Simulations of large-scale plasma systems are typically based on fluid approximations. However, these methods do not capture the small-scale physical processes available to fully kinetic models. Traditionally, empirical closure terms are used to express high-order moments of the Boltzmann equation, e.g., the pressure tensor and heat flux. In this paper, we propose different closure terms extracted using machine learning techniques as an alternative. We show in this work how two different machine learning models, a multi-layer perceptron and a gradient boosting regressor, can synthesize higher-order moments extracted from a fully kinetic simulation. The accuracy of the models and their ability to generalize are evaluated and compared to a baseline model. When trained from more extreme simulations, the models showed better extrapolation in comparison to traditional simulations, indicating the importance of outliers. We learn that both models can capture heat flux and pressure tensor very well, with the gradient boosting regressor being the most stable of the two models in terms of the accuracy. The performance of the tested models in the regression task opens the way for new experiments in multi-scale modelling.
Identification of closure terms using machine learning

I. INTRODUCTION

Fluid models are the keystone of macroscopic plasma modelling\textsuperscript{1}. Plasmas at large scales in all areas of application, from nuclear fusion to industrial plasmas and astrophysics, are treated with models based on the moments of the velocity distribution: e.g. density, momentum, temperature, pressure. Fluid models can be derived rigorously from kinetic theory by taking moments of the Boltzmann equation\textsuperscript{2}. The moments are computed as integrals in the velocity space weighted by a power of the velocity itself, where the density is linked to the zeroth order moment, momentum to the first, and temperature to the second.

Currently, the challenge lies in computing the closure\textsuperscript{2}. Closure is needed because the set of equations for the first \( N \) moments includes the moment \( N + 1 \) (and sometimes higher), which is missing from the description provide by the first \( N \) moments. Traditionally, closure relations express the missing terms as a function of the lower order moments. Many examples of the application of closure exists, and we refer the reader to the literature with examples ranging from simple equations of state (e.g. adiabatic\textsuperscript{4}, isothermal\textsuperscript{5}) to more complex descriptions of plasmas in strongly coupled, relativistic or quantum degenerate states\textsuperscript{6}.

All closures are typically derived in two ways\textsuperscript{2–9}. The first approach empirically determines the closure via experiments, similar to how equations describing the ideal gas law or the transport coefficient in fusion devices were discovered. Whereas the second approach relies on theory, particularly theoretical models summarizing kinetic processes, finite Larmor radius effects or Landau damping. The progress in this direction has accompanied the evolution of plasma science, with experiments as prominent as ITER, being based on the use of the latter approach\textsuperscript{10}.

However, the method comes under strain and can be criticized when applied to regimes where the hypothesis of the theoretical models or the range of experimental conditions are exceeded and the models are used to pioneer new applications. Moreover, in many situations, e.g. burning plasmas or warm dense plasmas, the complexity defies analytical treatment, and experiments are either inaccessible (astrophysics), difficult or even controversial (warm dense matter).

Recently, a new line of investigation is emerging: the use of machine learning (ML) tools to replace the analytical formulas used so far\textsuperscript{11}. ML has been used to automatically analyse PIC simulation data and extract the exact analytical expressions that govern their dynamics. The Sparse Identification of Nonlinear Dynamical systems (SINDy)\textsuperscript{12} method has been used to identify the different terms of the Vlasov equation governing a two-stream instability PIC simulation\textsuperscript{13}. More
Identification of closure terms using machine learning

recently, Ref.\textsuperscript{11} demonstrated that neural networks can be used to calculate the fluid closure terms in a specific type of 1D plasma problem. Using machine learning to find closure terms is also used in Computational Fluid Dynamics in order to incorporate effects of sub-grid turbulence\textsuperscript{14,15}.

Our idea is the following: using kinetic simulations as training sets, ML models can learn how to establish a relationship between the missing higher order moments and the available lower order moments.

We consider specifically the closure relations for the pressure tensor and the heat flux in a fluid moment description where only density, momentum and magnetic field are available. We use one of the most important physical phenomenon in plasmas: magnetic reconnection, the process by which the topology of the magnetic field lines changes. Reconnection not only plays a role in the Sun-Earth dynamo, but is a critical phenomena in the solar corona, nuclear fusion and many more types of high-energetic plasma\textsuperscript{16,17}. Modelling the reconnection process is a multi-scale problem, where a good closure is needed to correctly represent the underlying kinetic physics.

We proceed by first conducting highly resolved massively parallel kinetic simulations where the model represents all processes as accurately as possible. From these simulations, the necessary training data is extracted for the two different ML models. The models are deployed and compared to each other and to a baseline, linear regressor model.

The paper is structured as follows. Section\textsuperscript{II} discusses how the data is created from kinetic simulations, generated with a particle-in-cell method, and prepared for the ML models. Section\textsuperscript{III} discusses the different models used in the experiment and how they are evaluated. Section\textsuperscript{IV} discusses the results of the experiments.

\section{DATA}

The models aim to find a closure relation for the pressure tensor and heat flux. The data provided for the model originates from kinetic simulations of magnetic reconnection. The simulation, denoted in our work as "Double-Harris sheet experiment", is the same experimental setup described in Ref.\textsuperscript{18} to simulate magnetic reconnection, modelled with iPiC3D. Next, the moments extracted from the simulation are described, together with the exact data used for the ML experiments.
A. Double-Harris sheet Particle-in-Cell simulation

1. Simulation setup

The Double-Harris sheet is simulated with a Particle-in-Cell (PiC) method, specifically the implicit Particle-in-Cell (iPiC3D) method introduced in Ref. 19. The PiC method consists of first initializing the particles and fields, and then repeating a computational cycle on the configuration for a few ten thousand times.

The simulation is initialized as follows. First, the size of the simulation box is chosen and fixed to $L_x \times L_y \times L_z = 30d_i \times 40d_i \times 0.1d_i$, where $d_i$ is the ion inertial length. Each spatial direction has respectively $769 \times 1025 \times 1$ cells. As boundary conditions, the walls of the simulation are made periodic in every spatial direction.

Four species of particles are simulated, 2 electron species and 2 ion species. One set of ions and electrons are used as background as explained below, and the mass ratio between the ions and the electrons, $m_i/m_e$, is set to 256.

The electric field $E$ is initialized to zero. The magnetic field $B$ consists of a background component ($B_0$) and a perturbation component ($\delta B$), with the latter triggering the reconnection process. The magnetic field is defined as $B = B_0 + \delta B$. The background part is initialized using a double hyperbolic tangent profile that switches the direction of the X component of the magnetic field:

\begin{align*}
B_x &= B_{x0} \left( \tanh \left( \frac{y - \frac{L_y}{4}}{\delta} \right) - \tanh \left( \frac{y - \frac{3L_y}{4}}{\delta} \right) - 1.0 \right), \\
B_y &= B_{y0}, \\
B_z &= B_{z0},
\end{align*}

where $B_{x0}, B_{y0}$ and $B_{z0}$ are the constants, and $\delta$ is the thickness at half-height of the current layer, in this case set to $\delta = 0.5d_i$.

The magnetic field perturbation is focalized at the midpoint of the X direction and at the loca-
Identification of closure terms using machine learning

tion of each one of the two Harris layers in the Y direction with Gaussians:

\[ X_e = \exp \left( -\left( \frac{x - L_x}{\delta_x} \right)^2 \right), \quad (4) \]

\[ Y_{1/4} = \exp \left( -\left( \frac{y - \frac{L_y}{4}}{\delta_y} \right)^2 \right), \quad (5) \]

\[ Y_{3/4} = \exp \left( -\left( \frac{y - \frac{3L_y}{4}}{\delta_y} \right)^2 \right), \quad (6) \]

These focal points are used to excite the magnetic field perturbation that triggers the emergence of reconnection. The following are the equations used to impose the perturbation:

\[ \delta B_x = 2U_x B_{x0} X_e \cdot \left( -\left( \frac{y - \frac{L_y}{4}}{\delta_y} \right) Y_{1/4} + \left( \frac{y - \frac{3L_y}{4}}{\delta_y} \right) Y_{3/4} \right), \quad (7) \]

\[ \delta B_y = 2U_x B_{y0} \left( \frac{x - \frac{L_x}{2}}{\delta_x} \right) X_e \cdot (Y_{1/4} - Y_{3/4}), \quad (8) \]

\[ \delta B_z = 0.0, \quad (9) \]

with \( U_x = 0.4 \), \( \delta_x = 8\delta \) and \( \delta_y = 4\delta \).

The switch in direction imposed in the background magnetic field has to be balanced by a current, following Ampère’s law. We use one ion and one electron species to impose this charge neutral current. The following expression is derived from the application of Ampère’s law to the background magnetic field \( B_0 \):

\[ n_d = \frac{n_0}{4\pi} \left( \text{sech}^2 \left( \frac{y - \frac{L_y}{4}}{\delta} \right) + \text{sech}^2 \left( \frac{y - \frac{3L_y}{4}}{\delta} \right) \right), \quad (10) \]

with \text{sech} the hyperbolic secant and \( n_d \) the particle distribution of the background electron and ion species. The two remaining species, ion and electron, are the ones used to follow the evolution of the reconnection process. After the particles and fields are initialized, multiple thousands of computational cycles update the location of the particles and the values of the magnetic and electric field. The PIC algorithm used in this work is explained in detail in Ref 19.

