



Supervised machine learning-based Hall thruster scaling

Alfredo Marianacci, Stéphane Mazouffre

► To cite this version:

Alfredo Marianacci, Stéphane Mazouffre. Supervised machine learning-based Hall thruster scaling. Journal of Electric Propulsion, 2024, 3, pp.14. 10.1007/s44205-024-00077-y . hal-04666792

HAL Id: hal-04666792

<https://hal.science/hal-04666792v1>

Submitted on 2 Aug 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

RESEARCH

Open Access



Supervised machine learning-based Hall thruster scaling

Alfredo Marianacci^{1*} and Stéphane Mazouffre¹

*Correspondence:
alfredo.marianacci@cnrs-orleans.fr

¹ ICARE, CNRS, 1C Avenue
de la Recherche Scientifique,
Orléans 45100, France

Abstract

The scaling methodology described in this paper to find the geometry and working parameters of Hall Thrusters is based on algorithms of supervised Machine Learning. The approach considers the determination of the geometrical sizes, propellant mass flow rate and discharge voltage taking thrust and specific impulse as requirements. The magnetic field is also considered. The Gradient Boosting Regression is found as the most suitable algorithm for our purpose. Scaling relies on a specific database of 54 thrusters for the determination of all parameters. The database includes measurements carried out with xenon, krypton and argon as propellant. A unique analytical approach based on the GBR algorithm has been developed and validated to determine the suitable design for a Hall thruster according to space mission requirements.

Keywords: Hall thruster, Machine learning, Scaling laws, Electric propulsion

Introduction

In recent years, Hall Thrusters (HTs) have attracted increasing interest in the spacecraft propulsion field due to particular advantages such as high Δv and total impulse, high thrust-to-power ratio and thrust density, and an efficiency in excess of 50 %. In addition, HTs operate over a fairly wide range of power levels and can therefore be used for both microsatellites and large satellites in Earth orbit and for deep space missions. Although they are commonly used nowadays, some internal physical mechanisms remain unknown, such as anomalous electron transport and plasma-wall interactions[1, 2]. This means that building and optimization of a new thruster is complex, long in duration and mostly empirical whatever the size and power level. Unfortunately, advanced 2D Particle-In-Cell or fluid codes are of little help in the development of Hall Thrusters, since they are not predictive and must be validated using measurements. In fact, the construction of simple models to help with design and data interpretation has characterized the development of Hall thrusters from the outset. Thanks to the gradual development and use of these thrusters over the past decades, it has been possible to develop methodologies based on scaling laws derived from reference thrusters along with database [3, 4]. The common limitations of these approaches lie mainly in the equations on which they are based. Although useful for obtaining an initial estimate of thruster parameters and performances, the potential of 0-D scaling relations remains limited. The latter are in

fact too simple to correctly reproduce the physics at play in HTs. Simplifying assumptions are indeed made on loss factors, plume divergence, thermal load, magnetic field topology, presence of multiply-charged ions, to name just a few examples [5]. All these approximations often result in large differences between predictions and performance measurements.

In this work, the problem was addressed through an approach that was no longer purely statistical but based on Machine Learning (ML) algorithms. The field of ML is very broad and includes more or less complex algorithms covering a wide range of topics. In this study, supervised machine learning regression algorithms were analysed with the aim of determining continuous output values from input data [6]. Such an approach makes it possible to avoid in principle any kind of initial assumption and is disconnected from the physical relationships between the parameters involved [7]. The objective of this study was therefore to improve the scaling methodologies with a non-statistical approach, in order to assess the size and properties of HTs from requirements such as thrust and specific impulse. To achieve this, a validation of our database was carried out with the principal scaling laws derived over the years, see [Scaling laws: the classical approach](#) section. In [Machine learning approach](#) section, the Machine Learning algorithm that is most appropriate for exploring the database is selected. In [Procedure to design a new thruster](#) section, the selected model was used in an optimization method to determine the characteristics of a generic new Hall Thruster. Finally, conclusions and perspectives are given in [Conclusion](#) section.

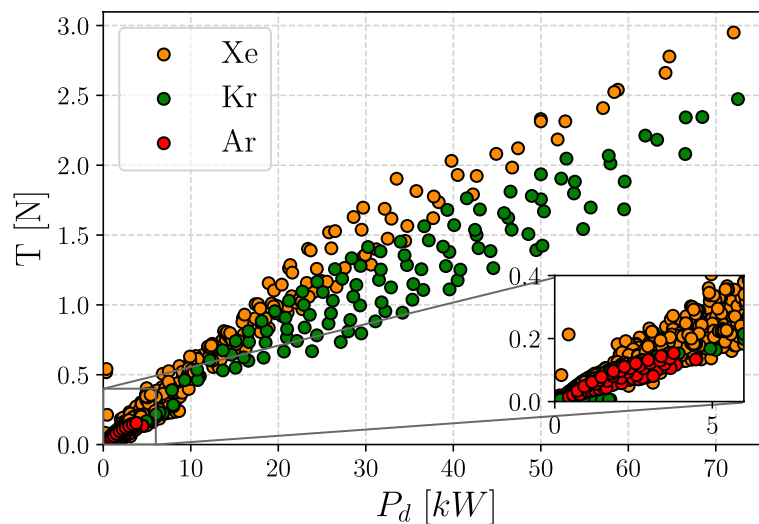
Scaling laws: the classical approach

Database

The data used in this study came from databases consisting of data collected since 2012 and stored in our laboratory, combined with an extensive search for new tests in the literature, sometimes supplemented by extrapolation of data from graphs provided in various scientific journal papers, doctoral theses and conference articles. The data collected concern Hall thrusters firing from 50 W up to 70 kW [8–34]. The database features 54 thrusters, with a total number of 3323 different operating conditions. The quantities included in the database are given in Table 1. The channel mean diameter is the parameter $d = \frac{1}{2}(d_{ext} + d_{int})$, where d_{ext} and d_{int} are the external and internal channel diameter, while $h = \frac{1}{2}(d_{ext} - d_{int})$ is the channel width. L is the discharge channel length. The database only contains information on stationary-plasma-thruster type HTs, meaning that TAL are not considered. Very little data is available on magnetically shielded thrusters, as it is a fairly recent technology. Most of the tests were performed with xenon as propellant. There is however data with krypton and argon, which allows to study the characteristics of thrusters according to the type of propellant. Xenon-fueled HTs cover 82 % of the database and Krypton-fueled thrusters 14 %. Unfortunately, the full set of characteristics is not available for each condition. In fact, leaving aside the magnetic field and considering only complete sets, 1570 points are available for xenon, 268 for krypton and 80 for argon. The parameters we considered cover wide ranges, as the database includes small thrusters and large thrusters with powers ranging from a few watts up to 70 kW, as can be seen in Fig. 1, where the thrust T is plotted as a function of the input power P_d for the three

Table 1 Database parameters

PARAMETER	SYMBOL	UNIT
Model	-	-
Developer	-	-
Type	HT(SPT or TAL), MS(Magnetic shielded)	-
Propellant	Xe,Ar,Kr	-
Diameter	d	m
Height	h	m
Length	L	m
Mass flow rate	\dot{m}_n	kg/s
Voltage	U_d	V
Current	I_d	A
Power	P_d	W
Thrust	T	N
Specific impulse	I_{sp}	s
Thrust Efficiency	η	-
Magnetic field	B	G

**Fig. 1** Thrust T as a function of the discharge power P_d . The zoom at the bottom represents the position of the majority of the data in the database

propellants. It is worth noting that the database has 79 % of the tests for powers below 5 kW, as illustrated in the inset plot in Fig. 1. In short our database is particularly focused on small and medium size Hall thrusters.

Standard scaling laws

The statistical scaling methodologies often rely upon well-known and widely tested thrusters. Starting from that, it is possible to determine proportionality relationships between intensive and extensive parameters that characterize thrusters. Assuming similarity criteria are valid, thanks to the above-mentioned laws it is possible to identify the variation in the value of the parameters when the thruster scale is changed. Since the

development of HTs in the 1970s, numerous works have concerned this field, varying one or more parameters while keeping others constant [35–37, 22, 38, 39].

Recent works have considered the extraction of scaling relations from physical processes validated against a database. Dannenmayer et al. took into account an atom density constraint inside the channel, they also assume the height h and the mean diameter d of the channel are proportional [5]. They developed two approaches, one with few assumptions and one with numerous assumptions on different performance parameters, based on the determination of coefficients associated with linear trends in the data. The same approach was followed by Lee et al. [40] with the creation of a database focusing on sub-kilowatt Hall thrusters.

