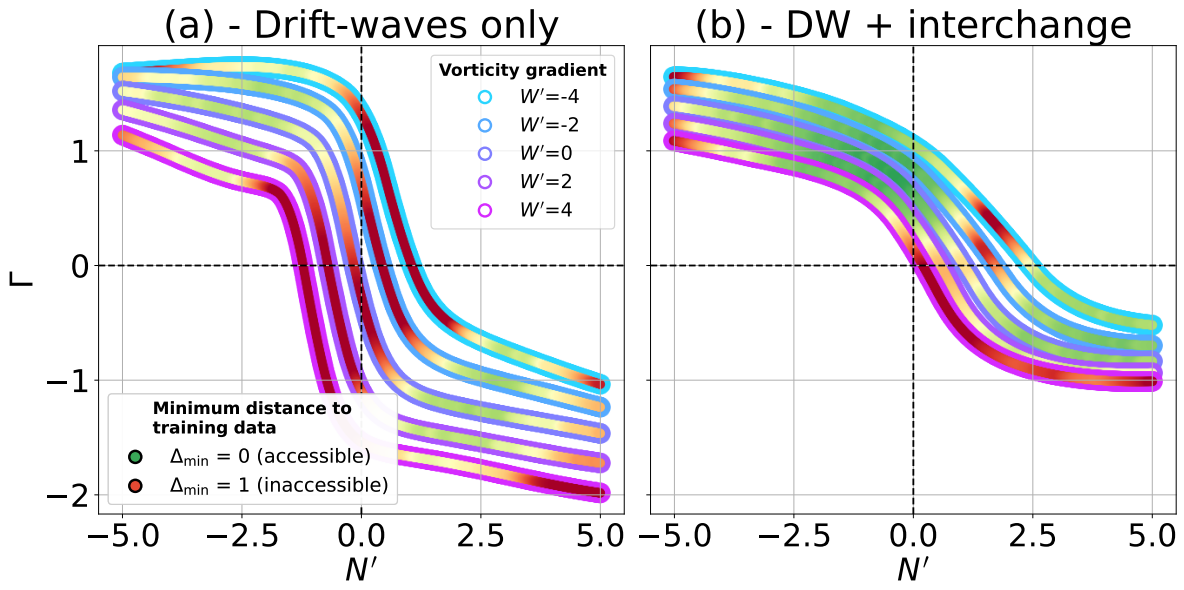

Figure 11. Predicted flux Γ from the neural network model vs. density gradient N' for different values of the vorticity gradient W' at $W = 0$ and $\varepsilon = 40$. Left plot: dataset with only drift waves. Right plot: dataset with both drift waves and interchange.

7.3. Anti-viscous behavior of the Reynolds stress

The final example is about the anti-viscous behavior of the Reynolds stress. Anti, or negative, viscosity plays a crucial role in the generation of zonal flows which are themselves critical for turbulence regulation. Negative viscosity is believed to arise from a modulational instability of a drift waves spectrum or from an inverse energy cascade, where energy is transferred from small-scale fluctuations to large-scale flows, driving these zonal flows [27, 47]. Zonal flows can suppress transport through their radial shear while pumping energy from the turbulence [27]. In the collisionless limit, zonal flows growth is limited by nonlinear damping mechanisms [27] — that can stem inter-alia from a tertiary instability, wave-packet scattering [47] or trapping [48, 49] — that eventually saturate their amplitude, forming a self-regulating feedback loop [50].

Following the logic of Eq(5), one expects an antiviscous behavior of turbulence if $\Pi \propto W$.

The whole data associated with each dataset is shown in Fig.12 where each point is represented, here in the (W, ε, Π) space. While noisier than the previous example, a clear pattern can be observed in both datasets with an anti-symmetric behavior of the Reynolds stress with respect to the vorticity W at a given potential enstrophy ε. This is expected from the structure of the Hasegawa-Wakatani equations Eqs(1), even with the presence of interchange. Indeed, these equations present three reflection symmetries when there is no interchange — i.e. $g = 0$ — and only one remains when accounting for the magnetic curvature such that they are invariant under the transformation $(x, n, \phi) \to (-x, -n, -\phi)$. This symmetry imposes that $(W, \Pi) \to (-W, -\Pi)$ in the data, which is what is observed in Fig.12. Note that interchange can still impact the

values of the Reynolds stress and vorticity, as long as the symmetry is preserved. As also somewhat noticeable in Fig.8 and Fig.10, the superimposition of the interchange instability to the drift waves turbulence significantly increases the ranges of potential enstrophy ε and vorticities W accessible by the system.