2. Simulation results

Four simulations of the double Harris field experiment are generated. Each simulation has the initial conditions as described in section [II A 1] except for the guiding background magnetic field
Identification of closure terms using machine learning

$B_0z$. The choice of the guiding field, together with the name of each simulation, are displayed in Table I. The different background magnetic field ensures that, while the simulation is initially the same, different physics will arise in and around the region of reconnection.

| Simulation | BG0 [nT] | BGlow [nT] | BGmed [nT] | BGhigh [nT] |
|------------|---------|-----------|-----------|------------|
| $B_{0x}$   | 0.0097  | 0.0097    | 0.0097    | 0.0097     |
| $B_{0y}$   | 0       | 0         | 0         | 0          |
| $B_{0z}$   | 0       | 0.00097   | 0.0097    | 3          |

Each of these simulations is ran for 20 000 cycles, where each cycle advances the simulation in time with a time step equal to the inverse of the ion plasma frequency $\omega_{p,i}$: $\Delta t = 1/\omega_{p,i} = 0.0625$.

B. Extracting the moments

This section describes the data extracted from the simulations. The focus of the present work is on the heat flux and the pressure tensor. We will only use the data from the two electron species in this paper. The electrons have the highest spread in velocity, and carry the most thermal energy, and thus are the most relevant and interesting pressure tensor and heat flux. The electron pressure is also an essential term in the generalized Ohm’s law. The ions will be considered in a later study.

From each simulation, the moments of the velocity distribution are computed and extracted, together with the values of the electric and magnetic fields. Given the phase-space of the particle $p$ of species $s$ (position $x_p$, velocity $v_p$ and charge $q_p$), the charge density $\rho_s$ and local bulk velocity $u_s$ are defined on a uniform grid $x_g$ as follows.

$$\rho_{sg} \equiv \rho_s(x_g) = \sum_{p} N_s S(x_g - x_p), \quad (11)$$

$$u_{sg} \equiv u_s(x_g, t) = \frac{1}{\rho_{sg}} \sum_p N_s S(x_g - x_p) v_p, \quad (12)$$

where the summation is computed over $N_s$, the total number of particles of species $s$. $S(x_g - x_p)$ is an interpolation function based on b-splines used to transfer information from particles to the cells, as is typical in PIC codes. Here we use b-splines of order 1, also called ‘cloud in cell’ in PIC literature.
Identification of closure terms using machine learning

In the following steps, the gradients of the velocity and the magnetic field are computed. Only the gradient in the $x$ and $y$ direction is used, since the $z$-direction is only a single cell thick. Let $u_i^j$ be the $x$-direction of the bulk velocity in cell $i$. Then the gradients of the velocity and magnetic field are given by

$$\partial_j u_i^k = \frac{u_{i+1}^j - u_{i-1}^j}{2\Delta j}, \quad \partial_j B_i^k = \frac{B_{i+1}^j - B_{i-1}^j}{2\Delta j},$$

with $j = \{x,y\}$ and $k = \{x,y,z\}$ and $\Delta j$ the step size of the grid used to compute the simulation.

Given the role of the electric field in multiple processes, we included the component into our models. However, instead of giving the exact electric field computed by the kinetic simulation to the model, only the simplified Lorentz electric field, generated by the moving electrons, is provided. The electric field is given as

$$E_L = -v_e \times B.$$  (14)

We define the speed of a particle of species $s$ as $w_p = v_p - u_{sg}$ with respect to the mean local bulk velocity. The pressure tensor can be computed as

$$P_{sg} = \sum_{p} S(x_g - x_p) m_p w_p \otimes w_p,$$  (15)

where $w_p \otimes w_p$ is a tensor dyadic with 9 terms. The heat flux $q_s$ is defined as

$$q_{sg} = \sum_{p} S(x_g - x_p) m_p w_p^2 w_p,$$  (16)

and its components are defined as $q_s = (q_x, q_y, q_z)$.

Finally, we transform the pressure tensor from Cartesian coordinates $\{\hat{e}, \hat{g}, \hat{z}\}$ to the the frame of reference of the magnetic field $\{\hat{e}_\parallel, \hat{e}_{\perp,1}, \hat{e}_{\perp,2}\}$, a so-called field-aligned basis. The first vector is chosen parallel to the total magnetic field $B$. The second vector is chosen to be perpendicular to $B$ and the $z$-direction, placing it inside the plane of reconnection. The unit vectors of this basis are defined as

$$\hat{e}_\parallel = \frac{B}{\|B\|}, \quad \hat{e}_{\perp,1} = \frac{B \times \hat{z}}{\|B \times \hat{z}\|}, \quad \hat{e}_{\perp,2} = \hat{b} \times \hat{e}_{\perp,1}.$$  (17)

Now let $\{\hat{e}_\parallel, \hat{e}_{\perp,1}, \hat{e}_{\perp,2}\}$ be column vectors. Then the field-aligned pressure tensor is defined as

$$P_{sg}^B = \left(\hat{e}_\parallel \ \hat{e}_{\perp,1} \ \hat{e}_{\perp,2}\right) P_{sg} \left(\hat{e}_\parallel \ \hat{e}_{\perp,1} \ \hat{e}_{\perp,2}\right)^T.$$  (18)

In our experiments, the main focus is set on the performance of the models on both the diagonal and off-diagonal components of $P^B$. The diagonal components are given the names $p_\parallel, p_{\perp,1}$ and
Identification of closure terms using machine learning

\( p_{\perp,2} \), respectively, and are defined as follows:

\[
\begin{align*}
    p_\parallel &= \hat{e}_\parallel P_{sg} \hat{e}_\parallel^T, \\
    p_{\perp,1} &= \hat{e}_{\perp,1} P_{sg} \hat{e}_{\perp,1}, \\
    p_{\perp,2} &= \hat{e}_{\perp,2} P_{sg} \hat{e}_{\perp,2}.
\end{align*}
\]  \hspace{1cm} (19)

The off-diagonal components are given the names \( p_{\parallel,\perp,1}; p_{\parallel,\perp,2}; p_{\perp,1,\perp,2} \) and are defined as

\[
\begin{align*}
    p_{\parallel,\perp,1} &= \hat{e}_\parallel P_{sg} \hat{e}_{\perp,1}, \\
    p_{\parallel,\perp,2} &= \hat{e}_\parallel P_{sg} \hat{e}_{\perp,2}, \\
    p_{\perp,1,\perp,2} &= \hat{e}_{\perp,1} P_{sg} \hat{e}_{\perp,2}.
\end{align*}
\]  \hspace{1cm} (20)

Note that there are only three terms, since the pressure tensor is symmetric.

C. Creating the model input

To introduce a new ML model, it is assumed to select and preprocess the data and to properly select training, testing and validation sets. In this section, the process that creates the training, validation and test set is described.

1. Selecting the input data

Closure models of an \((N+1)\)th order moment must consists of linear combinations of the \(N\) lower-order moments or their derivatives. The derivatives of the magnetic and velocity fields are defined in Equation 13. Furthermore, given that the off-diagonal terms of the pressure tensor are computed from crossed terms of the plasma velocity, we also include the crossed terms of the magnetic and velocity fields. Finally, features that are highly linearly correlated to other features are removed from the input in order to speed up the training process, and make the models more interpretable. Following these steps, 21 input features are chosen:

- The magnetic field \( B \) (3) together with the gradients without \( \partial_x B_y \) (5)
- The velocity field \( V \) (3), its cross-terms \( V_x V_y \) (1) and the gradients without \( \partial_x V_y \) (5)
- The density \( \rho_e \) (1)
- The electric field \( E_L \) (3)
Identification of closure terms using machine learning

2. Generating the training and testing data sets

From each simulation, the defined list of features are extracted and the training, validation and test set are constructed as follows:

1. Ten snapshots are extracted from the simulation.
2. The particle and field data described in section II B are computed for each snapshot.
3. Snapshots are assigned to either the training, validation or test set, see Table II.
4. From each snapshot, a number of cells are sampled based on the agyrotropy (see Eq. 21).
5. From each sampled cell the input features of section II C 1, together with the heat flux and pressure tensor, are extracted.
6. The extracted data is aggregated into their respective training, validation and test set.
7. The training, validation and test set are normalized.

At the end of this process, there are 12 data sets, consisting of a training, validation and test data set for each simulation.

The first step extracts 10 snapshots at fixed intervals from a simulation. Each snapshot contains all the field and particle information, frozen at a chosen time step. As mentioned in section II A 2, each simulation is run for 20000 cycles. Ten snapshots are taken per simulation, with the first snapshot taken at cycle 10000 (to ensure reconnection has fully developed), and a new snapshot taken every 1000 computational cycles. A single snapshot consists of 923MB of data, so from each simulation 9.23GB of data are extracted, for a total of 36.92GB of data used in the training, testing and validation of the ML models.

The snapshots are assigned to a specific data set. Table III displays which snapshots are assigned to either the training, validation and test set. The same snapshot number is used for every each one of the cases described in Table I. For each simulation, the data from the respective snapshots are combined into three large data sets, from which points can be sampled to respectively create the training, validation and test set.