In this article, this approach is applied to the new database described above. In order to obtain the design characteristics of a new thruster, basically its geometrical characteristics, it is necessary to consider assumptions on losses and proportionality between channel height and diameter, as described in [5]. Figure 2 shows clearly that $h \propto d$ is a very good assumption. However, as demonstrated by Mazouffre et al. [41], considering that the surface-to-volume ratio scales as $2/h$, to reduce losses and to increase thrust the height of the channel should increase, without relation to the diameter. Figure 2 in fact illustrates a technological consequence resulting from the fact that thrusters are always scaled from a reference thruster, quite often the Russian SPT100 thruster.

Starting from mission requirements such as thrust and specific impulse, it is possible to develop an iterative procedure to determine the characteristics of a new thruster. As explained, laws are determined from physical processes [5, 40] and coefficients are found using databases. An usual relation for the thrust is:

$$T \propto \dot{m}_n \cdot \sqrt{U_d}, \quad (1)$$

where \dot{m}_n is the propellant mass flow rate and U_d is the discharge voltage.

Figure 3 indicates Eq. 1 is valid for a broad range of parameters. Inserting the propellant atomic mass M_n in the linear relation avoids the dependence of the latter on the

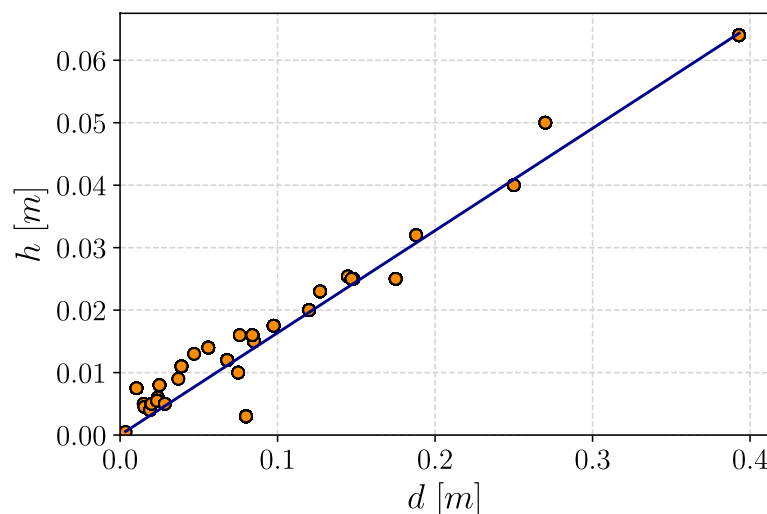


Fig. 2 Linear relation between channel height and diameter

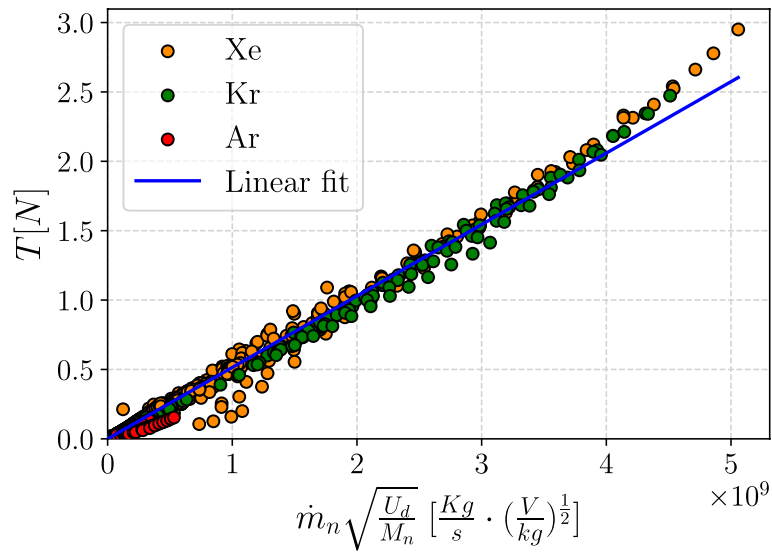


Fig. 3 Thrust against a function of (\dot{m}_n, U_d, M_n) for Xe, Kr and Ar. A linear fit to all data is also shown

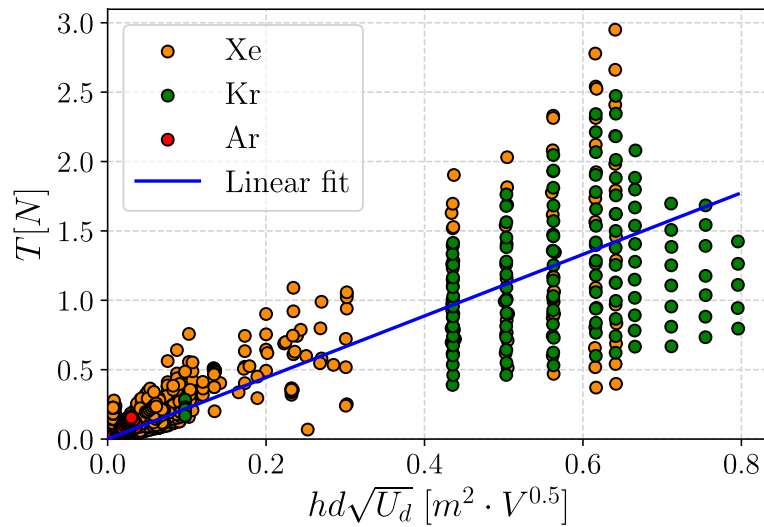


Fig. 4 Thrust as a function of $h d \cdot \sqrt{U_d}$ and linear fit for Hall Thrusters in the database. Data are divided by type of propellant

type of propellant, as can be seen in Fig. 3. Given the linear dependence of mass flow rate on cross-sectional area of the Hall Thruster, Eq. 1 becomes:

$$T \propto h d \cdot \sqrt{U_d} \quad (2)$$

where h, d are the channel height and mean diameter, respectively.

Figure 4 illustrates Eq. 2. The data spread in Fig. 4 is large compared to Fig. 3. The observed scatter in the data can be attributed to the collection methodology, wherein measurements were taken from same thrusters operating at varying mass flow rates. This approach results in a vertical distribution pattern on the graph in Fig. 4, characterized by

constant abscissa values. Consequently, the thrust, represented on the y-axis, depends on mass flow rate values. A scaling relationship must also be defined for the specific impulse. Since it is directly proportional to the ion velocity, one gets:

$$I_{sp} \propto \sqrt{U_d} \quad (3)$$

where I_{sp} is the specific impulse. Figure 5 shows the linear trend of data where the spread is due to the different anode flow rate values.

Using the laws presented above and the determination of linear fitting coefficients on the available data, the main characteristics of a generic new thruster can be estimated. The procedure illustrated helps to give an idea of the geometry that a thruster should have if certain thrust and specific impulse are required, in a first approximation. It is certainly necessary to go into more detail, considering aspects such as thermal loads on walls, multiple charged ions, several losses and anomalous electron transport. The main limitations of the traditional approach are the lack of consideration for the magnetic field and necessary assumptions about links between geometrical characteristics.

In the following approach, which is based on machine learning algorithms applied to the database, the model will be left free of initial assumptions, although it still depends on the available data, and in the second analysis the magnetic field will be included in the fundamental parameters of the model.

Machine learning approach

Introduction

The machine learning approach is based on the determination of thrust and specific impulse by knowing the values of mass flow rate, discharge voltage, geometric dimensions (h, d, L) and, secondarily, magnetic field. The type of propellant was also added as a parameter in the model. Thrust and specific impulse are in fact the mission requirements, while all the others parameters are the targets. But algorithms are designed

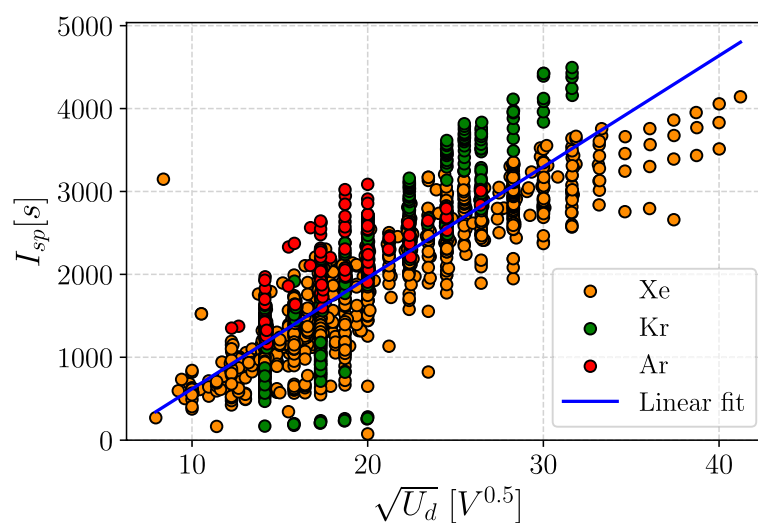


Fig. 5 Specific impulse as a function of $\sqrt{U_d}$ and linear fit to all data

to obtain a continuous value from a set of inputs, usually greater in number than the desired outputs.