The Reynolds stress prediction from the data-driven model for the dataset with only drift waves turbulence is shown in Fig.13a as a function of vorticity for different values of the potential enstrophy ε at a fixed density gradient $N' = -2$ and vorticity gradient $W' = 0$. Note that the y-axis does not match the data in Fig.12 as in previous examples. It is observed that the Reynolds stress linearly increases with the vorticity in the vicinity of $W = 0$, which is compatible with an anti-viscous behavior. At a certain threshold in $|W|$, the Reynolds stress saturates and then decreases. This is consistent with a hyperviscous saturation stemming from non-linear damping mechanisms [27]. Comparing this trend with the dataset with both drift waves and interchange, displayed in Fig.13b, the same pattern is observed. However, for given values of the potential enstrophy ε, the Reynolds stress is lower in the presence of interchange. This leads to a moderate decrease in the anti-viscous behavior of the Reynolds stress. One can also notice that, for both datasets, almost all predictions obtained with the neural network are fairly close to the training data. However, at large vorticities, the neural network is probably extrapolating and the predictions are less reliable. This is especially striking for the prediction at $\varepsilon = 0$ where the Reynolds stress should vanish regardless of the vorticity.

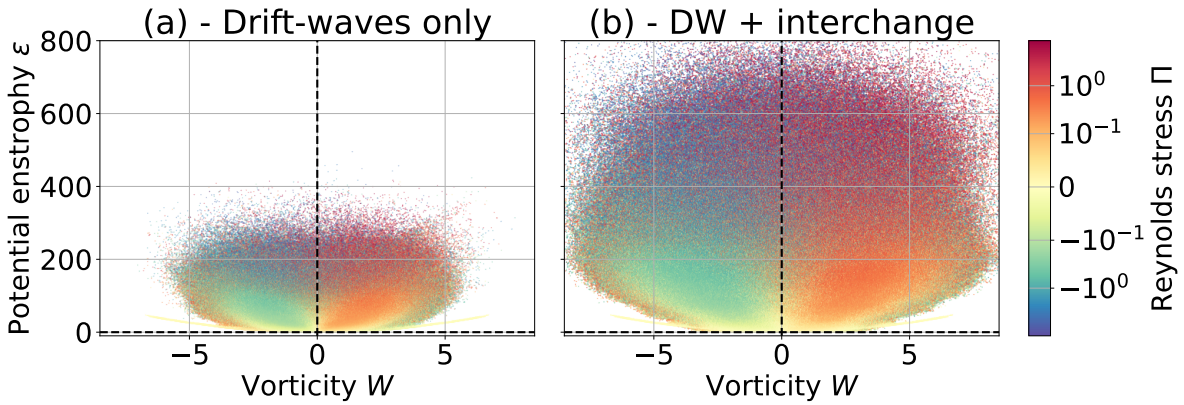

Figure 12. Visualisation of whole dataset distribution in the (W, ε, Π) space. Left plot: dataset with only drift waves. Right plot: dataset with both drift waves and interchange.

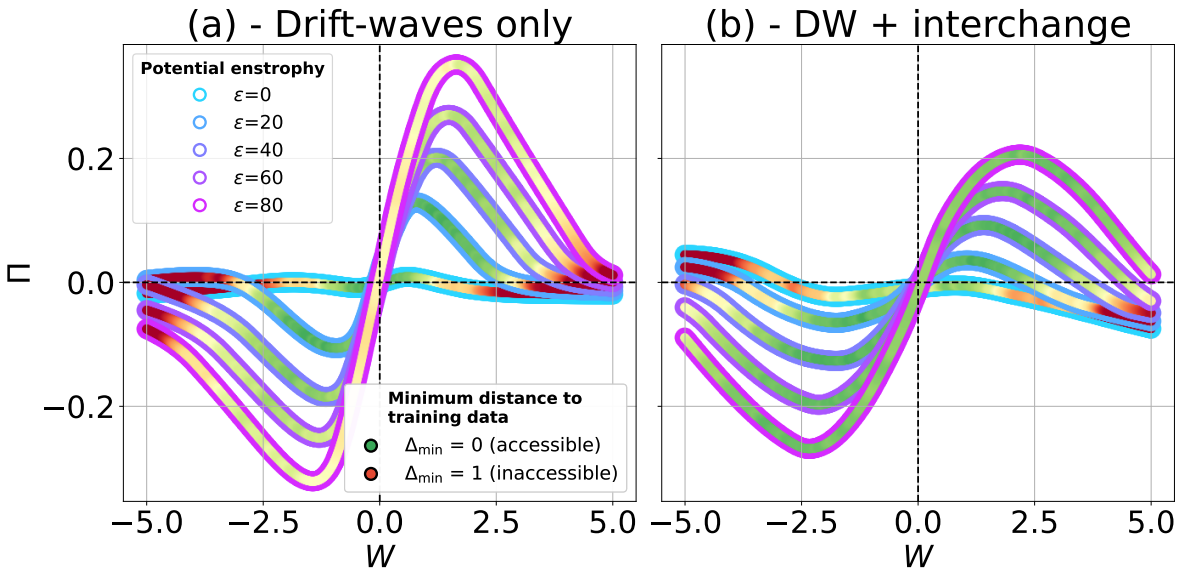

Figure 13. Predicted Reynolds stress Π from the neural network model vs. vorticity W for different values of the potential enstrophy ε at $N' = -2$ and $W' = 0$. Left plot: dataset with only drift waves. Right plot: dataset with both drift waves and interchange.

8. Discussion and conclusion

Data-driven surrogate models for turbulent systems appear to be a promising approach to get fast approximations of physical quantities that are expensive to compute. They can be used to discover patterns in the data that can lead to new insight into the physics of the system. With these models, it is fairly easy and fast to explore a multi-dimensional parameter space and to identify regions of interest, e.g. to motivate further high-fidelity simulations.