The next step samples cells from each snapshot based on the agyrotropy. Inspection of the data shows that the number of cells with interesting phenomena (such as reconnection) are far fewer than the number of cells where no relevant process is occurring at that time step. In order to ensure
Identification of closure terms using machine learning

TABLE II. The snapshots that are assigned to each data set. This is identical for each simulation.

| Data set  | Snapshots        |
|-----------|------------------|
| Training  | 10000, 13000, 15000, 18000 |
| Validation| 12000, 17000     |
| Test      | 11000, 14000, 19000 |

FIG. 1. A simplified visualization of the process that creates a training, validation and test set are created from a simulation. Orange slices represent the snapshots, the purple bar represents the computation of the input features from the snapshot data and the green bar represents the cell-sampling based on the agyrotropy. That the sampled data is not over-represented by the uninteresting cells, the agyrotropy is used to sample cells based on the influence of reconnection.

The agyrotropy is computed from the eigenvalues of the field-aligned pressure tensor, as computed in Eq \[18\]. Let \( \lambda_i, i \in \{1, 2, 3\} \) be the eigenvalues of the field-aligned electron pressure tensor \( P^B_{eg} \), with \( \lambda_1 > \lambda_2 > \lambda_3 \). Then the agyrotropy \( Ag \) is computed as

\[
Ag = 2 \frac{\lambda_3 - \lambda_2}{\lambda_3 + \lambda_2}.
\]

When the pressure tensor is gyrotropic, the agyrotropy will be close to 0. An agyrotropic pressure tensor (which occurs at regions close to or at reconnection) will have a larger agyrotropy.
Identification of closure terms using machine learning

Histogram of the agyrotropy for one of the snapshots of the simulations can be seen in Figure 2. The agyrotropy indicates when a cell would be of interest, and is therefore an excellent quantity to base our sampling on.

FIG. 2. Figure (a): the distribution of the values of the agyrotropy from a single snapshot of the simulation. Almost all of the cells in the snapshot have an agyrotropy between 0 and 0.2, showing the over-representation of ‘uninteresting’ regions. The red lines indicate the bin boundaries. The size of each bin increased exponentially, to account for the exponential decay in the number of cells with a high agyrotropy. Figure (b): an example of the cells sampled from the distribution shown in (a). Low agyrotropy values are seen in cells far from the reconnection site. High agyrotropy is seen at the reconnection sites.

Now cells can be sampled from each snapshot based on the agyrotropy of that cell. The agyrotropy distribution is split into a set of 5 bins. Because the distribution is right-sided, the size of the bins has been based on a logarithmic distribution, to ensure an even coverage of all the values. The bins are visualized in Figure 2 as vertical red lines. From each bin, a fixed number of cells are sampled. The sampling is done with replacement, meaning that points can be drawn from the data set multiple times, which ensures that each bin has the same number of points, as there are much fewer points with a high agyrotropy.
Identification of closure terms using machine learning

The number of cells sampled from each bin is decided by which data set the snapshot is assigned to. This is set to 6000, 4000, and 3000 cells from each bin if the snapshot is assigned to the training, validation and test set, respectively. This gives a training set containing 30,000 cells, the validation set containing 20,000 cells, and the test set containing 30,000 cells, out of the 788,225 cells per snapshot.

Finally, the constructed data sets are normalized. Normalizing has shown to give a positive effect on the stability and training speed of machine learning algorithms. The input features are normalized with the min-max approach to the interval \([0,1]\), by applying the following transformation to each feature:

\[
\tilde{X}_\text{tr} = \frac{X_\text{tr} - \min_{\text{tr}}}{\max_{\text{tr}} - \min_{\text{tr}}}, \quad \tilde{X}_\text{val} = \frac{X_\text{val} - \min_{\text{tr}}}{\max_{\text{tr}} - \min_{\text{tr}}}, \quad \tilde{X}_\text{test} = \frac{X_\text{test} - \min_{\text{tr}}}{\max_{\text{tr}} - \min_{\text{tr}}}. \tag{22}
\]

Here \(\max_{\text{tr}}\) and \(\min_{\text{tr}}\) refer to the maximum and minimum value of each feature in the training set, respectively. The output features are normalized by removing the mean and standard deviation from the data. This transformation is defined as

\[
\tilde{Y}_\text{tr} = \frac{Y_\text{tr} - \mu_{\text{tr}}}{\sigma_{\text{tr}}}, \quad \tilde{Y}_\text{val} = \frac{Y_\text{val} - \mu_{\text{tr}}}{\sigma_{\text{tr}}}, \quad \tilde{Y}_\text{test} = \frac{Y_\text{test} - \mu_{\text{tr}}}{\sigma_{\text{tr}}}. \tag{23}
\]

Here \(\mu_{\text{tr}}\) and \(\sigma_{\text{tr}}\) are respectively the mean and standard deviation of the training set. The maximum, minimum, mean and standard deviation of the training data are used to prevent any information leakage. Additionally, a histogram of the pressure tensor terms has shown that the diagonal terms have a heavy left-sided distribution, as seen in the first row of Figure 3. Before standardizing, the three target features are transformed to a normal distribution by applying a Box-Cox transformation. Notice that we can use a Box-Cox transformation because the pressure tensor components are all strictly positive. The Box-Cox transformation is defined as:

\[
BC(y, \lambda) = \begin{cases} 
\log(y) & \text{if } \lambda = 0 \\
\frac{y^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0
\end{cases} \quad \text{with } -5 < \lambda < 5. \tag{24}
\]

The following values for \(\lambda\) were used on each component of the pressure tensor:

\[
\bar{p}_\parallel = BC(p_\parallel, \lambda = -0.3), \quad \bar{p}_{\perp,1} = BC(p_{\perp,1}, \lambda = -0.45), \quad \bar{p}_{\perp,2} = BC(p_{\perp,2}, \lambda = -0.5) \tag{25}
\]

After applying this transformation to the diagonal components of the pressure tensor, all of the target features are standardized by normalizing the values and removing their mean. The effect of the transformation for the pressure components can be seen in Figure 3. When the model gives a prediction, the inverse transformation is applied to that prediction before the values are evaluated.
Identification of closure terms using machine learning

![Graphs showing distribution of pressure tensor diagonal components](image)

FIG. 3. Distribution of the pressure tensor diagonal components. Figure (a) shows a histogram of each feature before their transformation. Figure (b) displays the same features after applying a box-cox transformation and standardizing each feature.

### III. MODELS AND METHODS

Three ML models are considered in the experiments. Each model is trained to compute the pressure tensor $P_{\text{eg}}^B$ and the heat flux $q_e$, using lower-order moments, the electric and magnetic field, as described in Section II C 1, as input. The first model is a baseline model, the remaining models are two distinct machine learning algorithms. For details on the different concepts and models used, the interested reader is referred to Ref 25.

#### a. Baseline model: Linear Regression

A linear regression model is used as a baseline. This is based on Occam’s Razor, stating that given two plausible options, the simpler option is most often the correct option. The linear regression model is used to assess if a complex ML model is a better performing model. If the performance of the linear regressor is comparable to that of the ML model, we can conclude that the ML models are bad at modelling this specific problem.

By definition, the linear regression model aims to find a linear relation between the input features and output features by minimizing the sum of the mean squared loss. Let $y_i$ be the $i$th value of
Identification of closure terms using machine learning

a single observed features, \( i \in \{1, \ldots, N\} \), and \( x_i \) be the corresponding \( i \)th vector of input features. Then the linear regression model minimizes the following problem for each observed feature:

\[
\hat{y}_i = a \cdot x_i = a_0 + \sum_{j=1}^{n} a_{ij} x_{ij},
\]

\[
a = \arg \min_a \sum_{i=1}^{n} (a x_i - y_i)^2.
\]

We will use the linear regression method provided in the Python package Scikit-Learn\(^{28}\).

b. Machine learning models: Gradient Boosting  The first machine learning model is the Gradient Boosting\(^{29}\) algorithm. This algorithm takes a set of simple learners (in this case, decision trees\(^{30}\)) that, when combined, can model a complex problem that a single simple learner could never model. The model is trained by iteratively increasing the set of simple learners, where each new learner is trained to increase the accuracy on points in the data where the accuracy of the total learners is the lowest. The error on the data is computed using a differential error function, which allows the use of efficient optimization algorithms.