Data pre-processing

Before analysing the machine learning algorithms, it is necessary to prepare the available data. In fact, the range of values for the variables considered here is very different (e.g. mass flow rate in orders of 10^{-6} and voltage in orders of hundreds). This different scale does not allow an equal treatment of variables. The methods for scaling data in the same range are many and should be chosen carefully according to the type of data. The most common is the standardization that subtracts the mean value from each data item and places all data in a range with zero mean and unit variance [7]. Usually, this kind of scaling is used for data that presents a Gaussian distribution, which is not the case of our database. For this reason this method was discarded. The method chosen is the normalization, where all the values are normalized inside a range from 0 to 1. Technically, we normalize the data between a value very close to 0 and a value very close to 1 to avoid singularity. The normalization law reads:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (4)$$

where x is the actual value, x_{max} and x_{min} are the maximum and minimum value of each parameter. When scaling data for machine learning models, it is imperative to restrict the scaling process to the training set only. Including the entire dataset in the scaling process introduces information leakage, whereby the model inadvertently learns from the test set. This leakage occurs because the normalization parameters calculated from the whole dataset incorporate information from both the training and test sets. Consequently, when predictions are made on the test set, the model has indirectly been exposed to the test data during the scaling process. This exposure results in an unrealistic evaluation, as the test set should represent unseen data to accurately assess the model's performance. Hence, to ensure the validity of the evaluation procedure and the generalizability of the model, scaling should be strictly confined to the training data prior to any model fitting or testing.

Given the constraints imposed by the relatively small dataset size, traditional partitioning methods for training and testing were impractical. The Kfold-cross-validation procedure was adopted to make the model more robust and reliable [42].

This technique, illustrated in Table 2, is based on dividing the available data into sub-folders, in this case five, where 4/5 of the data is used for training and 1/5 for testing, per each sub-folder. In this way, the model is trained five times instead of just one, and there are five mean errors between the target and the predicted values. Finally, a global mean error is calculated. There are several metrics to evaluate the performance of a model. In this work, the Mean Absolute Error (MAE) was chosen. It evaluates the modulus of the difference between the predicted value and the actual value, without taking direction into account. The choice was dictated by the fact that this type of metric is much less sensitive to outliers, i.e. data that shows a different trend from the general trend, usually due to measurement errors [7]. The expression of MAE [6] is the following:

Table 2 Schematic representation of the work made by the cross-validation tool

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5		
TRAINING	TRAINING	TRAINING	TRAINING	TESTING	→ ERROR	→ MEAN ERROR
TRAINING	TRAINING	TRAINING	TESTING	TRAINING	→ ERROR	
TRAINING	TRAINING	TESTING	TRAINING	TRAINING	→ ERROR	
TRAINING	TESTING	TRAINING	TRAINING	TRAINING	→ ERROR	
TESTING	TRAINING	TRAINING	TRAINING	TRAINING	→ ERROR	

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (5)$$

where n is the number of samples, y_i are the actual values and \hat{y}_i are the predicted values. Furthermore, in order to enhance the clarity of the results presentation, percentage error was employed as a comparative measure across different models or outcomes. The percentage error is calculated according to Equation 6. This metric facilitates a straightforward comparison of the predictive accuracy of various models, thereby providing a consistent and interpretable measure of performance.

$$\%_{error} = \frac{|y_i - \hat{y}_i|}{y_i} \times 100\% \quad (6)$$

Finally, a one-hot encoding (O-HE) procedure [42] was considered to also include the propellant within the parameters, as it is intended that the model uses the type of propellant as useful data in determining the optimal characteristics of a thruster. The type of propellant was converted into a three dimensional vector in which the component corresponding to the propellant type was set to 1 and the others were set to 0, as shown in Table 3.

Model selection

Introduction

The first step in determining a model capable of predicting thrust and specific impulse values by learning from the available data is to choose the most suitable machine learning algorithm. The selected algorithms are part of the supervised group, in the regression branch, as the aim is to determine continuous values [43].

Model selection with default hyperparameters

The procedure started by evaluating various types of supervised machine learning algorithms, aiming to cover almost the full spectrum of this category. Below are the algorithms initially chosen [42, 43]:

Table 3 One-hot encoding for propellant

Type	O-HE	Xe	Kr	Ar
Xe	→	1	0	0
Kr	→	0	1	0
Ar	→	0	0	1

- Linear Regression (LR) [43]
- Decision Tree Regression (DTR) [43]
- K-Nearest Neighbors Regression (KNR) [43]
- Random Forest Regression (RFR) [43]
- Gradient Boosting Regression (GBR) [43]
- Bagging Regression (BR) [43]

Each of these algorithms was tested on the database which, as previously explained, only has 1918 tests available due to missing data; 30% of data was set aside for the final validation of the chosen model, while 70% was used for training and testing with the cross validation procedure described above. Each model has its own hyperparameters, which are parameters of the algorithm class that define its learning process. Algorithms have default hyperparameters, based on the most common functions used per each model [43]. In a first attempt, models were tested with them, to have an initial estimation of their performances. Table 4 shows the results of the approach. One can notice that the mean errors found are all in the same order of magnitude. Linear Regression is, however, the worst algorithm in predicting. The best two are instead the Random Forest and the Gradient Boosting, both coming from the group of Ensemble Learning and both having decision trees as estimators [43].

Model selection with hyperparameters tuning

A more efficient procedure is to allow the algorithm itself to define the best hyperparameters for the specific database, instead of using the default ones. A procedure of this type is called *hyperparameters tuning* [42]. The most common tool that performs this task is the GridSearchCV, that essentially tests all combinations of all possible choices and evaluate the performance for each one. It is a very long process when the possible hyperparameters choices are numerous and when the procedure has to be applied per each model. The computational time increases exponentially. For that reason the tool was discarded. To speed up the process of finding the best model with the best hyperparameters, the HalvingGridSearchCV tool was considered. Briefly, this machine learning tool does not check combinations of hyperparameters directly on all available data, but starts by evaluating combinations on a small amount of data called resources. Only some of these combinations called candidates are selected for the next iteration to which more resources will be allocated. The number of

Table 4 Mean Absolute Errors of the models with default hyperparameters analysed using a cross evaluation

Model	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
LR	0.052	0.062	0.06	0.054	0.059	0.057
DTR	0.019	0.026	0.022	0.021	0.027	0.023
KNR	0.023	0.031	0.023	0.024	0.028	0.026
RFR	0.012	0.022	0.017	0.014	0.021	0.017
GBR	0.015	0.019	0.017	0.015	0.019	0.017
BR	0.013	0.02	0.019	0.015	0.025	0.019

iterations is established based on the number of samples and the number of parameters involved. The combination that arrives at the end of the iterations is the best one in terms of the smallest prediction error. It's worth noting that this tool is much faster than GridSearchCV. In all the process, the algorithm performs a cross-field validation per each combination, as explained in [Data pre-processing](#) section. Thus, the tool combines the subdivision procedure for evaluating the analysed models through MAE with the search within the models themselves for the best hyperparameters in order to achieve the lowest possible error.

Table 5 shows the results obtained with the tuning procedure. All models show improvements over the case analyzed in the [Model selection with default hyperparameters](#) section, except for the Bagging Regression which now performs worst together with the linear regression. The other models work in a high-performance manner, with a percentage error below the 60 % with respect to the Linear Regression. This percentage has been calculated following the formula presented in Eq. 6. As a final step, the models are evaluated on the testing set left apart. The graph in Fig. 6 illustrates the comparative performance of the models by the absolute errors made in predicting the output values for each test case present in the testing dataset.

The performances of KNR, RFR and GBR are almost similar, but the distribution curve of the Gradient Boosting seems to drop first to null values of absolute errors. Moreover its histogram is more concentrated on the extreme left, where the errors are close to zero. Thus, considering in addition the results in [Model selection with default hyperparameters](#) section, the model chosen as the best one for our data is the Gradient Boosting Regression, with an overall Mean Absolute Error of 0.0134.