In this work, machine learning regression algorithms — based on neural networks — have been used to predict the flux-surface averaged particle flux and the Reynolds stress in a simplified 2D edge tokamak turbulence described by the Hasegawa-Wakatani equations. Two datasets have been generated, one for drift waves driven turbulence only, and one with the addition of the interchange instability stemming from the magnetic curvature. This additional instability further reduces the symmetries of the governing equations, making the problem more complex and closer to realistic edge plasma turbulence. These datasets consist of data from 50 simulations each where the background density gradient and initial electric potential profiles have been varied. The features chosen to explain the output of interest are the vorticity and its gradient, the density gradient and the potential enstrophy — a proxy for the turbulent intensity — all averaged over the flux surface. The choice of such features restrains the study to fundamental physics, compared to an applicative approach that would constrain the features to engineering parameters and/or experimentally accessible quantities. The reason for this choice is 1) these quantities evolve in time so a single simulation can

provide a lot of data, whereas linking the output to engineering parameters typically requires changing the parameter of control of the simulation, and 2) interesting physics can be assessed on a well-known set of equations.

While the dimensionality of the datasets is relatively low — 4 features and 2 outputs — the machine learning aspect of this study is not trivial due to the specificities of the problem at hand.

First, the data from simulated turbulence is highly fluctuating — especially for the Reynolds stress — and one wants the model to capture the mean behavior, i.e. excluding the fluctuations. This makes the assessment of the model performance challenging, as the error between the surrogate model predictions and the true data (that includes fluctuations) is then not only a measure of the model's performance but also of the fluctuations' level. To tackle this issue, a large amount of different models have been trained with different hyperparameters and architectures. The results showed that each model performed similarly, which is a good indicator that the patterns in the data are quite clear and that the model's training is robust as it does not depend on the parameters specific to the training process.

Second, when using the data-driven surrogate model, one must be cautious when providing a set of values for the features. Indeed, if a specific set of features is far from the values spanned in the training set, the model is either extrapolating or interpolating between faraway regions of the parameter space. In any regression problem, this would raise the question of the reliability of the predictions. In the context of a physical system such as turbulence, this is even more constraining, as asking for features outside of the dataset means that these values are not even accessible by the system in the first place (according that the dataset is representative of the whole range of accessible values). For multi-dimensional problems, even as low as 4 dimensions as in this study, choosing a valid set of values for the features is not trivial. For this reason, a figure of merit based on the distance between the input features and the training set has been devised to assess the reliability of the model predictions. This quantity allows to quickly identify if a set of features make sense or not from a physical point of view.

In addition, when selecting one model, some examples of predictions have been presented to illustrate the capability of such an approach to capture known physics and provide insight into less known physics. Note that a more comprehensive application of the data-surrogate model for a physics study is left for future work.

The first example illustrates the impact of the density gradient and on the particle flux for a given turbulent intensity. For the dataset with only drift waves driven turbulence, the particle flux Γ is found positive for negative density gradient N' and the expected reflection symmetry $(N', \Gamma) \rightarrow (-N', -\Gamma)$ is retrieved even though it was not enforced in the data nor in the neural network. The figure of merit indicates that the regions of simultaneous high turbulent intensity and low-density gradient are not accessible by the system, which makes sense as N' is the only drive for turbulence in that case. For the dataset also including the interchange instability, the reflection symmetry is broken and the particle flux amplitude is reduced for positive density gradient, which

is also expected as the interchange is stabilizing in this case. These observations have been used as a validation of the model to capture patterns in the data.

The second example allows to assess the vorticity gradient contribution to the particle flux, which is a less-known effect. For both datasets, a linear contribution of the vorticity gradient is found to the particle flux, which can be quite significant. Interestingly, the interchange instability does not appear to change the magnitude of this contribution, although it does significantly change the range of values accessible for the vorticity gradient.

Finally, the third example illustrates the antiviscous nature of the Reynolds stress at the origin of zonal flow generation. The surrogate model captures a positive linear trend of the Reynolds stress with the vorticity W at the vicinity of $W \sim 0$, corresponding to an antiviscous behavior, and also the saturation at high vorticity. This result is quite remarkable when considering the non-linear nature of zonal flow generation and that the Reynolds stress is a highly fluctuating quantity. The presence of interchange instability does not change the general trend, which is not surprising as the $(W, \Pi) \rightarrow (-W, -\Pi)$ reflection symmetry is preserved even with this additional instability. However, it appears that the antiviscosity is slightly reduced in the presence of magnetic curvature. This observation remains unexplained and is left for future studies.

Careful considerations must nonetheless be taken in both the training process and the usage of the model to get physically relevant results. It should be mentioned that the scalability of this approach to more complex high-fidelity codes is challenging. To add to the typical issues of data generation cost, storage limits and training time, this study recommends the training of multiple models to ensure an invariance to the hyperparameters and the architecture of the final results. In addition, the figure of merit introduced in this work is based on a k-nearest neighbors approach. While it scales relatively well with the size of the dataset and number of dimensions, it does require GPUs for fast computation which adds a layer of complexity regarding the user-friendliness of the model.