Given a model \( F_n \), containing a set of \( n \) learners at step \( n \) of the iteration, and an input vector \( x_i \), the model returns \( F_n(x_i) = \hat{y}_i \). The next step of the training process is to improve the model by adding a new learner \( g_n(x) \):

\[
F_{n+1}(x_i) = F_n(x_i) + g_n(x_i) = y_i.
\]

This equivalently states that the new learner is equal to

\[
g_n(x_i) = y_i - F_n(x_i).
\]

The boosting algorithm fits the new estimator to the residual between the observation and the modelled output. If the error function is chosen as the residual mean square error, gradient descent can be used to find the best minimum. In this paper, we will use the Histrogram Gradient Boosting Regressor\(^{31}\) (HGBR) model implemented in the Scikit-Learn package.

c. Machine learning models: multi-layer perceptron  The multi-layer perceptron (MLP) is the one of the most basic types of neural network based models in machine learning. The MLP consists of a set of building blocks, called perceptrons. Perceptrons are simple models, taking as input a vector \( x \), multiplying them by their internal weight vector \( w \) and adding a constant \( b \), and giving this as an input to the internal function. The internal function is typically a threshold-like function, such as a tanh or a logistic function. The output of a perceptron is a scalar, typically of
Identification of closure terms using machine learning

A typical perceptron activation function can look like the following function:

\[ y = \sum_{i=1}^{n} \tanh(w_i x_i + b) \]  

(30)

where \( n \) is the number of elements in the input vector \( x \). By creating layers of these perceptrons, that take as input the output of a previous layer of perceptrons, we can create a complex model, the so-called multi-layer perceptron. The number of layers, together with the number of perceptrons in each layer, are parameters that can be freely chosen, and are very much problem dependent. Also the internal function of the perceptrons in each layer can be chosen, and can have an effect on the accuracy of the model. The interested reader is referred to Ref. 25 for more information. The Python package PyTorch\(^a\) is used to implement the MLP model.

The code and experiments will be made available after acceptance on Github, at https://github.com/brechtlaperre.

A. Hyperparameter tuning

Each model has a set of hyperparameters, parameters used to fine-tune the model based on the data, that can be chosen. Because of the exhaustive number of parameters, the Python package Optuna\(^b\) is used to perform an extensive hyperparameter search for each model based on the training and validation set. Optuna applies two methods for finding the optimal hyperparameter. First it applies a combination of a relational sampling method, which uses the correlation among the parameters, and an independent sampling method, to pick promising hyperparameters. After picking a set of parameters, each one tested in a trial, and a pruning algorithm determines the best trials. Optuna makes use of Asynchronous Successive Halving Algorithm\(^c\), a state-of-the-art pruning algorithm, capable of running in parallel, greatly increasing its speed.

B. Training a model

Training a model \( M \) consists of changing the internal set parameters \( \theta \), such that, given the input parameters \( x \), the error between \( M(x, \theta) \) and \( y \) is minimized. By using two data sets, the training and validation set, the training algorithm repeats a cycle that iterates over the full training and validation set until convergence. Each cycle \( i \) first iterates over the full training set and computes the error between the model output \( M(x, \theta_i) \) and the known output \( y \). The error is used to change
the model parameters through gradient descent\textsuperscript{35}. The validation set is used to prevent over-fitting. Over-fitting is the process where the model starts memorizing the training set instead of learning the dynamics behind the training set. After every training cycle, the performance of the model on the validation set is computed. Once the model is trained, the test data set is only used to evaluate the model. The test data set is never used to change the parameters of the model.

C. Evaluation metrics

In order to evaluate the models, we define a set of metrics. Let \( M_i \) be the value predicted by the model at cell \( i \), and \( O_i \) the observed (true) value at cell \( i \), with \( i = 1, \ldots, N_c \), and \( N_c \) the total number of cells in the simulation. The absolute error is measured with the root mean square error (RMSE), defined in Equation \textsuperscript{31}.

\[
RMSE = \sqrt{\frac{1}{N_c} \sum_{i=1}^{N_c} (M_i - O_i)^2}.
\]

The Pearson correlation \( r \), defined in Equation \textsuperscript{32} is used to determine if the model prediction follows the same behaviour as the observations\textsuperscript{36}:

\[
r = \frac{\text{cov}(M, O)}{\sigma(M)\sigma(O)},
\]

where \text{cov} is the covariance and \( \sigma \) the standard deviation. The Pearson correlation has a range \( r \in [-1, 1] \), and indicates if \( O \) and \( M \) have a linear relation.

Finally, we define the prediction efficiency, also called the \( R^2 \)-score, which quantifies the model’s ability to predict the variations in the output features. The \( R^2 \) is defined as

\[
R^2 = 1 - \frac{\sum_{i=1}^{N_c} (M_i - O_i)^2}{\sum_{i=1}^{N_c} (O_i - \bar{O})^2},
\]

where \( \bar{O} \) is the average of the observational values. The range of the prediction efficiency is \( R^2 \in ]-\infty, 1] \). Notice that the value of \( R^2 \) is 1 when the prediction by the model is perfect. A prediction efficiency lower than 1 indicates that the model has trouble with predicting the variations, and that the model is worse than the average of the data at predicting the observations\textsuperscript{37}. 
IV. EXPERIMENT AND RESULTS

A. The experiments

The ML models are tested with four kinetic simulations, each with identical initial conditions except for the guiding background magnetic field. While all four simulations simulate a Double-Harris sheet with reconnection, the different guiding background magnetic field create distinctive physical phenomena in each simulation that affect the pressure tensor and heat flux. The models are evaluated on both accuracy of their prediction, as well as their extrapolation to simulations not included in the training.

Before continuing, we would like to highlight two definitions. First, take a model $M_A$ that has been trained on the training data set of simulation A. When the model $M_A$ predicts the outcome of the test set of simulation A, we call this a prediction on a known simulation. And secondly, when the model $M_A$ makes a prediction on the test set of simulation B, we call this a prediction on an unknown simulation.

The first experiment tests the ability to extrapolate of each model. Each model is trained on two out of the four simulations, and then evaluated on the test set of all the four simulations. We expect the models to perform well on the known simulations, and we want to evaluate how they perform on the unknown simulations. This experiment is performed twice, where the first run trains the models on simulation BGmed and BGlow, while the second run trains the models on simulation BG0 and BGhigh.

The second experiment evaluates the model’s ability to generalize. The models are trained on all of the four simulation training sets, and then evaluated on the test set of every simulation. The performance of the models are compared to the performance of the models of the first experiment. While we expect the models to always outperform any evaluation of an unknown simulation, we also want to test if the overall accuracy on the known simulations has increased by adding more information from all the different simulations.

The evaluation is computed using the metrics defined in Section III.C and placed into tables, one for the heat flux and and one for the pressure tensor. A visual linear comparison between the predicted values computed by the models and the values computed by the kinetic simulations is also provided. These figures plot the values predicted by the models against the values that the numerical simulation computed.
As mentioned previously in section II B, the models are trained to predict the electron pressure tensor and heat flux, as these have the largest influence in reconnection. The ions are omitted and will be discussed in a later research.

B. Experiment 1 - Train on BGlow and BGmed: Results

We start by training the three models on the training sets of simulation BGlow and BGmed. The models are evaluated on the test set of the simulations, and the results are displayed, for the heat flux, in Table III and for the pressure tensor in Table IV. The best performance for each feature in the unknown simulations, simulations BG0 and BGhigh, are highlighted in bold in each table.

We start by focusing on the evaluation of the heat flux, shown in Table III. First we find that the two machine learning models always outperform the linear regressor, in every simulation and on each metric, except in simulation BGhigh. Next we compare the two machine learning models with each other. Beginning with the BGlow simulation, the HGBR model slightly outperforms the MLP model, for each metric. The performance of the two ML models on the BGmed simulation, however, is very similar. For simulation BG0, the HGBR model is overall the best performing model. While the MLP model is slightly better for $q_y$, the HGBR model shows large improvements over the MLP, with $R^2 = 0.54$ for $q_x$ and $R^2 = 0.712$ for $q_z$, compared to $R^2 = 0.234$, $R^2 = 0.211$ for the MLP, respectively. Finally, for simulation BGhigh, the HGBR model is once again the best performing. The linear regressor does show a similar performance for $q_x$ in terms of the RMSE and $R^2$, but drops for $q_y$ and $q_z$. We remark the large RMSE and low $R^2$. The MLP model fails to capture the dynamics of the heat flux of BGhigh. This could be caused by overfitting of the MLP model on the training sets, combined with the extreme physics present in simulation BGhigh. This can also be seen in Figure 4, where the prediction of the heat flux by the MLP on simulation BGhigh diverges considerably from the truth.