Gradient Boosting Regression

Building of trees and prediction

The Gradient Boosting Regression (GBR) is an ensemble algorithm that puts together weaker models to perform better as a whole [43]. It is based on decision trees, whose number is decided in the hyperparameters tuning phase. In this analysis, the model generates 200 sequential trees. Gradient Boosting works on the gradient of the Loss function, minimizing the errors between predicted and actual data [44]. The model is initialized with an initial value:

Table 5 Hyperparameters tuning and evaluation of the six models

Model	Best hyperparameters	MAE
LR	(‘fit_intercept’: True, ‘positive’: False)	0.019
DTR	(‘criterion’: ‘absolute_error’, ‘max_depth’: 10, ‘min_samples_leaf’: 1, ‘min_samples_split’: 3, ‘random_state’: 10, ‘splitter’: ‘best’)	0.009
KNR	(‘algorithm’: ‘kd_tree’, ‘n_neighbors’: 3, ‘weights’: ‘distance’)	0.007
RFR	(‘criterion’: ‘absolute_error’, ‘max_depth’: 10, ‘min_samples_leaf’: 1, ‘min_samples_split’: 2, ‘n_estimators’: 200, ‘random_state’: 500)	0.007
GBR	(‘criterion’: ‘squared_error’, ‘learning_rate’: 0.1, ‘loss’: ‘absolute_error’, ‘max_depth’: 10, ‘min_samples_leaf’: 1, ‘min_samples_split’: 5, ‘n_estimators’: 200, ‘random_state’: 500)	0.006
BR	(‘estimator’: DecisionTreeRegressor(), ‘max_features’: 8, ‘max_samples’: 10, ‘n_estimators’: 200, ‘random_state’: 500)	0.02

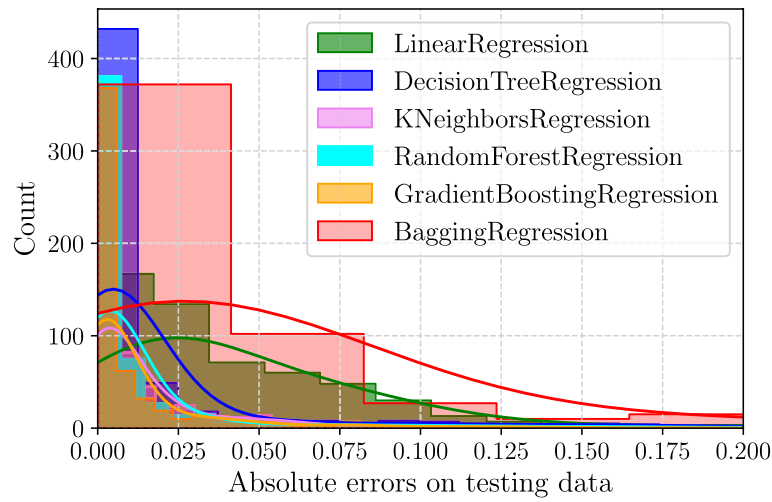


Fig. 6 Models comparison in terms of absolute errors on the testing set

$$F_0(x) = \arg \min_{\gamma} \left[\sum_{i=1}^n L(y_i, \gamma) \right] \quad (7)$$

where F_0 is the initial value of the model and it is equal to the value of γ that minimizes the loss function $\sum_{i=1}^n L(y_i, \gamma)$, represented by the Mean Absolute Error, where y_i are the actual value, γ the predicted values and n the number of tests. To find therefore the value of γ , a derivative of the loss function is taken and is set to zero:

$$\frac{dL}{d\gamma} = \frac{d \sum_{i=1}^n |y_i - \gamma|}{d\gamma} = - \sum_{i=1}^n \text{sign}(y_i - \gamma) = 0 \quad (8)$$

The sign function is either -1, 0 or 1 and no matter how distant the target is from the current prediction. The model is trained on just the direction, without the magnitude. Considering the latter in fact the computations are easily skewed by outliers. Solving the equation above, we obtain that the value of γ that minimizes the loss function is the median of the output values in the training dataset. It is worth remind that we have both thrust and specific impulse as outputs of the model, therefore the GBR is trained on each output separately but the performance evaluation is a mean of the performance evaluation of the model on the two separate outputs. Once $F_0(x)$ is calculated, the GBR generates decision trees. Each leaf is created by splitting the training data through values greater or lower of a certain threshold. The algorithm itself is able to analyse the consequences of each splitting and from there to decide for which threshold value the results are best in terms of prediction. The evaluation of the split is made through the *criterion* function, see Table 5. Since the magnetic field is not considered in this first analysis, our inputs are eight. The GBR is capable of dividing the training set by acting on all parameters simultaneously, finding the optimal threshold values for best performance. The mechanism is therefore quite complex. The same formula is then applied for each leaf, where the output is in fact:

$$\gamma_{jm} = \arg \min_{\gamma} \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma) \quad (9)$$

where j is the leaf, m is the tree and R_{ij} are the output of the leaves. For the same reason as before, the output in each leaf is exactly the median value. Finally, this value is first multiplied by a α , the learning rate, that controls the speed of the process and then is added to the previous one, that in this case was $F_0(x)$:

$$F_m(x) = F_{m-1}(x) + \alpha \sum_{j=1}^{J_m} \gamma_{jm} \quad (10)$$

Once the first decision tree is built on the training data, it has to be evaluated on the testing set through the MAE. It is important to remind that the Gradient Boosting Regression builds sequential decision trees, thus the first one gives the worst prediction and step by step, i.e. tree by tree, the performance is improved.

Evaluation of the model

Splitting the data with respect to a certain parameter values is also useful to know which are the most important parameters in the model. In fact, if the performance of the model varies significantly when the threshold value of a certain parameter is changed, it means that the parameter is important.

Figure 7 represents the percentage of importance of the various parameters considered in the decision tree building procedure. The histograms are normalized to a value up to 1. The mass flow rate is the most important input of the model: a change in its values generates a change in the model predictions. A physical explanation for this behavior may lie in the fact that the geometrical features of the thruster are linked to the mass flow rate, which by varying implicitly generates a change in the others. Furthermore, the correlation between the geometrical features themselves makes very difficult for the model to understand the single contribution of them in the prediction of the thrust and specific impulse. Therefore the model gives more importance to parameters where the

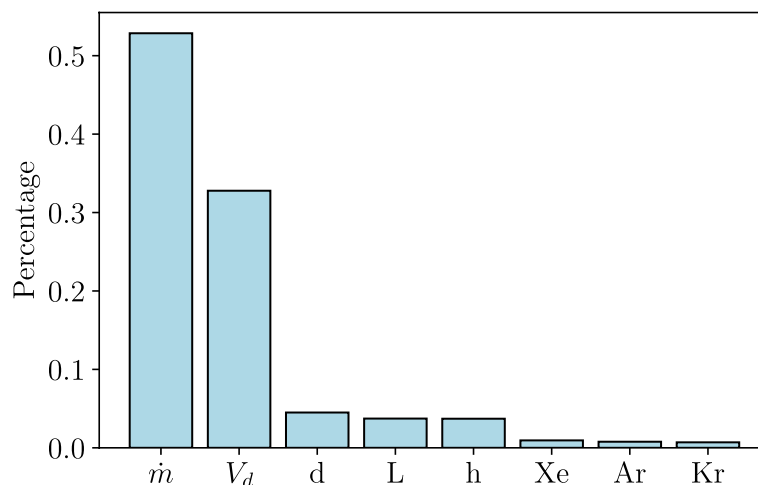


Fig. 7 Features importance in the training phase of the Gradient Boosting

independence is clearer, as mass flow rate and voltage. This correlation can be visualized with a correlation matrix, see Fig. 8.

The correlation matrix helps in understanding the correlation between variables. It measures the covariance, i.e. how much a parameter changes if another one is changed. It is normalized between -1 and 1. Usually, if the value is between 0.3 and 1 there is a direct correlation between the two parameters. In fact, see Fig. 8, the geometrical features have an high value of correlation among themselves and with mass flow rate. If the values are between -0.3 and -1, there is an inverse correlation, while if it is between -0.3 and 0.3 there is no correlation. In our case, the only independent variable is the discharge voltage V_d and the propellants are dependent among themselves.

As explained previously, the model is evaluated on the testing dataset. In Fig. 9 is represented the prediction of the two outputs made by GBR after all the 200 trees are built.