The authors want to thank X. Garbet, P. Ghendrih and Ö. D. Gürcan for stimulating interactions. This research is supported by the National Research Foundation, Singapore. The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg). R. A. Heinonen received funding from the European Union's Horizon 2020 Program under grant agreement No. 882340.

Appendix A. Model structure

For a regression problem using neural networks with multiple outputs, there are multiple ways to choose the structure of the neural network. Some of the different structure options are shown in Fig.A1. The simplest one, labeled "Option A", is a single fully connected neural network with multiple neurons in the output layer, each corresponding to one output. However, in this case, the weights of all neurons of hidden layers are

trained to minimize the loss of both outputs simultaneously. When using such a model, the "choice" of the output is made when the information is passed from the last hidden layer to the output layer. Mathematically, the output layer receives a linear combination of the last hidden layer's neurons. In other words, the prediction of one output differs from another only by the weights of the output layer. The issue is that, if the nature of the output is very different, one output can dominate the other in the learning process. In other words, one can have the illusion that the model is able to predict both outputs, while in reality it is only predicting one of them. In that case, relying only on the weights of the output layer, which yields a simple linear combination of the information shuffled to capture the pattern of a single output, can lead to a poor prediction of the other output. In this study, this effect was clearly observed. While it is possible to adjust the loss of each output to balance the learning process, this makes the model quite specific to the problem at hand and such a tuning can be challenging, especially for future more multi-dimensional problems.

For these reasons, other model structures have been considered. In Fig.A1, two other options are shown. The "Option B" is straightforward: separate neural networks are trained to predict each output. This structure is robust as the information is not shared between outputs. However, it is computationally more expensive as it requires to train one neural network per output. This also prevents the network to learn some general patterns from the physics that could be shared between the outputs.

Finally, the "Option C" is, on paper, a compromise between the two previous options. It consists of training a single neural network with *shared* layers - that are trained on both outputs simultaneously - and *specific* layers - that are trained on each output separately. The idea, to be demonstrated, is that the shared layers will capture some general patterns, while the specific layers will capture the specific patterns of each output. However, preliminary tests on the problem addressed in this paper suggest that shared layers may not be helpful. Such ideas could nonetheless be explored in future works as "Option B" may not be scalable to more complex problems.

In this study, "Option B" was chosen for its robustness and numerical simplicity.

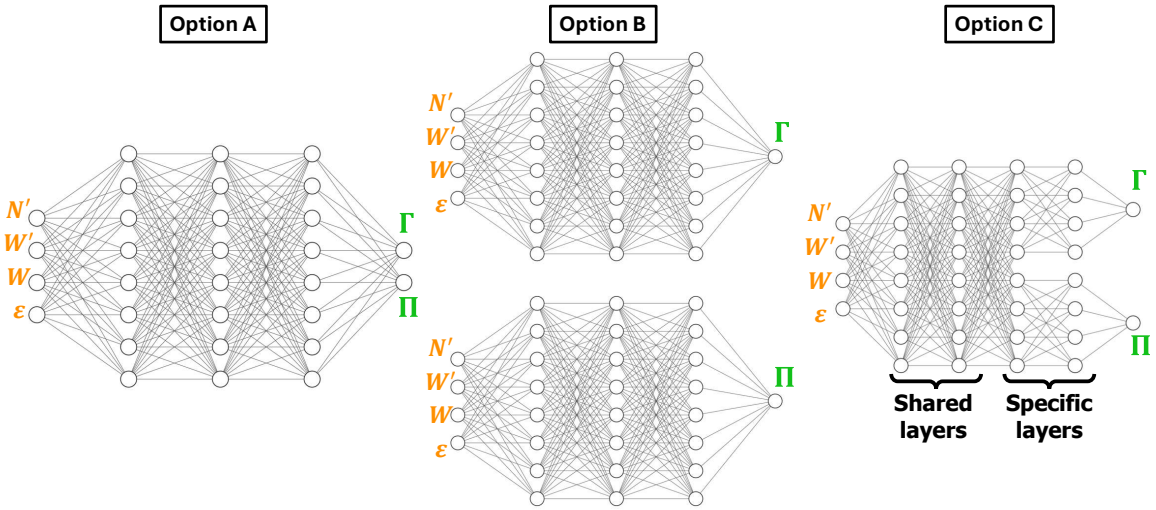

Figure A1. Different available options for the structure of the neural network with multiple outputs (here two only).

Appendix B. Multicollinearity

A straightforward way of assessing multicollinearity is to compute the correlation matrix of the features and outputs of the dataset. The correlation matrix associated with a dataset is a square matrix with dimensions equal to the number of variables (features and/or outputs) in the dataset. Its entries are the Pearson correlation coefficients $\rho(X, Y)$ between each pair of variables (X, Y) such that

$$\rho(X, Y) = \frac{\mathrm{cov}(X, Y)}{\sigma_X \sigma_Y} \tag{B.1}$$

where $\mathrm{cov}(X, Y)$ is the covariance between X and Y, and σ_X and σ_Y are the standard deviations of X and Y respectively. The correlation matrices for the variables $(N', W, W', \varepsilon, \Gamma, \Pi)$ are shown in Fig.B1 for both training datasets: with drift waves only and with both drift waves and interchange.