The performances reached for the pressure tensor prediction are shown in Table IV. From the evaluation on BGlow and BGmed, it is clear that the HGBR is the best-performing model on the known simulations. Looking at the unknown simulations, starting with BG0, both the HGBR and MLP model are performing similar on the diagonal components $p_{\parallel}, p_{\perp,1}, p_{\perp,2}$, and outperform the linear regressor by a large margin. However, the linear regressor is more accurate for the off-diagonal components, with the ML models only having a better Pearson correlation $r$. The exception is the $p_{\perp,1,\perp,2}$-component, where the HGBR model shows the best performance.
Identification of closure terms using machine learning

TABLE III. Evaluation of the three models for the first experiment on the heat flux. The models were trained on the training sets of the BGmed and BGlow simulation (as indicated by ‘T’), and evaluated on the test set of each simulation. The best evaluations on the unknown simulations are highlighted. The best value for the RMSE is 0, while the best value for $r$ and $R^2$ is 1.

| Target | LinReg | | HGBR | | MLP | |
|--------|--------|--------|--------|--------|--------|
|        | RMSE   | $R^2$  | $r$    | RMSE   | $R^2$  | $r$    | RMSE   | $R^2$  | $r$    |
| BGlow - T |        |        |        |        |        |        |        |        |        |
| $q_x$  | 2.43e-09 | 0.323  | 0.639  | 1.41e-09 | 0.777  | 0.883  | 1.6e-09 | 0.712  | 0.845  |
| $q_y$  | 9.49e-10 | 0.0701 | 0.446  | 6.69e-10 | 0.556  | 0.746  | 7.6e-10 | 0.428  | 0.681  |
| $q_z$  | 3.54e-09 | 0.237  | 0.658  | 1.43e-09 | 0.876  | 0.936  | 1.91e-09 | 0.779  | 0.908  |
|        |        |        |        |        |        |        |        |        |        |
| BGmed - T |        |        |        |        |        |        |        |        |        |
| $q_x$  | 6.8e-09  | 0.362  | 0.609  | 3.48e-09 | 0.831  | 0.912  | 3.55e-09 | 0.825  | 0.909  |
| $q_y$  | 2.38e-09 | 0.299  | 0.555  | 1.35e-09 | 0.775  | 0.881  | 1.43e-09 | 0.747  | 0.868  |
| $q_z$  | 1.62e-08 | 0.314  | 0.573  | 8.42e-09 | 0.813  | 0.902  | 8.34e-09 | 0.816  | 0.907  |
|        |        |        |        |        |        |        |        |        |        |
| BG0    |        |        |        |        |        |        |        |        |        |
| $q_x$  | 2.1e-09  | 0.0488 | 0.589  | 1.46e-09 | 0.54   | 0.788  | 1.89e-09 | 0.234  | 0.773  |
| $q_y$  | 8.18e-10 | -0.267 | 0.361  | 6.86e-10 | 0.158  | 0.528  | 6.82e-10 | 0.167  | 0.45   |
| $q_z$  | 3.03e-09 | 0.0641 | 0.724  | 1.7e-09  | 0.712  | 0.879  | 2.82e-09 | 0.211  | 0.813  |
|        |        |        |        |        |        |        |        |        |        |
| BGhigh |        |        |        |        |        |        |        |        |        |
| $q_x$  | 5.15e-09 | 0.32   | 0.59   | 5.15e-09 | 0.321  | 0.656  | 1.53e-08 | -5     | 0.166  |
| $q_y$  | 2.26e-09 | 0.255  | 0.506  | 1.81e-09 | 0.521  | 0.729  | 7.44e-09 | -7.07  | 0.307  |
| $q_z$  | 3.59e-08 | 0.281  | 0.691  | 2.46e-08 | 0.662  | 0.823  | 4.27e-08 | -0.0199 | 0.683  |

Now looking at simulation BGhigh, the contrast with BG0 is evident. The MLP model fails to capture the dynamics of almost all the components, with negative values for $R^2$ on the diagonal components. The HGBR model, on the other hand, is much more capable to capture the dynamics of almost all the features. We note a decrease in $R^2$ compared to simulation BG0, where the performance of $R^2$ on $p_{\perp,1}$ went from 0.954 to 0.236, and on $p_{\perp,2}$ from 0.955 to 0.422. This is a clear indication that the physics of simulation BGhigh is much more difficult to capture for the models when they have only been trained on simulation BGlow and BGmed. This can also be
Identification of closure terms using machine learning

TABLE IV. Evaluation of the three models for the first experiment. The models were trained on the training sets of the BGmed and BGLow simulation (as indicated by ‘T’), and evaluated on the test set of each simulation. The best performance for each feature in the unknown simulations are highlighted. The best value for the RMSE is 0, while the best value for $r$ and $R^2$ is 1.

| Target    | LinReg | HGBR | MLP |
|-----------|--------|------|-----|
|           | RMSE   | $R^2$ | $r$ | RMSE   | $R^2$ | $r$ | RMSE   | $R^2$ | $r$ |
| **BGlows - T** |        |      |     |        |      |     |        |      |     |
| \( p\| \) | 2.34e-07 | 0.466 | 0.683 | 8.02e-08 | 0.955 | 0.977 | 1.5e-07 | 0.903 | 0.95 |
| \( p\perp,1 \) | 9.81e-07 | 0.48 | 0.699 | 4.87e-08 | 0.972 | 0.986 | 5.8e-08 | 0.958 | 0.979 |
| \( p\perp,2 \) | 0.00678 | 0.477 | 0.695 | 5.31e-08 | 0.974 | 0.987 | 7.74e-08 | 0.97 | 0.985 |
| \( p\parallel,\perp,1 \) | 1.27e-08 | -0.00322 | 0.0483 | 7.28e-09 | 0.669 | 0.828 | 7.66e-09 | 0.633 | 0.797 |
| \( p\parallel,\perp,2 \) | 1.92e-08 | 0.0711 | 0.314 | 9.35e-09 | 0.78 | 0.883 | 1.1e-08 | 0.693 | 0.852 |
| \( p\perp,1,\perp,2 \) | 8.46e-09 | 0.238 | 0.495 | 5.33e-09 | 0.697 | 0.836 | 6.31e-09 | 0.576 | 0.764 |
| **BGmeds - T** |        |      |     |        |      |     |        |      |     |
| \( p\| \) | 4.96e-07 | 0.284 | 0.534 | 7.25e-08 | 0.952 | 0.975 | 1.22e-07 | 0.914 | 0.956 |
| \( p\perp,1 \) | 7.39e-07 | 0.545 | 0.745 | 1.64e-08 | 0.962 | 0.981 | 1.68e-08 | 0.957 | 0.979 |
| \( p\perp,2 \) | 0.816 | 0.55 | 0.748 | 1.7e-08 | 0.956 | 0.978 | 2.68e-08 | 0.957 | 0.978 |
| \( p\parallel,\perp,1 \) | 1.12e-08 | 0.0234 | 0.158 | 5.47e-09 | 0.768 | 0.877 | 6.39e-09 | 0.683 | 0.847 |
| \( p\parallel,\perp,2 \) | 2.35e-08 | 0.277 | 0.532 | 9.35e-09 | 0.886 | 0.942 | 9.87e-09 | 0.873 | 0.941 |
| \( p\perp,1,\perp,2 \) | 4.83e-09 | 0.0851 | 0.362 | 3.17e-09 | 0.606 | 0.779 | 4e-09 | 0.374 | 0.63 |
| **BG0** |        |      |     |        |      |     |        |      |     |
| \( p\| \) | 2.64e-07 | 0.476 | 0.69 | \textbf{1.4e-07} | \textbf{0.858} | \textbf{0.952} | 1.53e-07 | \textbf{0.884} | 0.946 |
| \( p\perp,1 \) | 1.48e-06 | 0.482 | 0.707 | 7.46e-08 | \textbf{0.954} | \textbf{0.98} | \textbf{6.3e-08} | 0.939 | 0.97 |
| \( p\perp,2 \) | 0.0012 | 0.488 | 0.709 | 8.02e-08 | \textbf{0.955} | \textbf{0.983} | \textbf{7.66e-08} | \textbf{0.955} | 0.978 |
| \( p\parallel,\perp,1 \) | \textbf{9.58e-09} | \textbf{-0.0345} | 0.00434 | 1.19e-08 | -0.606 | 0.199 | 1.34e-08 | -1.02 | \textbf{0.213} |
| \( p\parallel,\perp,2 \) | \textbf{2.05e-08} | \textbf{-0.477} | 0.0241 | 2.4e-08 | -1.03 | 0.347 | 2.73e-08 | -1.63 | \textbf{0.387} |
| \( p\perp,1,\perp,2 \) | 9.4e-09 | 0.195 | 0.447 | \textbf{7.26e-09} | \textbf{0.521} | \textbf{0.737} | 9.31e-09 | 0.211 | 0.603 |
| **BGhigh** |        |      |     |        |      |     |        |      |     |
| \( p\| \) | 3.94e-06 | -0.0921 | 0.428 | \textbf{1.71e-07} | \textbf{0.792} | \textbf{0.943} | 7.11e-07 | -3.18 | 0.334 |
| \( p\perp,1 \) | 2.87e-07 | \textbf{0.289} | 0.719 | \textbf{5.32e-08} | 0.236 | \textbf{0.79} | 1.03e-07 | -1.23 | 0.631 |
| \( p\perp,2 \) | 1.68e-05 | 0.317 | 0.662 | \textbf{5.28e-08} | \textbf{0.422} | \textbf{0.847} | 2.12e-07 | -3.56 | 0.358 |
| \( p\parallel,\perp,1 \) | 1.53e-08 | -0.00444 | 0.219 | \textbf{9.61e-09} | \textbf{0.601} | \textbf{0.78} | 1.38e-08 | 0.181 | 0.461 |
| \( p\parallel,\perp,2 \) | 2.77e-08 | -0.238 | 0.309 | \textbf{2.32e-08} | \textbf{0.128} | \textbf{0.585} | 3.02e-08 | -0.479 | 0.288 |
| \( p\perp,1,\perp,2 \) | 6.18e-09 | -0.101 | 0.165 | \textbf{5.46e-09} | \textbf{0.139} | \textbf{0.377} | 8.32e-09 | -0.997 | 0.0181 |
seen from Figure 5 where both MLP and HGBR show a much larger spread in their prediction for BGhigh. From these results it is clear that predicting the diagonal components is a much easier task than predicting the off-diagonal components of the pressure tensor. From Figure 6 the improvements of the HGBR and MLP model over the linear regressor are clearly visible. But once again, the performance on BGhigh shows how the MLP model fails to capture the intrinsic dynamics.