The collective behavior is good enough, since the model, represented by the blue crosses, is effective in prediction of thrust and specific impulse values in the entire range. In the upper right prediction is weaker due to the lack of data for high power thrusters in the database. To better understand the improvement in performance of the Gradient Boosting algorithm, it is useful to compare it to a thrust prediction model based on a scaling law, see Eq. (2), which is recalled for clarity:

$$T \propto h d \cdot \sqrt{U_d} \quad (11)$$

Figure 10 shows the thrust predictions of the Gradient Boosting model for the thrust as a function of diameter, while Fig. 11 shows the predictions of the linear scaling law described by the Eq. (2). The vertical distribution of data for a given diameter value is linked to changes of mass flow rate values for tests performed with a given Hall thruster (d fixed then).

It is worth noting that the two models were not tested on the same data, because the splitting of data in training and testing is done randomly. The GB model in Fig. 10 was

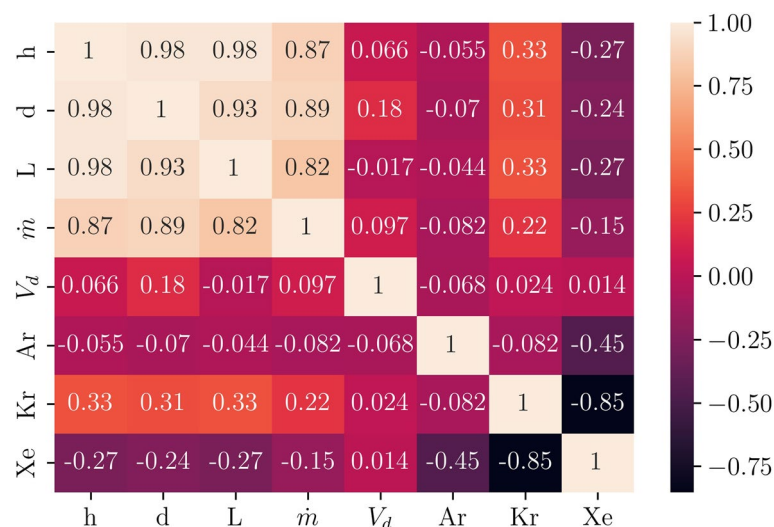


Fig. 8 Correlation matrix between inputs parameters in GBR

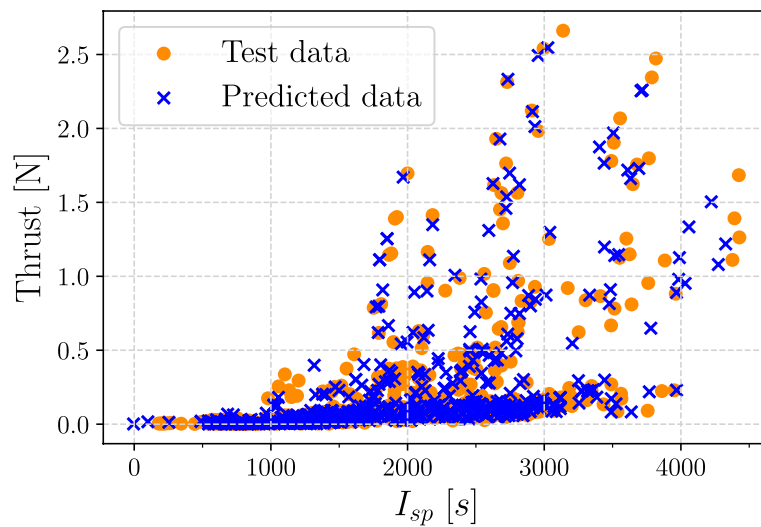


Fig. 9 Model prediction of thrust and specific impulse on testing data

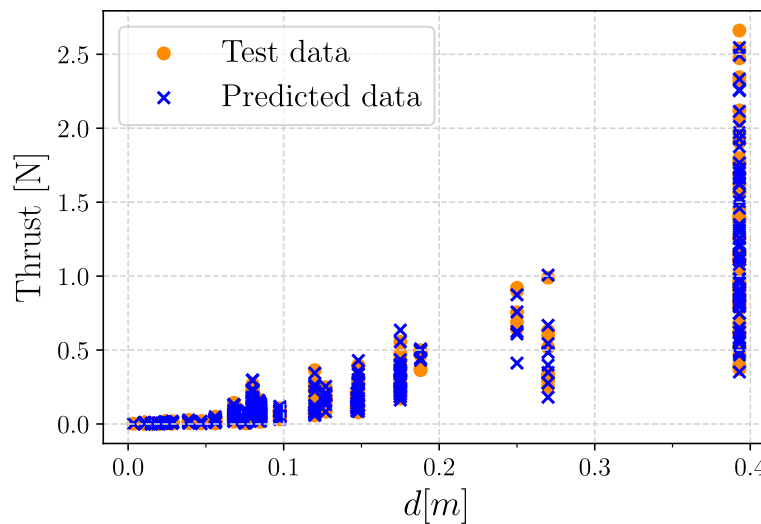


Fig. 10 Gradient Boosting Regression predictions of the thrust against mean diameter for the testing set

tested on 576 samples, while the scaling law model in Fig. 11 on 493 samples. The difference in prediction quality is obvious. In Fig. 10 the model predicts thrust values very accurately, whereas in Fig. 11 agreement between data and calculation is poor. Moreover, the model follows a parabolic trend due to the quadratic dependence of thrust on diameter and fails to capture the punctual distribution of the test data. Figure 12 illustrates the absolute errors of the two models, in a graph similar to the one in Fig. 6. The errors of the Gradient Boosting are very small and the density curve goes fast to zero. On the contrary, the scaling law errors are almost two orders of magnitude greater.

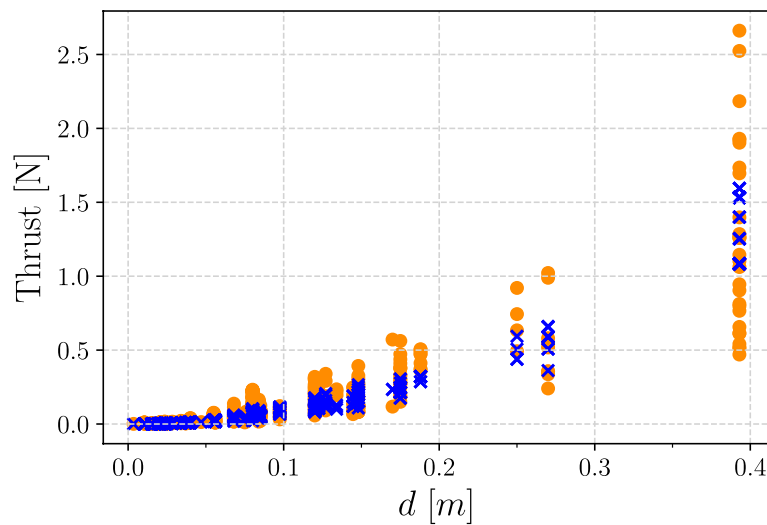


Fig. 11 Scaling law predictions of the thrust with respect to mean diameter values for the testing set

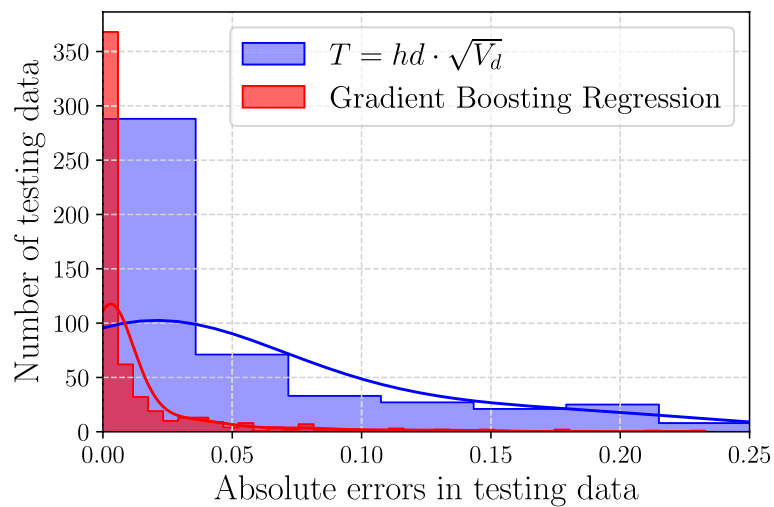


Fig. 12 Comparison of absolute errors for the GBR model and the scaling relation when predicting the $T = f(d)$ behavior

Procedure to design a new thruster

Introduction

As explained in [Introduction](#) section, in the process of building a new thruster some parameters are specified by mission requirements. Usually, thrust and specific impulse are the initial design parameters from which the other characteristic parameters of the thruster are determined, as the geometrical sizes, voltage, mass flow rate and magnetic field. The Gradient Boosting model predicts accurately thrust and specific impulse values, knowing the other parameters. However, when the thruster design is considered, a back-propagation problem arose. Indeed the model should be able to reconsider the relationship generation process underlying the creation of decision trees in order to extrapolate information about the various parameters when a particular output is

desired. Since Gradient Boosting is built with sequential decision trees and each one splits the parameters in different ways finding relations among them, the challenge is practically unfeasible, and the problem must be tackled in another way. The solution was found in an analytical approach based on an optimization problem. The database was initially divided based on the propellant type. Furthermore, the magnetic field was inserted secondarily in the model, as described in [Magnetic field](#) section, as it plays a key role in the thrust process although very little data is available, resulting in reduced model performance.