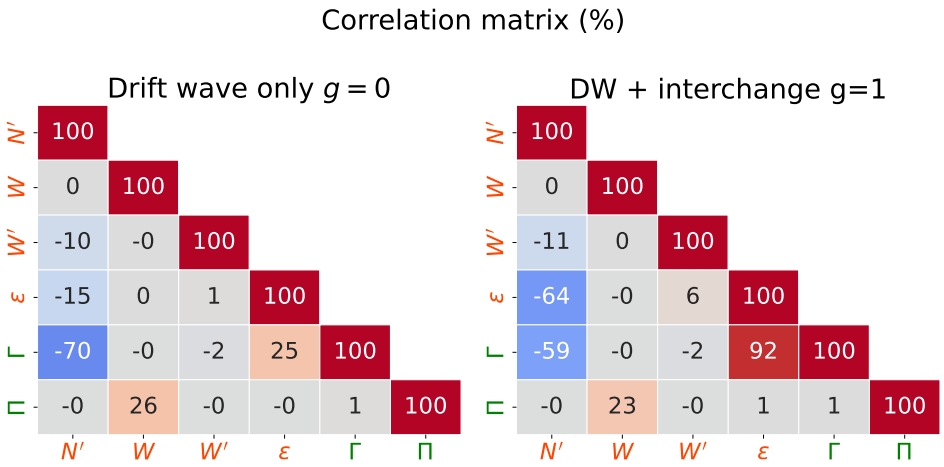

Figure B1. Correlation matrices of the features and outputs for both training datasets: with drift waves only $g = 0$ (left) and with both drift waves and interchange $g = 1$ (right).

For the dataset with drift waves only, the correlation matrix shows that the features are not highly correlated with each other, with the highest correlation being between the density gradient N' and the potential enstrophy ε at 15%. However, for the dataset with both drift waves and interchange, there is an important anti-correlation of 64% between N' and ε. From the physics point of view, this correlation is not too surprising as the density gradient drives the interchange instability. From the numerical point of view, this correlation could be a sign of multicollinearity which can ultimately lead to the contribution of the density gradient and the potential enstrophy to be mixed up in the model's prediction of the outputs. However, given the model predictions detailed in Section 7, it is *a posteriori* estimated that this is unlikely by comparing the prediction given by this dataset with the one obtained with the dataset with drift waves only, sharing similarities which are well explained by the physics of the system.

Interestingly, the outputs also appear to be highly correlated with some features. For both datasets, the flux Γ and density gradient N' are strongly correlated as expected from the Fick's law, and the Reynolds stress Π and vorticity W are also slightly correlated which is already a clue of the antiviscous nature of the Reynolds stress. For the dataset with both drift waves and interchange, a substantial correlation of Γ with ε of more than 90% is observed. This, as well as the correlation of N' and ε, suggests that the potential enstrophy could be an output of the model rather than a feature. For the sake of comparison of the two datasets, the potential enstrophy is kept as a feature in this model.

Appendix C. Relative importance of the features

The relative impact of the features on the outputs is an important point to consider. Indeed, even if a machine learning model is correctly specified and the metrics are satisfactory, it can still happen that one or several features have a negligible contribution

to the outputs. There are at least two ways to evaluate the relative importance of the features: the permutation feature importance [51] and the SHapley Additive exPlanations (SHAP) values [52]. Both are *a posteriori* analysis and differ significantly in approach and computational complexity.

On the one hand, permutation importance evaluates the impact of shuffling a feature's values on the model's outputs. In the permutation importance approach, the values of a feature in the training dataset are shuffled randomly, thus making the dataset wrong. These new features' values are then fed to the model, which will output predictions based on this wrong dataset. If the feature is important to explain the output, shuffling it will significantly decrease the model's performance, while unimportant features will have little to no effect. A measure of importance based on the residuals between the model's output and the true values is then calculated, giving a global measure of a feature's importance across the entire dataset.

On the other hand, SHAP values offer a more detailed insight by attributing a specific contribution of each feature to every individual prediction by considering all possible feature combinations. While more insightful - as it includes information on local contributions importance of each feature - SHAP values are also substantially more computationally intensive than permutation importance. Indeed, SHAP values require multiple model evaluations for each data point - depending on the number of features and the complexity of the model - while permutation importance requires just one model evaluation per feature per permutation (or a few if multiple permutations are used for robustness).

For this reason, the permutation importance is here chosen for this sensitivity analysis. This algorithm first calculate a metric for the "baseline" dataset - i.e. the dataset with the correct features' values - and then shuffles the values of each feature in the dataset and calculate the metric again. The difference between the two metrics is then a measure of the feature's importance: a low value means the model perform similarly with faulty features - i.e. the feature is not relevant to the model's prediction - while a high value means the model's performance is significantly impacted by the feature - i.e. the feature is important to the model's prediction. Here, the chosen metric is the coefficient of determination R^2 calculated as:

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

where the y_i are the true value, $\hat{y}_i$ the model's predictions, $\bar{y} \equiv \sum_{i=1}^{n} y_i$ is the mean of the true values and n is the number of points in the dataset. Note that, despite the name, the coefficient of determination can be negative if, on average, the model predictions are worse than $\bar{y}$ to explain the variance of the true values.