C. Experiment 1 - Train on BG0 and BGhigh : Results

The three models are trained using the data sets of simulation BG0 and BGhigh. The evaluation on the test set of all four simulations can be found for the heat flux in Table V and for the pressure tensor in Table VI. The best performance for each feature in the unknown simulations, BGmed and BGlow, are highlighted in each table.

Looking at the heat flux in Table V we can already see that the HGBR model outperforms the other models for simulation BG0, while the MLP model shows better results for simulation BGhigh. Starting with simulation BGlow, we find that the MLP model is only slightly outperforming the HGBR on the features \( q_x \) and \( q_y \). The HGBR model performs best on \( q_z \), with most notably the very high \( R^2 \) value of 0.73, compared to the 0.365 of the MLP model. Next, for simulation BGmed, the HGBR and MLP model are again equivalent in performance on \( q_x \) and \( q_y \), but this time the HGBR is performing slightly better than the MLP model. For \( q_z \), the HGBR model is clearly the strongest, with the lowest RMSE and highest \( R^2 \) and \( r \) values out of the three models. In conclusion, the best overall performing model for the heat flux is the HGBR model, as can also be seen by observing Figure 7.

The performances reached for the pressure tensor prediction are shown in Table VI. From the diagonal components, the MLP and HGBR model perform equivalently on the known simulations BG0 and BGhigh, while also outperforming the linear regressor. For BGlow, both HGBR and MLP perform very well, with almost equivalent performances, with both ML models again outperforming the linear regressor. For simulation BGmed, the HGBR has the best performance compared to the other models, but again, the MLP model is not far behind in terms of the evaluation. Both ML models perform well and are capable of interpolating the diagonal components of the BGmed and BGlow simulation after training on BG0 and BGhigh. This can also be seen in Figure 8 where the prediction of the machine learning models follows the values of the simulation.
Identification of closure terms using machine learning

FIG. 4. Linear comparison between predicted and simulation values of the heat flux, for models trained on the BGlow and BGmed simulations. The line represents the simulation values. Figure (a) shows the linear regressor, figure (b) the HGBR model and figure (c) the MLP model.
Identification of closure terms using machine learning

FIG. 5. Linear comparison between predicted and simulation values of the diagonal pressure tensor components, for models trained on the BGlow and BGmed simulations. The line represents the simulation values. Figure (a) shows the linear regressor, figure (b) the HGBR model and figure (c) the MLP model.
Identification of closure terms using machine learning

FIG. 6. Linear comparison between predicted and simulation values of the off-diagonal pressure tensor components, for models trained on the BGlow and BGmed simulations. The line represents the simulation values. Figure (a) shows the linear regressor, figure (b) the HGBR model and figure (c) the MLP model.
Identification of closure terms using machine learning

Looking at the off-diagonal components of the pressure tensor in Table VI, we notice that both ML models show a lot more variation in their performance. For the known simulation BG0, the HGBR model is the best model, but the performance has decreased significantly compared to the diagonal components. For example, the HGBR performance on $p_{\parallel,\perp 1}$ is 0.249 and 0.5 for $R^2$ and $r$, respectively. The performance on simulation BGhigh is much better compared to BG0. Both the HGBR and MLP model have equivalent performance, except for $p_{\perp 1,\perp 2}$, where the HGBR model is slightly better. The analysis of the results obtained for the unknown simulations similarly shows that the ML models are much stronger than the linear regressor. Both HGBR and MLP have comparable performances for simulation BGlow. Notice that the value of $R^2$ is close to zero for the $p_{\parallel,\perp 1}$ feature, as also reflected by the linear comparison shown in Figure 9, indicating that the prediction of both models are uncorrelated with the truth. Finally, for BGmed, the HGBR model is again the best model. While the MLP model outperforms the HGBR slightly for $p_{\parallel,\perp 2}$, the HGBR model has a much better performance on $p_{\parallel,\perp 1}$ and $p_{\perp 1,\perp 2}$ in terms of $R^2$ and $r$ compared to the MLP model. These interpretations are strengthened by the linear comparison of Figure 9. The linear regressor, as well as the MLP model, struggle with prediction the $p_{\parallel,\perp 1}$ feature, predicting that almost all of the values are equal to one. For the two remaining features, a much more accurate prediction is seen for the machine learning models, compared to the linear regressor.

We can conclude that training the models on the more extreme cases of the simulations allows better performance on the less extreme cases, in particular on the more difficult off-diagonal pressure tensor components. The results also indicate that the HGBR model is a very stable model, showing a good performance on known simulations, and not showing extreme degradation in the accuracy on unknown simulations. The MLP model, on the other hand, sometimes has large decreases in performance, for example the $R^2$ value of -1.24 in Table VI for simulation BGmed on the $p_{\perp 1,\perp 2}$ feature.

D. Experiment 2 - Train on all simulations: Results

In this experiment, the models are trained on the training set of all four simulations, in order to determine if the overall accuracy on the test sets increased because of the additional information from other simulations. The model evaluation, for the heat flux, can be found in Table VII. The model evaluation of the pressure tensor is shown in Table VIII. The values in Table VII and VIII have
TABLE V. Evaluation of the three models for the first experiment on the electron species. The models were trained on the training sets of the BG0 and BGhigh simulation (as indicated by the T in the table, the so-called known simulations), and evaluated on the test set of each simulation.

| Target | LinReg | | HGBR | | MLP | |
|--------|--------|----|--------|----|--------|----|
|        | RMSE   | $R^2$ | $r$    | RMSE | $R^2$ | $r$ |
| $q_x$  | 2.47e-09 | 0.278 | 0.608  | 2e-09 | 0.551 | 0.77 |
| $q_y$  | 1.04e-09 | -0.0888 | 0.452  | 8.39e-10 | 0.302 | 0.57 |
| $q_z$  | 8.28e-09 | -3.13 | 0.573  | 2.11e-09 | 0.73 | 0.897 |
| $q_x$  | 7.03e-09 | 0.315 | 0.562  | 6.15e-09 | 0.474 | 0.697 |
| $q_y$  | 2.47e-09 | 0.241 | 0.499  | 2.14e-09 | 0.432 | 0.662 |
| $q_z$  | 2.09e-08 | -0.143 | 0.548  | 9.49e-09 | 0.762 | 0.875 |
| $q_x$  | 2.25e-09 | -0.158 | 0.582  | 1.12e-09 | 0.729 | 0.854 |
| $q_y$  | 9.06e-10 | -0.5 | 0.382  | 6.03e-10 | 0.349 | 0.591 |
| $q_z$  | 8.46e-09 | -6.11 | 0.574  | 1.22e-09 | 0.851 | 0.924 |
| $q_x$  | 5e-09 | 0.361 | 0.605  | 2.29e-09 | 0.866 | 0.931 |
| $q_y$  | 2.16e-09 | 0.318 | 0.566  | 1.32e-09 | 0.745 | 0.863 |
| $q_z$  | 2.95e-08 | 0.513 | 0.717  | 1.46e-08 | 0.88 | 0.943 |

been highlighted in bold, or red, when an increase, respectively decrease, of 10% or more has happened in their performance.

From the results obtained on the heat flux predictions in Table VII, we find that the performance remains unchanged for the HGBR model. The MLP model shows a notable decrease in $R^2$ for simulation BGlow, feature $q_z$, going from an $R^2$ of 0.428 (see Table III) to 0.361. However, the performance improved significantly for BG0, feature $q_z$, with the $R^2$ increasing from 0.508 to 0.711, and with a subsequent decrease in RMSE, from $2.23 \cdot 10^{-09}$ to $1.71 \cdot 10^{-09}$. The linear regressor also received a boost in its accuracy on simulation BG0, but the $R^2$ values of $q_z$ remain
Identification of closure terms using machine learning

TABLE VI. Evaluation of the three models for the first experiment on the electron species. The models were trained on the training sets of the BG0 and BGhigh simulation (as indicated by the T in the table, the so-called known simulations), and evaluated on the test set of each simulation.