Analytical approach

The problem of determining the parameter values when a certain thrust and a certain specific impulse are required is addressed by the analytical approach through an optimization algorithm. The way this approach proceeds is the following: first, a desired output is provided. The Gradient Boosting model then takes random inputs and calculates thrust and specific impulse values. After this, the error between the output found and the desired output is calculated. The task of the optimization algorithm is then to modify the values of the input parameters to minimize the loss function represented by the absolute error between the two output pairs. Once the error is minimized, the model found the best parameters that provide the required outputs. This procedure is summarized in Fig. 13, where the box 'Differential Evolution' represents the optimization algorithm.

The choice of the optimization algorithm is mainly based on the loss function minimization procedure. In fact, it is a stochastic algorithm that does not require the optimization problem to be differentiable, as is required by other classic algorithms as gradient descent [45]. Since the GBR consists of sequential decision trees, it is impossible to determine how the model depends on the various parameters individually and therefore it is not feasible to calculate their partial derivatives. Instead, Differential Evolution optimizes the problem iteratively, trying to improve a candidate solution [46]. The working principle of this algorithm is given below. Only the main steps are described; a full explanation can be found in [46] and [47]. The Differential Evolution (DE) is based on four steps, schematically represented in Fig. 14.

In the initialization phase, random values are generated for each parameter, creating vectors made by random numbers. After, for each of these vectors, three other ones are chosen to generate respective mutant vectors, the so-called Mutation phase. Considering the vector a , its mutant vector V is:

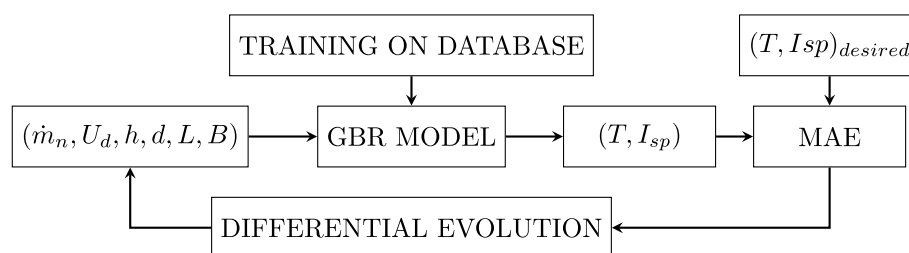


Fig. 13 Flowchart of the analytical approach

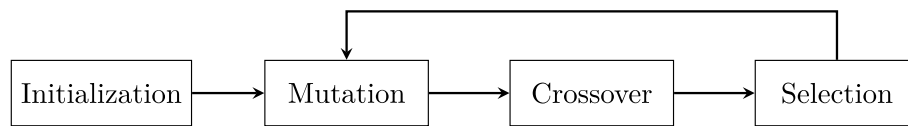


Fig. 14 Schematic representation of DE

$$V = b + F(c - d) \quad (12)$$

where b , c and d are three vectors chosen initially and F is a scaling factor that controls the mutation process, with a value in the range $[0,1]$. The performances of the new vectors are evaluated then, and a certain threshold is decided, the Crossover: if the error is less than the crossover value, the mutant vector is changed with the old one. There is therefore a recombination of the initial arrays where some places are occupied by initial vectors, while other by mutant ones. The same procedure is repeated until a certain threshold is reached or after a certain amount of iterations. Bounds within which the algorithm is guided in searching the optimal inputs are defined. The database was divided in subgroups and the DE was looking for parameters values only in the subgroup containing values able to generate that desired output. Different algorithm were developed for the three propellants and also for the consideration or not of the magnetic field. Naturally, the more the outputs are far from values present in the database, the worst are the performances of Differential Evolution algorithm, struggling to converge. However, in the various cases we tried, the algorithm was performing quite well, as shown in [Final example](#) section.

Final example

To help understand how the approach works, it is useful to show an example for determining the characteristics of two new thrusters: one for a possible low-power HT (Example A) and the other for a high-power HT (Example B), both working with xenon as propellant. The requirements in terms of thrust and specific impulse are reported in Table 6. The characteristics of the thrusters are obtained with the analytical approach, determining the mass flow rate \dot{m}_n , discharge voltage U_d , diameter d , height h and length L of the channel. In [Magnetic field](#) section, the magnetic field B is also added. The approach is based on the application of the diagram shown in Fig. 13. It is worth noting that the mean absolute errors are calculated between scaled outputs, since the GBR is working on a scaled set of inputs. At the end, the parameters are unscaled to be shown in the appropriate way. Finally, for a better understanding of the discrepancy between desired and predicted values, the error is presented as a percentage and separately for the two outputs, although the algorithm optimizes their average.

Table 6 Desired outputs

	$T[N]$	$I_{sp}[s]$
Example A	0.02	1900
Example B	1	4000

For the Example A, the differential evolution algorithm takes 84 iterations to find the minimum values of the MAE. The percentage error on the thrust is 0.14 % while on the specific impulse is 0.002 %. The output obtained are therefore almost identical to the ones desired. Whereas, for Example B, the algorithm uses 33 iterations and, while the percentage error for the thrust is 0.023 %, for the specific impulse it is 20.68 %. Basically, the optimization stop to converge at 3188 s of I_{sp} . This is due to the limited amount of data in the range selected. The approach in fact relies on data: the more data is available, the more precise is the model. Table 7 illustrates the optimal inputs found with the analytical approach for the two cases. Furthermore, in the table are present also values corresponding to real thrusters taken from our database that were giving quite the same values of thrust and specific impulse, as an help to understanding the physical meaningfulness of the results obtained. The outputs required are recalled. In conclusion, it can be stated that the performance of the optimization algorithm is highly contingent upon the availability of data. Specifically, in the case of Low-Power Hall Thrusters, the algorithm demonstrates robust performance, successfully iterating numerous times to progressively reduce error and achieve the desired values. Conversely, in the scenario involving High-Power HT, the algorithm encounters difficulties in locating available data that simultaneously satisfy both thrust and specific impulse requirements, resulting in convergence with a significantly higher error in specific impulse.

Magnetic field

The magnetic field B has been treated separately, as it plays a crucial role in the performance of a thruster [1]. But the very limited data availability does not allow it to be considered as one of the main parameters in the model. In fact, tests available when the magnetic field is considered amount to 1000, while without the number is 1918. Furthermore, the indicated value only represents the maximum intensity on the channel axis, when what counts is the the entire topology of the magnetic field. This results in a loss of performances of the model that can be directly seen in Figs. 15 and 16. It is clear that the model predictions no longer match the actual data as closely as they used to without B , see Figs. 9 and 10.