The permutation importance of features for the predictions of flux Γ and the Reynolds stress Π are shown in Fig.C1 for both datasets: with drift-waves only and with both drift-waves and interchange.

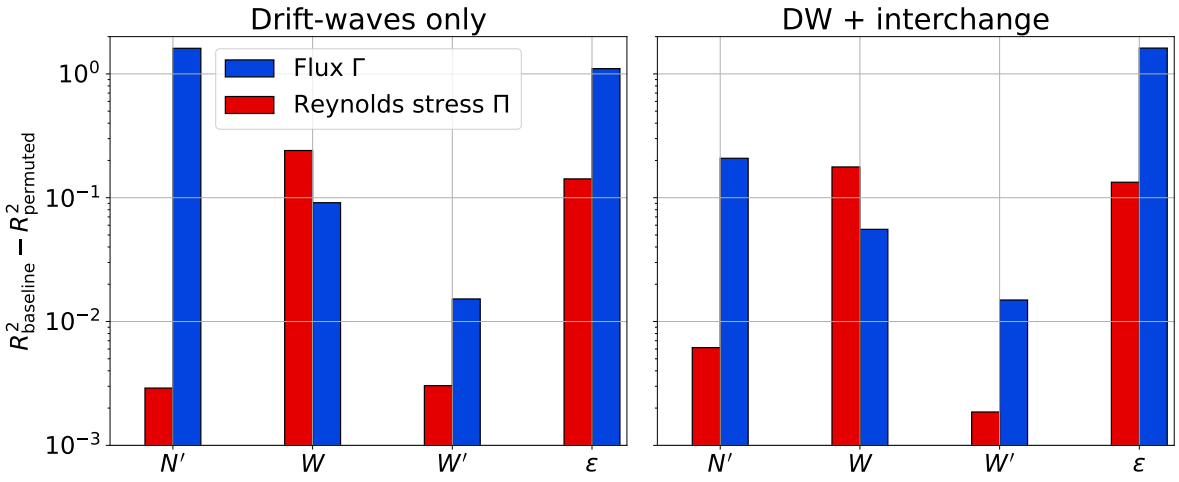

Figure C1. Permutation importance measure through the difference of the R^2 value calculated on the baseline dataset and datasets with shuffled features for the predictions of flux Γ and the Reynolds stress Π for both datasets: with drift-waves only (left) and with both drift-waves and interchange (right). High/low values mean that the feature is important/unnecessary in explaining the prediction. Note that this difference can be larger than 1 as the R^2 value can be negative when predictions are poor, typically for a permuted dataset.

It appears that, for both datasets, the turbulent flux Γ is mostly explained by the density gradient N' and the potential enstrophy ε. Interestingly, the vorticity W seems important to explain Γ even though no pattern has been found in the data, the model's prediction or the correlation matrix in Fig.B1. Furthermore, the contribution of the vorticity gradient W' is the lowest even though a clear pattern has been found in the data and the model's prediction as depicted in Section 7. For the Reynolds stress Π, the vorticity W and potential enstrophy ε are the most important features for both datasets. For this output, the density gradient N' and vorticity gradient W' appear to have a subdominant impact.

While the actual threshold to determine if a feature should be kept in the model or not is hard to determine, it should be noted that this work was initiated with another feature: the vorticity's curvature W''. The permutation importance analysis - typically returning $R^2_{\text{baseline}} - R^2_{\text{permuted}} < 10^{-3}$ for both outputs and datasets - with the addition of a handful of tests varying W'' with random values, showed that the feature had a negligible impact on the model's prediction.

An important closing remark for this appendix is the numerical cost of such procedures. Indeed, even if this method is numerically cheaper than computing the SHAP values, the number of calls to the model can be substantial, especially for datasets with a large number of features and outputs. Thus, applying such a method to larger datasets might be challenging, especially on low-end hardware.

References

[1] C. Bourdelle et al. "A New Gyrokinetic Quasilinear Transport Model Applied to Particle Transport in Tokamak Plasmas". In: *Physics of Plasmas* 14.11 (2007). DOI: 10.1063/1.2800869.

[2] C Bourdelle et al. "Core Turbulent Transport in Tokamak Plasmas: Bridging Theory and Experiment with QuaLiKiz". In: *Plasma Phys. Control. Fusion* 58.1 (2016). DOI: 10.1088/0741-3335/58/1/014036.

[3] R. E. Waltz et al. "A Gyro-Landau-fluid Transport Model". In: *Physics of Plasmas* 4.7 (1997). DOI: 10.1063/1.872228.

[4] J. E. Kinsey, G. M. Staebler, and R. E. Waltz. "The First Transport Code Simulations Using the Trapped Gyro-Landau-fluid Model". In: *Physics of Plasmas* 15.5 (2008). DOI: 10.1063/1.2889008.