| Target         | LinReg | HGBR | MLP |
|----------------|--------|------|-----|
|                | RMSE   | $R^2$ | $r$ |
| $p_{\parallel}$ | 2.97e-07 | 0.466 | 0.686 |
| $p_{\perp,1}$  | 8.1e-07  | 0.493 | 0.702 |
| $p_{\perp,2}$  | 0.00633  | 0.487 | 0.699 |
| $p_{\parallel,\perp,1}$ | 1.27e-08  -0.00486 0.0539 |
| $p_{\parallel,\perp,2}$ | 1.97e-08  0.0181 0.191 |
| $p_{\perp,1,\perp,2}$ | 8.48e-09  0.235 0.487 |

| Target         | RMSE   | $R^2$ | $r$ |
|----------------|--------|------|-----|
| $p_{\parallel}$ | 3.57e-07 | 0.479 | 0.496 |
| $p_{\perp,1}$  | 1.03e-06  | 0.443 | 0.719 |
| $p_{\perp,2}$  | 0.000566  | 0.458 | 0.723 |
| $p_{\parallel,\perp,1}$ | 1.15e-08  -0.0188 0.144 |
| $p_{\parallel,\perp,2}$ | 2.75e-08  0.0153 0.253 |
| $p_{\perp,1,\perp,2}$ | 5.19e-09  -0.0555 0.349 |

| Target         | RMSE   | $R^2$ | $r$ |
|----------------|--------|------|-----|
| $p_{\parallel}$ | 3.37e-07 | 0.478 | 0.692 |
| $p_{\perp,1}$  | 1.02e-06  | 0.512 | 0.716 |
| $p_{\perp,2}$  | 0.00177  | 0.508 | 0.714 |
| $p_{\parallel,\perp,1}$ | 9.58e-09  -0.0344 0.0107 |
| $p_{\parallel,\perp,2}$ | 1.68e-08  0.00871 0.111 |
| $p_{\perp,1,\perp,2}$ | 9.39e-09  0.197 0.444 |

| Target         | RMSE   | $R^2$ | $r$ |
|----------------|--------|------|-----|
| $p_{\parallel}$ | 4.38e-07 | 0.19  | 0.445 |
| $p_{\perp,1}$  | 4.33e-07  | 0.526 | 0.732 |
| $p_{\perp,2}$  | 0.000124  | 0.473 | 0.695 |
| $p_{\parallel,\perp,1}$ | 1.49e-08  0.0445 0.225 |
| $p_{\parallel,\perp,2}$ | 2.4e-08  0.0691 0.27 |
| $p_{\perp,1,\perp,2}$ | 5.71e-09  0.0581 0.329 |

| Target         | RMSE   | $R^2$ | $r$ |
|----------------|--------|------|-----|
| $p_{\parallel}$ | 4.28e-07 | 0.019 | 0.445 |
| $p_{\perp,1}$  | 4.33e-07  | 0.526 | 0.732 |
| $p_{\perp,2}$  | 0.000124  | 0.473 | 0.695 |
| $p_{\parallel,\perp,1}$ | 1.49e-08  0.0445 0.225 |
| $p_{\parallel,\perp,2}$ | 2.4e-08  0.0691 0.27 |
| $p_{\perp,1,\perp,2}$ | 5.71e-09  0.0581 0.329 |
Identification of closure terms using machine learning

FIG. 7. Linear comparison between predicted values and simulation values of the heat flux, for the models trained on the BG0 and BGhigh simulations. The line represents the simulation values. Figure (a) shows the linear regressor, figure (b) the HGBR model and figure (c) the MLP model.
Identification of closure terms using machine learning

FIG. 8. Linear comparison between predicted values and simulation values of the diagonal pressure tensor components, for the models trained on the BG0 and BGhigh simulations. The line represents the simulation values. Figure (a) shows the linear regressor, figure (b) the HGBR model and figure (c) the MLP model.
FIG. 9. Linear comparison between predicted values and simulation values of the off-diagonal pressure tensor components, for the models trained on the BG0 and BGhigh simulations. The line represents the simulation values. Figure (a) shows the linear regressor, figure (b) the HGBR model and figure (c) the MLP model.
Identification of closure terms using machine learning

very small for every simulation except BGhigh.

TABLE VII. Evaluation of the three models for the second experiment on the electron species. The models were trained on the training sets of all the simulations, and evaluated on the test set of each simulation. The bold values indicate an improvement of 10% or more of a performance, while a values in red indicate a decrease of 10% or more in performance, compared to the performance on the known simulations of Tables III and V.

| Target  | LinReg | HGBR | MLP |
|---------|--------|------|-----|
|         | RMSE   | $R^2$ | $r$ | RMSE   | $R^2$ | $r$ | RMSE   | $R^2$ | $r$ |
| BGlow - T |        |      |    |        |      |    |        |      |    |
| $q_x$   | 2.41e-09 | 0.327 | 0.637 | 1.44e-09 | 0.766 | 0.879 | 1.68e-09 | 0.681 | 0.838 |
| $q_y$   | 9.59e-10 | 0.0607 | 0.455 | 6.73e-10 | 0.551 | 0.746 | 8.03e-10 | 0.361 | 0.625 |
| $q_z$   | 6.16e-09 | -1.3 | 0.616 | 1.32e-09 | 0.894 | 0.951 | 1.95e-09 | 0.77 | 0.898 |
| BGmed - T |        |      |    |        |      |    |        |      |    |
| $q_x$   | 6.84e-09 | 0.35 | 0.594 | 3.42e-09 | 0.838 | 0.916 | 3.68e-09 | 0.812 | 0.904 |
| $q_y$   | 2.4e-09 | 0.284 | 0.535 | 1.32e-09 | 0.783 | 0.886 | 1.52e-09 | 0.714 | 0.847 |
| $q_z$   | 1.76e-08 | 0.186 | 0.555 | 7.72e-09 | 0.843 | 0.919 | 7.64e-09 | 0.846 | 0.921 |
| BG0 - T |        |      |    |        |      |    |        |      |    |
| $q_x$   | 2.11e-09 | **0.0241** | 0.599 | 1.12e-09 | 0.728 | 0.858 | 1.27e-09 | 0.654 | 0.825 |
| $q_y$   | 8.21e-10 | **-0.246** | 0.392 | 5.86e-10 | 0.385 | 0.625 | 6.43e-10 | 0.26 | 0.51 |
| $q_z$   | **6.53e-09** | **-3.26** | 0.637 | 1.18e-09 | 0.862 | 0.928 | **1.71e-09** | **0.711** | 0.871 |
| BGhigh - T |        |      |    |        |      |    |        |      |    |
| $q_x$   | 5.05e-09 | 0.348 | 0.602 | 2.33e-09 | 0.861 | 0.928 | 2.25e-09 | 0.87 | 0.934 |
| $q_y$   | 2.19e-09 | 0.303 | 0.551 | 1.32e-09 | 0.744 | 0.863 | 1.3e-09 | 0.754 | 0.868 |
| $q_z$   | 3.06e-08 | 0.477 | 0.715 | 1.66e-08 | 0.845 | 0.922 | 9.1e-09 | 0.954 | 0.977 |

The evaluation obtained from the pressure tensor predictions are shown in Table VIII. Once again, the HGBR model shows no notable change in performance. The MLP model, on the other hand, does show some improvements in the evaluation. More specifically, the accuracy on the off-diagonal pressure components increased for some of the features, as the $R^2$ value of $p_{11,12}$ in simulation BGhigh, almost doubled compared to the performance in Table VII. Finally, the linear
Identification of closure terms using machine learning regressor remains largely unchanged.

V. SUMMARY AND CONCLUSION

In this paper, three models were created that formed a closure relation for the electron pressure tensor and heat flux. The models used the magnetic and electric field, together with the first and second order moment were extracted and used as input features, extracted from four Double Harris sheet simulations generated with iPiC3D. The four simulations are identical in initial conditions except for the guiding background magnetic field, where four different values were taken from the range 0nT to 3nT.

The three models, a linear regressor, a multi-layer perceptron and a gradient boosting algorithm, were evaluated and compared to each other in two different experiments. The first experiment determined if the models could extrapolate the heat flux and pressure tensor to simulations outside of their trained domain, while the second experiment determined how much the models could generalize from an increased training set.

We learned that the gradient boosting algorithm, the HGBR model, is a stable model, capable of extracting the important dynamics from the data without overfitting. However, since the model showed almost no improvement when the training set was enlarged, one should consider if there might also be a limiting factor to how much this model can learn. The later point constitutes the focus for a later study, when more simulations are available for training. The MLP model on the other hand, benefited from the augmented training set. The MLP model can attain better performances than the HGBR model, at the cost of overfitting, which causes inaccurate predictions when extrapolating to other simulations.

Both models are viable candidates for further experiments, but the context of the available data should be taken into account. When a large training data set of simulations is available, the MLP model would be the most promising. If this is not the case, it seems the HGBR model would be the best choice as predictor.