Table 7 Parameters values obtained with the analytical approach

	Example A	Real LP HT	Example B	Real HP HT
Prop.	Xe	Xe	Xe	Xe
\dot{m}_p [mg/s]	0.49	1.5	32.36	29.59
U_d [V]	620.4	700	551.73	550
d [mm]	57.07	68	259.48	250
h [mm]	10.38	12	37.93	40
L [mm]	18.04	11	35.87	35
T % _{err}	0.14 %	-	0.023 %	-
I_{sp} % _{err}	0.002 %	-	20.68 %	-
iter	84	-	33	-
$T_{desired}$ [N]	0.02	-	1	-
$I_{sp,desired}$ [s]	1900	-	4000	-
T_{actual} [N]	0.0198	0.024	0.99	0.92
$I_{sp,actual}$ [s]	1869.9	1893.5	3188	3171

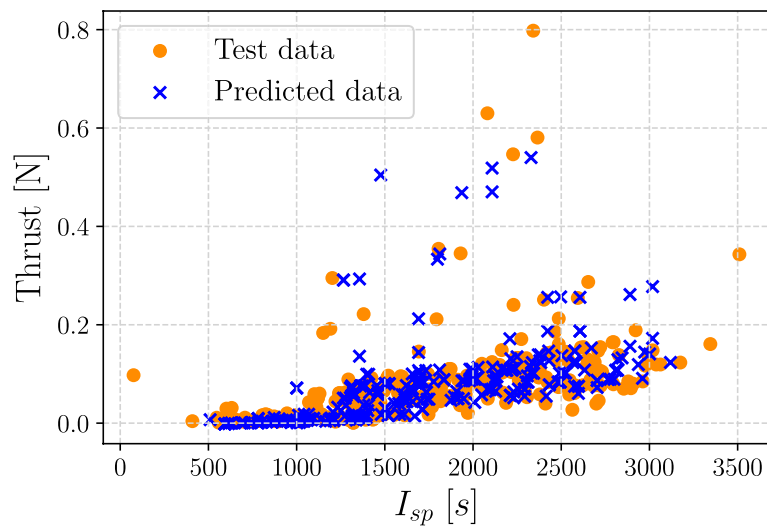


Fig. 15 Model prediction of thrust and specific impulse on data including the magnetic field

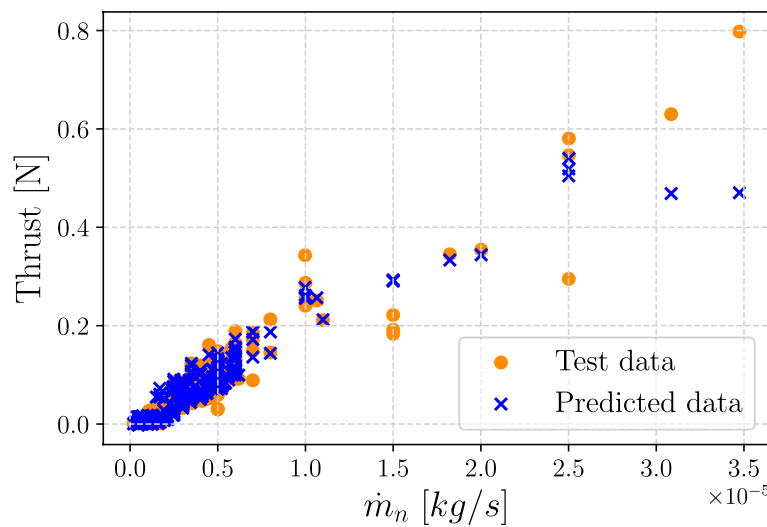


Fig. 16 Predicted thrust as a function of the mass flow rate for data that include magnetic field values

However, an attempt was made introducing the magnetic field as a model parameter for examples A and B previously described. Table 8 shows the optimal parameters values found with the optimization. The errors are higher as expected and the values are changed, due to less data available for the GBR model to learn. Magnetic field values are given and, as made before, results are compared to real tests present in the database.

In general, the errors are smaller when the magnetic field is not taken into account, but there is still a large difference between the low-power and high-power cases. While the algorithm performs more accurately in the former case where the outputs obtained are close to the desired ones, in the latter, represented by Example B, it struggles to converge. In fact, no data are available in the working range requested and therefore the model is forced to consider data where the specific impulse is of 2777 s, since the error is

of 31.15 % as it's possible to see in Table 8. The poor performances regard the geometrical features mainly, where the model finds difficult to generalize from existent thrusters.

Conclusion

The aim of this work was to develop a model capable of determining the geometrical characteristics and operating parameters of a new Hall thruster, based on the use of Machine Learning algorithms. For this purpose, a database of 54 thrusters was created, collecting data from previous databases and publications. The database was initially validated through a statistical approach based on scaling laws extensively discussed in previous works. Scaling relations served as a starting point in the creation of a supervised machine learning model based on the Gradient Boosting Regression algorithm. This model is highly efficient in the process of predicting thrust and specific impulse values when the magnetic field is not considered in the model input parameters, as its presence reduces the amount of data available and, consequently, the performance of the model itself. The Machine Learning model is capable of combining all the parameters with each other by learning relationships that allow accurate determination of thrust and specific impulse values for data never seen before. A backpropagation problem was then tackled using an analytical method to generate a procedure capable of guiding the process of building a new thruster. The advantages of using this model lie mainly in the possibility of avoiding initial assumptions related to loss factors or dependencies between geometric parameters. Naturally this approach relies completely on data. Although data used by the model is derived from tests carried out on thrusters built using the classical approach, the model nevertheless recognizes dependencies between parameters, in particular between geometric ones. The model performs very well in the range available for training, while the extrapolation remains less accurate where there is a lack of data. It is important to clarify that the Gradient Boosting model was meticulously constructed and demonstrates robustness in predictions on unseen data. However, what slightly undermines

Table 8 Parameters values obtained with the analytical approach including the magnetic field

	Example A	Real LP HT	Example B	Real HP HT
Prop.	Xe	Xe	Xe	Xe
\dot{m}_p [mg/s]	0.96	1	39.55	39.77
U_d [V]	391.87	350	612.58	500
d [mm]	38.27	39	270	270
h [mm]	10.65	11	50	50
L [mm]	25	25	70	70
B [G]	177.59	180	121.15	130
T % _{err}	52.11 %	-	5.25 %	-
I_{sp} % _{err}	2.87 %	-	31.15 %	-
iter	24	-	14	-
$T_{desired}$ [N]	0.02	-	1	-
$I_{sp\,desired}$ [s]	1900	-	4000	-
T_{actual} [N]	0.03	0.019	0.947	1.047
$I_{sp\,actual}$ [s]	1847.6	1924.78	2777	2683.37

the model is the optimization algorithm, which heavily relies on the available data. Consequently, the error increases significantly when the algorithm is required to optimize in regions with sparse or absent data. Nonetheless, the adoption of this algorithm was the only method found to fulfill the objective of determining the characteristics of new Hall thrusters based on a supervised Machine Learning model. This suggests that future studies should first aim to reduce the rigidity of the constraints applied during optimization to enable the use of a broader dataset. However, care must be taken to ensure that the solutions obtained remain physically feasible. Besides, the utilization of neural networks might mitigate or improve the issues associated with backpropagation.

However, this work clearly demonstrates the interest and power of a Machine-Learning-based approach to defining scaling laws and, what is new, predicting the geometric characteristics and operating points of a Hall thruster according to mission requirements. The approach developed and validated here, although complex, remains accessible and relatively simple to implement, since the tools used, notably the Gradient Boosting Regression and the tree generation, exist in different versions and languages.

Finally, our work shows that the quantity and quality of data is the key to obtaining accurate and reliable results. It is therefore essential today to generate databases containing a large amount of data over a wide power range, either through measurements, which are nevertheless long and costly, or through numerical simulations to help engineers build the next generation of Hall thrusters.

Authors' contributions

All authors contributed to the study conception and design. Code writing and data analysis were performed by A.M. The first draft of the manuscript was written by A.M. and S.M. All authors read and approved the final manuscript.

Funding

This work was carried out thanks to internal laboratory funding.

Availability of data and materials

Data will be made available on request.

Declarations

Competing interests

The authors declare no competing interests.