[5] G.M. Staebler et al. "Successful Prediction of Tokamak Transport in the L-mode Regime". In: *Nucl. Fusion* 64.8 (2024). DOI: 10.1088/1741-4326/ad5a1b.

[6] Remi Lam et al. "Learning Skillful Medium-Range Global Weather Forecasting". In: *Science* 382.6677 (2023). DOI: 10.1126/science.adi2336.

[7] J J Hopfield. "Neural Networks and Physical Systems with Emergent Collective Computational Abilities." In: *Proceedings of the National Academy of Sciences* 79.8 (1982). DOI: 10.1073/pnas.79.8.2554.

[8] P. Ghendrih et al. "Generation and Dynamics of SOL Corrugated Profiles". In: *J. Phys.: Conf. Ser.* 1125 (2018). DOI: 10.1088/1742-6596/1125/1/012011.

[9] Philippe Ghendrih et al. "Role of Avalanche Transport in Competing Drift Wave and Interchange Turbulence". In: *J. Phys.: Conf. Ser.* 2397.1 (2022). DOI: 10.1088/1742-6596/2397/1/012018.

[10] R. A. Heinonen and P. H. Diamond. "Turbulence Model Reduction by Deep Learning". In: *Phys. Rev. E* 101.6 (2020). DOI: 10.1103/PhysRevE.101.061201.

[11] R. A. Heinonen and P. H. Diamond. "Learning How Structures Form in Drift-Wave Turbulence". In: *Plasma Phys. Control. Fusion* 62.10 (2020). DOI: 10.1088/1361-6587/abad02.

[12] Constantin Gahr, Ionuţ-Gabriel Farcaş, and Frank Jenko. "Scientific Machine Learning Based Reduced-Order Models for Plasma Turbulence Simulations". In: *Physics of Plasmas* 31.11 (2024). DOI: 10.1063/5.0225584.

[13] K. L. van de Plassche et al. "Fast Modeling of Turbulent Transport in Fusion Plasmas Using Neural Networks". In: *Physics of Plasmas* 27.2 (2020). DOI: 10.1063/1.5134126.

[14] B. Clavier et al. *A Generative Machine Learning Surrogate Model of Plasma Turbulence*. 2024. DOI: 10.48550/arXiv.2405.13232. arXiv: 2405.13232.

[15] A. Mathews et al. "Uncovering Turbulent Plasma Dynamics via Deep Learning from Partial Observations". In: *Phys. Rev. E* 104.2 (2021). DOI: 10.1103/PhysRevE.104.025205.

[16] I. Abramovic, E.P. Alves, and M. Greenwald. "Data-Driven Model Discovery for Plasma Turbulence Modelling". In: *J. Plasma Phys.* 88.6 (2022). DOI: 10.1017/S0022377822001192.

[17] Steven L. Brunton and Jose Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. 2nd edition. Cambridge: Cambridge university press, 2022. ISBN: 978-1-00-909848-9.

[18] Pankaj Mehta et al. "A High-Bias, Low-Variance Introduction to Machine Learning for Physicists". In: *Physics Reports*. A High-Bias, Low-Variance Introduction to Machine Learning for Physicists 810 (2019). DOI: 10.1016/j.physrep.2019.03.001.

[19] Ricardo Vinuesa and Steven L. Brunton. "Enhancing Computational Fluid Dynamics with Machine Learning". In: *Nat Comput Sci* 2.6 (2022). DOI: 10.1038/s43588-022-00264-7.

[20] Ryusuke Numata, Rowena Ball, and Robert L. Dewar. "Bifurcation in Electrostatic Resistive Drift Wave Turbulence". In: *Physics of Plasmas* 14.10 (2007). DOI: 10.1063/1.2796106.

[21] Akira Hasegawa and Masahiro Wakatani. "Plasma Edge Turbulence". In: *Phys. Rev. Lett.* 50.9 (1983). DOI: 10.1103/PhysRevLett.50.682.

[22] S. Benkadda, X. Garbet, and A. Verga. "Interchange Instability Turbulence Model in Edge Tokamak Plasma". In: *Contributions to Plasma Physics* 34.2-3 (1994). DOI: 10.1002/ctpp.2150340224.

[23] Yanick Sarazin. "Étude de La Turbulence de Bord Dans Les Plasmas de Tokamaks". These de Doctorat. Université Joseph Fourier (Grenoble ; 1971-2015), 1997.

[24] Akira Hasegawa and Kunioki Mima. "Pseudo-three-dimensional Turbulence in Magnetized Nonuniform Plasma". In: *The Physics of Fluids* 21.1 (1978). DOI: 10.1063/1.862083.

[25] Lord Rayleigh. "On Convection Currents in a Horizontal Layer of Fluid, When the Higher Temperature Is on the under Side". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 32.192 (1916). DOI: 10.1080/14786441608635602.

[26] R. Varennes et al. "Turbulent Relaxation Patterns in SOL Plasma". In: *Plasma Phys. Control. Fusion* 66.10 (2024). DOI: 10.1088/1361-6587/ad705c.

[27] P. H. Diamond et al. "Zonal Flows in Plasma—a Review". In: *Plasma Phys. Control. Fusion* 47.5 (2005). DOI: 10.1088/0741-3335/47/5/R01.