From our experiments, we conclude that machine learning is a valuable tool to compute the higher-order moments of the plasma transport equations without using expensive computational methods like Particle-in-Cell. The next steps would involve training the models on more simulations with more variance in the initial conditions, and determining how far the models can generalize their predictive abilities on unseen simulation data.
Identification of closure terms using machine learning

TABLE VIII. Evaluation of the three models for the second experiment. The models were trained on the training sets of all the simulations, and evaluated on the test set of each simulation. The bold value indicate an improvement of 10% or more of a performance, while a values in red indicate a decrease of 10% or more in performance, compared to the performance on the known simulations of Table IV and VI.

| Target  | LinReg | HGBR | MLP |
|---------|--------|------|-----|
|         | RMSE   | $R^2$ | $r$ | RMSE   | $R^2$ | $r$ | RMSE   | $R^2$ | $r$ |
| **BGlow - T** |        |      |    |        |      |    |        |      |    |
| $p\parallel$ | 2.67e-07 | 0.467 | 0.684 | 7.67e-08 | 0.947 | 0.973 | 1.44e-07 | 0.914 | 0.958 |
| $p_{\perp,1}$ | 8.82e-07 | 0.485 | 0.699 | 4.89e-08 | 0.97 | 0.985 | 4.89e-08 | 0.965 | 0.983 |
| $p_{\perp,2}$ | 0.000203 | 0.479 | 0.694 | 4.7e-08 | 0.973 | 0.987 | 5.55e-08 | 0.975 | 0.988 |
| $p_{\parallel,1}$ | 1.26e-08 | 0.00274 | 0.0626 | 7.65e-09 | 0.634 | 0.81 | 6.78e-09 | 0.713 | 0.845 |
| $p_{\parallel,2}$ | 1.89e-08 | 0.0955 | 0.312 | 9.51e-09 | 0.772 | 0.88 | **9.94e-09** | 0.751 | 0.867 |
| $p_{\perp,1,\perp,2}$ | 8.47e-09 | 0.237 | 0.49 | 5.33e-09 | 0.698 | 0.836 | 5.71e-09 | **0.653** | 0.809 |
| **BGmed - T** |        |      |    |        |      |    |        |      |    |
| $p\parallel$ | 4.04e-07 | 0.271 | 0.524 | 7.69e-08 | 0.947 | 0.973 | 1.09e-07 | 0.923 | 0.962 |
| $p_{\perp,1}$ | 1.01e-06 | 0.532 | 0.747 | 1.56e-08 | 0.957 | 0.978 | 1.79e-08 | 0.955 | 0.977 |
| $p_{\perp,2}$ | **0.00119** | 0.538 | 0.749 | 1.76e-08 | 0.955 | 0.977 | 2.07e-08 | 0.957 | 0.978 |
| $p_{\parallel,1}$ | 1.12e-08 | 0.0217 | 0.161 | 5.21e-09 | 0.789 | 0.888 | **5.43e-09** | **0.772** | 0.883 |
| $p_{\parallel,2}$ | 2.49e-08 | 0.191 | 0.499 | 9.49e-09 | 0.882 | 0.941 | **8.23e-09** | 0.912 | 0.957 |
| $p_{\perp,1,\perp,2}$ | 4.96e-09 | 0.0384 | 0.366 | 3.27e-09 | 0.581 | 0.762 | 4.12e-09 | 0.336 | 0.596 |
| **BG0 - T** |        |      |    |        |      |    |        |      |    |
| $p\parallel$ | 3.21e-07 | 0.477 | 0.691 | 8.63e-08 | 0.951 | 0.975 | 1.47e-07 | 0.924 | 0.962 |
| $p_{\perp,1}$ | 1.12e-06 | 0.493 | 0.707 | 5.52e-08 | 0.976 | 0.988 | **4.78e-08** | 0.971 | 0.986 |
| $p_{\perp,2}$ | **0.00107** | 0.495 | 0.707 | 5.53e-08 | 0.978 | 0.989 | 5.79e-08 | 0.981 | 0.991 |
| $p_{\parallel,1}$ | 9.53e-09 | -0.0238 | 0.00872 | 8.38e-09 | 0.207 | 0.482 | 9.09e-09 | 0.0687 | 0.423 |
| $p_{\parallel,2}$ | 1.78e-08 | **-0.118** | **0.0511** | 9.67e-09 | 0.671 | 0.819 | 1.05e-08 | 0.61 | 0.784 |
| $p_{\perp,1,\perp,2}$ | 9.38e-09 | 0.199 | 0.447 | 6.52e-09 | 0.614 | 0.784 | 7.12e-09 | **0.539** | 0.741 |
| **BGhigh - T** |        |      |    |        |      |    |        |      |    |
| $p\parallel$ | 5.18e-07 | 0.177 | 0.44 | 9.29e-08 | 0.975 | 0.988 | 1.21e-07 | 0.958 | 0.98 |
| $p_{\perp,1}$ | 5.02e-07 | 0.534 | 0.735 | 2.62e-08 | 0.927 | 0.964 | 3.17e-08 | 0.908 | 0.956 |
| $p_{\perp,2}$ | **7.19e-06** | 0.479 | 0.695 | 2.54e-08 | 0.937 | 0.968 | 3.06e-08 | 0.927 | 0.964 |
| $p_{\parallel,1}$ | 1.5e-08 | 0.0352 | 0.236 | 6.64e-09 | 0.81 | 0.9 | 6.63e-09 | 0.81 | 0.903 |
| $p_{\parallel,2}$ | 2.38e-08 | 0.0844 | 0.313 | 1.04e-08 | 0.826 | 0.909 | 9.09e-09 | 0.866 | 0.933 |
| $p_{\perp,1,\perp,2}$ | 5.73e-09 | 0.0545 | 0.292 | 4.14e-09 | 0.506 | 0.714 | **4.84e-09** | **0.324** | **0.573** |
Identification of closure terms using machine learning

AUTHOR'S CONTRIBUTION

BL performed and analysed the experiment and wrote the manuscript. GL created and provided the PiC simulation data and contributed to the introduction. GL and JA planned the study and GL, JA and SJ provided intellectual contribution to the manuscript. All authors took part in the manuscript revision and have read and approved the submitted version.

ACKNOWLEDGEMENT

This work was supported by the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 776262 (AIDA, Artificial Intelligence for Data Analysis, www.aida-space.eu).

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

1 S. Chapman and T. G. Cowling, The mathematical theory of non-uniform gases: an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases (Cambridge university press, 1990).
2 J. M. Burgers, Flow equations for composite gases (Academic Press, 1969).
3 C. D. Levermore, Journal of statistical Physics 83, 1021 (1996).
4 C. Negulescu and S. Possanner, Multiscale Modeling & Simulation 14, 839 (2016).
5 T. Passot, P. Sulem, and E. Tassi, Journal of Plasma Physics 83 (2017).
6 F. Haas, J. Zamanian, M. Marklund, and G. Brodin, New Journal of Physics 12, 073027 (2010).
7 S. Braginskii, Sov. Phys. JETP 6, 358 (1958).
8 P. Helander and D. J. Sigmar, Collisional transport in magnetized plasmas, Vol. 4 (Cambridge university press, 2005).
9 J. P. Freidberg, Plasma physics and fusion energy (Cambridge university press, 2008).
Identification of closure terms using machine learning

10. A. Fasoli, S. Brunner, W. Cooper, J. Graves, P. Ricci, O. Sauter, and L. Villard, Nature Physics 12, 411 (2016).
11. C. Ma, B. Zhu, X.-Q. Xu, and W. Wang, Physics of Plasmas 27, 042502 (2020).
12. B. M. de Silva, D. M. Higdon, S. L. Brunton, and J. N. Kutz, Frontiers in artificial intelligence 3, 25 (2020).
13. E. P. Alves and F. Fiuza, arXiv preprint arXiv:2011.01927 (2020).
14. A. Beck and M. Kurz, GAMM-Mitteilungen 44, e202100002 (2021).
15. J.-L. Wu, H. Xiao, and E. Paterson, Physical Review Fluids 3, 074602 (2018).
16. M. Yamada, F. Levinton, N. Pompfrey, R. Budny, J. Manickam, and Y. Nagayama, Physics of plasmas 1, 3269 (1994).
17. M. Yamada, R. Kulsrud, and H. Ji, Reviews of Modern Physics 82, 603 (2010).
18. S. Markidis, G. Lapenta, et al., Mathematics and Computers in Simulation 80, 1509 (2010).
19. P. Ricci, G. Lapenta, and J. Brackbill, Journal of computational physics 183, 117 (2002).
20. G. D. Knott, Interpolating cubic splines, Vol. 18 (Springer Science & Business Media, 2000).
21. C. K. Birdsall and A. B. Langdon, Plasma physics via computer simulation (CRC press, 2018).
22. R. W. Hockney and J. W. Eastwood, Computer simulation using particles (crc Press, 2021).
23. M. V. Goldman, D. L. Newman, and G. Lapenta, Space Science Reviews 199, 651 (2016).
24. J. Scudder and W. Daughton, Journal of Geophysical Research: Space Physics 113 (2008).
25. I. Goodfellow, Y. Bengio, and A. Courville, Deep learning (MIT press, 2016).
26. E. Camporeale, Space Weather 17, 1166 (2019), arXiv: 1903.05192.
27. G. E. Box and D. R. Cox, Journal of the Royal Statistical Society: Series B (Methodological) 26, 211 (1964).
28. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., the Journal of machine Learning research 12, 2825 (2011).
29. J. H. Friedman, Annals of statistics , 1189 (2001).
30. J. R. Quinlan, Machine learning 1, 81 (1986).
31. G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, Advances in neural information processing systems 30, 3146 (2017).
32. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Advances in neural information processing systems 32, 8026 (2019).
Identification of closure terms using machine learning

33T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2019) pp. 2623–2631.

34L. Li, K. Jamieson, A. Rostamizadeh, E. Gonina, M. Hardt, B. Recht, and A. Talwalkar, arXiv preprint arXiv:1810.05934 (2018).

35S. Ruder, arXiv preprint arXiv:1609.04747 (2016).

36D. Freedman, R. Pisani, and R. Purves, Pisani, R. Purves, 4th edn. WW Norton & Company, New York (2007).

37M. W. Liemohn, A. D. Shane, A. R. Azari, A. K. Petersen, B. M. Swiger, and A. Mukhopadhyay, Journal of Atmospheric and Solar-Terrestrial Physics 218, 105624 (2021).