Received: 18 April 2024 Accepted: 21 July 2024

Published online: 01 August 2024

References

1. Boeuf JP (2017) Tutorial: Physics and modeling of hall thrusters. *J Appl Phys* 121(1):6–7
2. Mazouffre S (2016) Electric propulsion for satellites and spacecraft: established technologies and novel approaches. *Plasma Sources Sci Technol* 25(3):033002
3. Andrenucci M, Biagioni L, Marcuccio S, Paganucci F (2003) Fundamental scaling laws for electric propulsion concepts. In: 28th International Electric Propulsion Conference. p 1721
4. Shagayda AA (2014) On scaling of hall effect thrusters. *IEEE Trans Plasma Sci* 43(1):12–28
5. Dannenmayer K, Mazouffre S (2011) Elementary scaling relations for hall effect thrusters. *J Propuls Power* 27(1):236–245
6. Burkov A (2019) The hundred-page machine learning book, vol 1, 1st edn. Andriy Burkov, Quebec City
7. Plyashkov YV, Shagayda AA, Kravchenko DA, Lovtsov AS, Ratnikov FD (2022) On scaling of hall-effect thrusters using neural nets. *J Propuls Power* 38(6):935–944
8. Szabo JJ, Tedrake R, Metivier E, Paintal S, Taillefer Z (2017) Characterization of a One Hundred Watt, Long Lifetime Hall Effect Thruster for Small Spacecraft. In: 53rd AIAA/SAE/ASEE Joint Propulsion Conference. American Institute of Aeronautics and Astronautics, Atlanta. <https://doi.org/10.2514/6.2017-4728>

9. Hargus W, Nakles M (2008) Ion Velocity Measurements Within the Acceleration Channel of a Low-Power Hall Thruster. *IEEE Trans Plasma Sci* 36(5):1989–1997. <https://doi.org/10.1109/TPS.2008.2003967>
10. De Grys K, Welander B, Dimicco J, Wenzel S, Kay B, Khayms V, Paisley J (2005) 4.5 Kw Hall Thruster System Qualification Status. In: 41st AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. American Institute of Aeronautics and Astronautics, Tucson. <https://doi.org/10.2514/6.2005-3682>
11. De Grys K, Rayburn C, Haas J (2003) Study of Power Loss Mechanisms in BPT-4000 Hall Thruster. In: 39th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit. American Institute of Aeronautics and Astronautics, Huntsville. <https://doi.org/10.2514/6.2003-5277>
12. Kronhaus I, Kapulkin A, Balabanov V, Rubanovich M, Guelman M, Natan B (2013) Discharge Characterization of the Coaxial Magnetoisolated Longitudinal Anode Hall Thruster. *J Propuls Power* 29(4):938–949. <https://doi.org/10.2514/1.B34754>
13. Kronhaus I, Kapulkin A, Guelman M, Natan B (2012) Investigation of two discharge configurations in the CAMILA Hall thruster by the particle-in-cell method. *Plasma Sources Sci Technol* 21(3):035005. <https://doi.org/10.1088/0963-0252/21/3/035005>
14. Kronhaus I, Kapulkin A, Balabanov V, Rubanovich M, Guelman M, Natan B (2012) Investigation of physical processes in CAMILA Hall thruster using electrical probes. *J Phys D Appl Phys* 45(17):175203. <https://doi.org/10.1088/0022-3727/45/17/175203>
15. Grimaud L (2018) Magnetic shielding topology applied to low power hall thrusters. PhD thesis, Université d'Orléans
16. Mazouffre S, Grimaud L (2018) Characteristics and performances of a 100-w hall thruster for microspacecraft. *IEEE Trans Plasma Sci* 46(2):330–337
17. Bugrova A, Desiatskov A, Kaufman H, Kharchevnikov V, Morozov A, Zhurin V (2001) Design and experimental investigation of a small closed drift thruster. In: Proceedings of the International Electric Propulsion Conference. Pasadena, pp 15–19
18. Lopez Ortega A, Mikellides IG, Conversano R, Lobbia RB, Chaplin VH (2019) Plasma simulations for the assessment of pole erosion in the Magnetically Shielded Miniature Hall Thruster (MaSMi). Pasadena, Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2019
19. Conversano RW, Goebel DM, Hofer RR, Arora N (2017) Performance enhancement of a long-life, low-power hall thruster for deep-space smallsats. In: 2017 IEEE Aerospace Conference. IEEE Xplore, New York City, pp 1–12
20. Conversano RW, Lobbia RB, Kerber TV, Tilley KC, Goebel DM, Reilly SW (2019) Performance characterization of a low-power magnetically shielded hall thruster with an internally-mounted hollow cathode. *Plasma Sources Sci Technol* 28(10):105011
21. Conversano RW, Goebel DM, Hofer RR, Mikellides IG, Wirz RE (2017) Performance analysis of a low-power magnetically shielded hall thruster: Experiments. *J Propuls Power* 33(4):975–983
22. Khayms V, Martinez-Sanchez M (1996) Design of a miniaturized hall thruster for microsatellites. In: 32nd Joint Propulsion Conference and Exhibit. p 3291
23. Kamhawi H, Haag T, Jacobson D, Manzella D (2011) Performance evaluation of the nasa-300m 20 kw hall thruster. In: 47th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. p 5521
24. Kamhawi H, Huang W, Haag T, Shastry R, Soulas G, Smith T, Mikellides I, Hofer R (2013) Performance and thermal characterization of the nasa-300ms 20 kw hall effect thruster. In: International Electric Propulsion Conference (IEPC 2013), GRC-E-DAA-TN11609
25. Bernikova MY, Gopanchuk V (2017) Parametric family of the plas-type thrusters: development status and future activities. In: 35th International Electric Propulsion Conference. Georgia Institute of Technology, Atlanta, pp 8–12
26. Potapenko MY, Gopanchuk V, Olotin S (2015) Plas-40 development status: New results. In: 34th International Electric Propulsion Conference
27. Piragino A, Giannetti V, Reza M, Faraji F, Ferrato E, Kitaeva A, Pedrini D, Andreussi T, Paganucci F, Andreucci M (2019) Development status of sitael's 20 kw class hall thruster. In: AIAA Propulsion and Energy 2019 Forum. p 3812
28. Piragino A, Ferrato E, Faraji F, Reza M, Giannetti V, Kitaeva A, Pedrini D, Andreucci M, Andreussi T (2019) Sitaels magnetically shielded 20 kw hall thruster tests. In: Proceedings of the 36th International Electric Propulsion Conference. Vienna, pp 15–20
29. Arkhipov B, Maslennikov N, Murashko V, Veselovzorov A, Morozov A, Pokrovsky I, Gavryushin V, Khartov S, Kim V, Kozlov V (1993) Development and investigation of characteristics of increased power spt models. paper iepc-93-222. In: 23rd International Electric Propulsion Conference. Seattle (USA), 13-16 September. p 2087
30. Manzella D, Jacobson D, Jankovsky R (2001) High voltage spt performance. In: 37th Joint Propulsion Conference and Exhibit. p 3774
31. Misuri T, Andreucci M (2008) Het scaling methodology: Improvement and assessment. In: 44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. p 4806
32. Jacobson D, Jankovsky R (1998) Test results of a 200w class hall thruster. In: 34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit. p 3792
33. Saevets P, Semenenko D, Albertoni R, Scremin G (2017) Development of a long-life low-power hall thruster. In: The 35th International Electric Propulsion Conference. pp 1–11
34. Hofer R, Jankovsky R (2001) A hall thruster performance model incorporating the effects of a multiply-charged plasma. In: 37th Joint Propulsion Conference and Exhibit. p 3322
35. Morozov A, Savelyev V (2000) Fundamentals of stationary plasma thruster theory. *Rev Plasma Phys* 21:203–391
36. Zhurin VV, Kaufman HR, Robinson RS (1999) Physics of closed drift thrusters. *Plasma Sources Sci Technol* 8(1):R1
37. Ahedo E, Gallardo J (2003) Scaling down hall thrusters. In: 28th Int. Electric Propulsion Conf. Toulouse, pp 2003–104
38. Daren Y, Yongjie D, Zhi Z (2005) Improvement on the scaling theory of the stationary plasma thruster. *J Propuls Power* 21(1):139–143

39. Misuri T, Battista F, Barbieri C, De Marco EA, Andrenucci M, et al (2007) High power hall thruster design options. In: Proceedings of the 30th International Electric Propulsion Conference (Florence). IEPC, Florence, pp 07–311
40. Lee E, Kim Y, Lee H, Kim H, Doh G, Lee D, Choe W (2019) Scaling approach for sub-kilowatt hall-effect thrusters. *J Propuls Power* 35(6):1073–1079
41. Mazouffre S, Bourgeois G, Dannenmayer K, Lejeune A (2012) Ionization and acceleration processes in a small, variable channel width, permanent-magnet hall thruster. *J Phys D Appl Phys* 45(18):185203
42. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. *J Mach Learn Res* 12:2825–2830
43. Russell SJ, Norvig P (2010) Artificial intelligence a modern approach, 4th edn. Pearson, London
44. Friedman JH (2002) Stochastic gradient boosting. *Comput Stat Data Anal* 38(4):367–378. *Nonlinear Methods and Data Mining*
45. Ilonen J, Kamarainen JK, Lampinen J (2003) Differential evolution training algorithm for feed-forward neural networks. *Neural Process Lett* 17:93–105
46. Feoktistov V (2006) Differential evolution, 1st edn. Springer, New York
47. Ahmad MF, Isa NAM, Lim WH, Ang KM (2022) Differential evolution: A recent review based on state-of-the-art works. *Alex Eng J* 61(5):3831–3872

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.