[28] Wendell Horton. "Statistical Properties and Correlation Functions for Drift Waves". In: *The Physics of Fluids* 29.5 (1986). DOI: 10.1063/1.865667.

[29] Lev Davidovič Landau, Evgeniĭ Mikhaĭlovich Lifšic, and Евгений Михайлович Лифшиц. *Fluid Mechanics*. 2nd ed., 2nd English ed., rev. Course of Theoretical Physics v. 6. Oxford, England New York: Pergamon Press, 1987. ISBN: 978-0-08-033933-7.

[30] Arash Ashourvan. "How Mesoscopic Staircases Condense to Macroscopic Barriers in Confined Plasma Turbulence". In: *Phys. Rev. E* 94.5 (2016). DOI: 10.1103/PhysRevE.94.051202.

[31] Arash Ashourvan and P. H. Diamond. "On the Emergence of Macroscopic Transport Barriers from Staircase Structures". In: *Physics of Plasmas* 24.1 (2017). DOI: 10.1063/1.4973660.

[32] G. Staebler et al. "Quasilinear Theory and Modelling of Gyrokinetic Turbulent Transport in Tokamaks". In: *Nucl. Fusion* 64.10 (2024). DOI: 10.1088/1741-4326/ad6ba5.

[33] Ting Long et al. "The Role of Shear Flow Collapse and Enhanced Turbulence Spreading in Edge Cooling Approaching the Density Limit". In: *Nucl. Fusion* 64.6 (2024). DOI: 10.1088/1741-4326/ad3e15.

[34] Ting Long et al. "On How Structures Convey Non-Diffusive Turbulence Spreading". In: *Nucl. Fusion* 64.6 (2024). DOI: 10.1088/1741-4326/ad40c0.

[35] M. D. McKay, R. J. Beckman, and W. J. Conover. "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code". In: *Technometrics* 21.2 (1979). DOI: 10.2307/1268522. JSTOR: 1268522.

[36] Jonas Mockus. *Bayesian Approach to Global Optimization: Theory and Applications*. Dordrecht: Springer Netherlands, 1989. ISBN: 978-94-009-0909-0.

[37] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. 2016. DOI: 10.48550/arXiv.1511.07289. arXiv: 1511.07289 [cs].

[38] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. DOI: 10.48550/arXiv.1412.6980. arXiv: 1412.6980 [cs].

[39] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, 2015.

[40] L. D. Landau and Evgenij Mihailovič Lifšic. *Statistical Physics: Volume 5*. 3rd ed. Elsevier, 1980. ISBN: 978-0-08-057046-4.

[41] Kerson Huang. *Statistical Mechanics*. 2nd ed. New York: J. Wiley & Sons, 1987. ISBN: 978-0-471-81518-1.

[42] Hongxuan Zhu and I. Y. Dodin. "Wave-Kinetic Approach to Zonal-Flow Dynamics: Recent Advances". In: *Physics of Plasmas* 28.3 (2021). DOI: 10.1063/5.0043784.

[43] J Garcia et al. "A New Mechanism for Increasing Density Peaking in Tokamaks: Improvement of the Inward Particle Pinch with Edge $E \times B$ Shearing". In: *Plasma Phys. Control. Fusion* 61.10 (2019). DOI: 10.1088/1361-6587/ab31a4.

[44] V. Naulin, A. H. Nielsen, and J. Juul Rasmussen. "Dispersion of Ideal Particles in a Two-Dimensional Model of Electrostatic Turbulence". In: *Physics of Plasmas* 6.12 (1999). DOI: 10.1063/1.873745.

[45] V Naulin et al. "The Application of Passive Tracers for Investigating Transport in Plasma Turbulence". In: *Phys. Scr.* T122 (2006). DOI: 10.1088/0031-8949/2006/T122/016.

[46] Shrish Raj et al. "Argon, Neon, and Nitrogen Impurity Transport in the Edge and SOL Regions of a Tokamak". In: *Physics of Plasmas* 30.6 (2023). DOI: 10.1063/5.0142975.

[47] A. I. Smolyakov, P. H. Diamond, and M. Malkov. "Coherent Structure Phenomena in Drift Wave–Zonal Flow Turbulence". In: *Phys. Rev. Lett.* 84.3 (2000). DOI: 10.1103/PhysRevLett.84.491.

[48] X. Garbet et al. "Wave Trapping and $E \times B$ Staircases". In: *Physics of Plasmas* 28.4 (2021). DOI: 10.1063/5.0042930.

[49] Predhiman Kaw, Raghvendra Singh, and P. H. Diamond. "Coherent Nonlinear Structures of Drift Wave Turbulence Modulated by Zonal Flows". In: *Plasma Phys. Control. Fusion* 44.1 (2001). DOI: 10.1088/0741-3335/44/1/305.

[50] K Itoh et al. "Collisional Effects on Coherent Structures of Zonal Flows and Turbulent Transport". In: *Plasma Phys. Control. Fusion* 46.5A (2004). DOI: 10.1088/0741-3335/46/5A/037.

[51] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001). DOI: 10.1023/A:1010933404324.

[52] Scott M Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017.