Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemometrics

MADGUI: Multi-Application Design Graphical User Interface for active learning assisted by Bayesian optimization

Christophe Bajan, Guillaume Lambard *

Center for Basic Research on Materials, Data-driven Materials Design Group, National Institute for Materials Science, 1-1 Namiki, Tsukuba, Ibaraki, 305-0044, Japan

ARTICLE INFO

Keywords:
Graphical user interface
Statistical analysis
Machine learning
Bayesian optimization
Process optimization
User-friendly

ABSTRACT

We present MADGUI, Multi-Application Design Graphical User Interface (GUI) using Bayesian Optimization and prediction model for data analysis and optimize process or composition. Its strength is its user-friendly design, which requires no programming knowledge. It is built using the Streamlit library in Python and is divided into three parts, allowing users to select various parameters and fill csv/xlsx files without any coding required. Overall, MADGUI is designed as an optimal experiment design platform with active machine learning, which accelerates the discovery of optimal solutions and provides an intuitive GUI for users with no experience in coding, machine learning, or optimization.

1. Introduction

In the evolving landscape of scientific research and process engineering, the integration of machine learning (ML) techniques has emerged as a transformative tool, aiding in analyzing and optimizing various experimental designs and process pipelines, leading to improved target returns. Particularly in materials science, Wu et al. [1] note that integrating ML and computer science with current material synthesis and optimization approaches will effectively speed up the discovery of high-performance photoactive materials in organic solar cells. Similarly, Deringer et al. [2] demonstrate the current enabling of new degrees of realism in material science through ML, with atomic-scale computer simulations focusing on quantum-mechanical simulations as the central approach. According to Rodrigues et al. [3], big data and ML applied to materials science can accelerate the solution of intricate chemical problems and even solve problems that would otherwise not be tractable. Thus, ML can accelerate the discovery of materials with tailored properties and lead to rapid predictive modelling [4,5].

During this endeavour, material experimentalists, theoreticians and engineers attempt to predict the relationship between the composition of a material and its intrinsic physicochemical properties, to tune controllable parameters involved in the processing pipeline of a material, or more generally, to choose the correct statistical technique to properly analyze and gather data necessary to test a hypothesis or perform a data-driven improvement of a material's properties. However, the usage of ML techniques and more classic statistical tools have traditionally required significant programming knowledge and setup, making them less accessible to scientists who may not have a background in computer science [1]. Furthermore, deploying ML for new projects often involves starting from scratch or adapting a previously used ML model, which can be time-consuming and prone to errors for non-practitioners [1].

Recognizing challenges in applying ML and statistics in various past academic and industrial collaborations, we have developed MADGUI [6], Multi-Application Design Graphical User Interface, which aims to democratize access to applied statistics, ML and optimization strategy by providing a user-friendly platform that does not require any programming knowledge. Developed using Python libraries, including Streamlit [7] for the GUI development, Scikit-learn [8] for a Random Forests ML algorithm and a k-fold cross-validation strategy for models evaluation, Seaborn [9] for the graphical representation, XGBoost [10] for a distributed gradient-boosted decision tree algorithm, and GpyOpt [11] for Bayesian Optimization (BO), MADGUI offers a powerful tool for researchers and engineers desirous of engaging in statistical analysis and optimization tasks assisted by ML and BO for a wide range of applications. Indeed, MADGUI is a user-friendly, no-code interface built upon a robust foundational method known as active learning, which ML and BO enhance. This approach, termed Active learning assisted by machine learning and Bayesian optimization (ALMLBO), is not pre-trained on any particular domain. Instead, it leverages the user's dataset to facilitate learning. The ML component of MADGUI actively engages with the data to predict outcomes and to determine the significance of different input

* Corresponding author.
E-mail addresses: BAJAN.Christophe@nims.go.jp (C. Bajan), LAMBARD.Guillaume@nims.go.jp (G. Lambard).

https://doi.org/10.1016/j.chemolab.2025.105323
Received 4 June 2024; Received in revised form 31 October 2024; Accepted 18 January 2025
Available online 21 January 2025

features. Then, the BO element strategically suggests new data points to refine the learning process and eventually reach the objective(s) after a few cycles of learning from the updated dataset, proposing protocols related to the studied system and performing consequent experimental characterizations. This process has shown promise in designing new materials with exceptional properties across various applications. Notably, it led to improvements in the adhesive strength, electrocatalytic activity in methanol oxidation, charge/discharge performance, and maximum energy product for epoxy resins [12], mesoporous trimetallic PtPdAu alloy films [13], electrolyte composition of Li–O_2 batteries [14], and hot-extruded Nd-Fe-B anisotropic magnets [15], respectively. The versatility of Bayesian optimization extends beyond materials science, with successful applications in various engineering fields. For instance, Lam et al. [33] demonstrated its effectiveness in aerospace engineering, while Gongora et al. [34] developed a Bayesian experimental autonomous researcher (BEAR) for mechanical design. These examples further highlight the broad applicability of this technique across different domains of science and engineering.

Notably, MADGUI possesses the BO [16,17,32] at its core. BO solves combinatorial optimization problems by minimizing the number of experimental trials required to identify the optimum on an objective function f of an unknown form. The objective function f generally links input features, e.g., composition and process parameters, of a material with its output target properties, e.g., adhesive strength or charge/discharge performance, and determining its optimum is equivalent to locating a set of input features leading to improved material's performances. Therefore, the classic trial-and-error experimental setting is directed by the experiments proposed by the BO to improve a material's performance.

There are several alternatives to MADGUI for performing ALMLBO, like scikit-optimize [20], pyGPGO [21], or bayes_opt [22]. However, these alternatives do not possess a dedicated graphical user interface (GUI) and demand a minimum of knowledge in coding to set them up on an individual machine. Closer to MADGUI, AutoOED [23] and BOXVIA [24] software are much more accessible. AutoOED is a design platform working with automated machine learning to accelerate the discovery of optimal solutions, with a GUI for visualization and guiding experiments. And BOXVIA is a GUI-based application that allows users to optimize objective functions and visualize the process. AutoOED and BOXVIA are based on Bayesian optimization for finding optima in a minimum number of experimental evaluations and can be used through their executable or source code. However, they both lack basic visualization of the input data, e.g., the evolution of target properties over evaluated samples, multiple feature-property one-to-one scatter plots, evaluation of one-to-one linear statistical correlations, or the potential to perform a selection of essential features through the assessment of prediction performance of target properties through an autonomous ML pipeline.

MADGUI, on the other hand, display a visualization of the input data, the evolution of target properties over evaluated samples, multiple feature-property one-to-one scatter plots, evaluation of one-to-one linear statistical correlations, the potential to perform a selection of important features through the assessment of prediction performance of target properties through an autonomous ML pipeline and the possibility to add limits and constraints to the features for the BO.

Working in material science, we developed MADGUI to help us in this area. However, it is not limited to this domain; it can generally be used for various research areas as long as features and targets exist. Examples of features can be temperature, time, mixing speed, quantity, equipment A or B, type of reactant, etc.

MADGUI is freely available for public use via the Streamlit cloud share system [25] at https://lambard-ml-team-madgui.streamlit.app/. It is available as an executable for Windows, Mac M1 and Mac Intel at https://madgui.odoo.com/. We are confident that this tool will contribute significantly to scientific discovery and process engineering by making ALMLBO more accessible and manageable for users across various domains of expertise and applications.

MADGUI is designed for real-time use, with its web-based interface allowing immediate access and quick data processing. A dataset of around 100 samples takes around 10 minutes from the inference step to propose new experimental samples on a regular laptop or desktop computer. MADGUI is highly cost-efficient as its freely available and typically enables users to gain an order of magnitude in time and cost for experiments and characterization [31].

This paper is organized as follows: First, we describe the software architecture of MADGUI. Then, we provide a detailed description of its main features, including the main page, prediction page, and Bayesian optimization page. Finally, we discuss future developments and conclude with the potential impact of MADGUI on various research fields.

2. Software architecture

MADGUI [6] is a user-friendly GUI developed for statistical data analysis and ML modelisation. MADGUI is entirely written in Python. It uses the Streamlit library [7] to create the web-based interface.

Here's a brief overview of the software architecture.

1. Importing libraries: The software utilizes a range of libraries, including Streamlit for the GUI, Pandas [26] and Numpy [27] for data manipulation, Scikit-learn and XGBoost for ML models and k-fold cross-validation model evaluations, Matplotlib [28] and Seaborn for graphical visualization, and GpyOpt for the BO.
2. Setting up the interface: The software uses Streamlit to set up a web-based interface. It defines containers for various interface sections, such as the header, dataset, feature selection, correlation, prediction, and Bayesian optimization.
3. Data upload: Users can upload their data in CSV or Excel format. The software reads the data and stores it in a Streamlit Dataframe.
4. Features and targets selection: Users can select columns from their dataset for analysis. They choose which columns are features to analyze and which are targets to predict or improve.
5. Navigation: The software provides a navigation sidebar that allows users to navigate between different interface sections. These sections include the main, prediction, Bayesian optimization, about, and contact pages for the online version.
6. Data analysis and ML: The software uses various ML algorithms (ElasticNet [8], Random Forests [18], XGBoost [10]) from the Scikit-learn [8] library for prediction and optimization tasks. It also provides functionality for data visualization using libraries like Matplotlib and Seaborn.
7. Optimization: The software uses the GPyOpt library [11] for Bayesian optimization, a strategy for finding the maxima/minima of an objective function using the expected improvement (EI) [19].
8. Reset functionality: The software provides a reset button that clears all the session states, allowing users to start their analysis from scratch.

The flowchart in Fig. 1 depicts the architecture of MADGUI.

3. Software description

3.1. Main page

The Main page is the initialization hub, where users can find a comprehensive guide on effectively utilizing the GUI. It begins by instructing users on the proper data formatting required for the GUI to function correctly. The data should not contain any blank cells or non-numerical values. Users are advised to convert any non-numerical values into categorical values (e.g., “Low”, “Medium”, and “High” become 1, 2, and 3, respectively). Once the dataset is prepared, users can upload the data (in CSV or XLSX formats) using the sidebar button. The GUI then displays a dataframe of the uploaded dataset, allowing users to

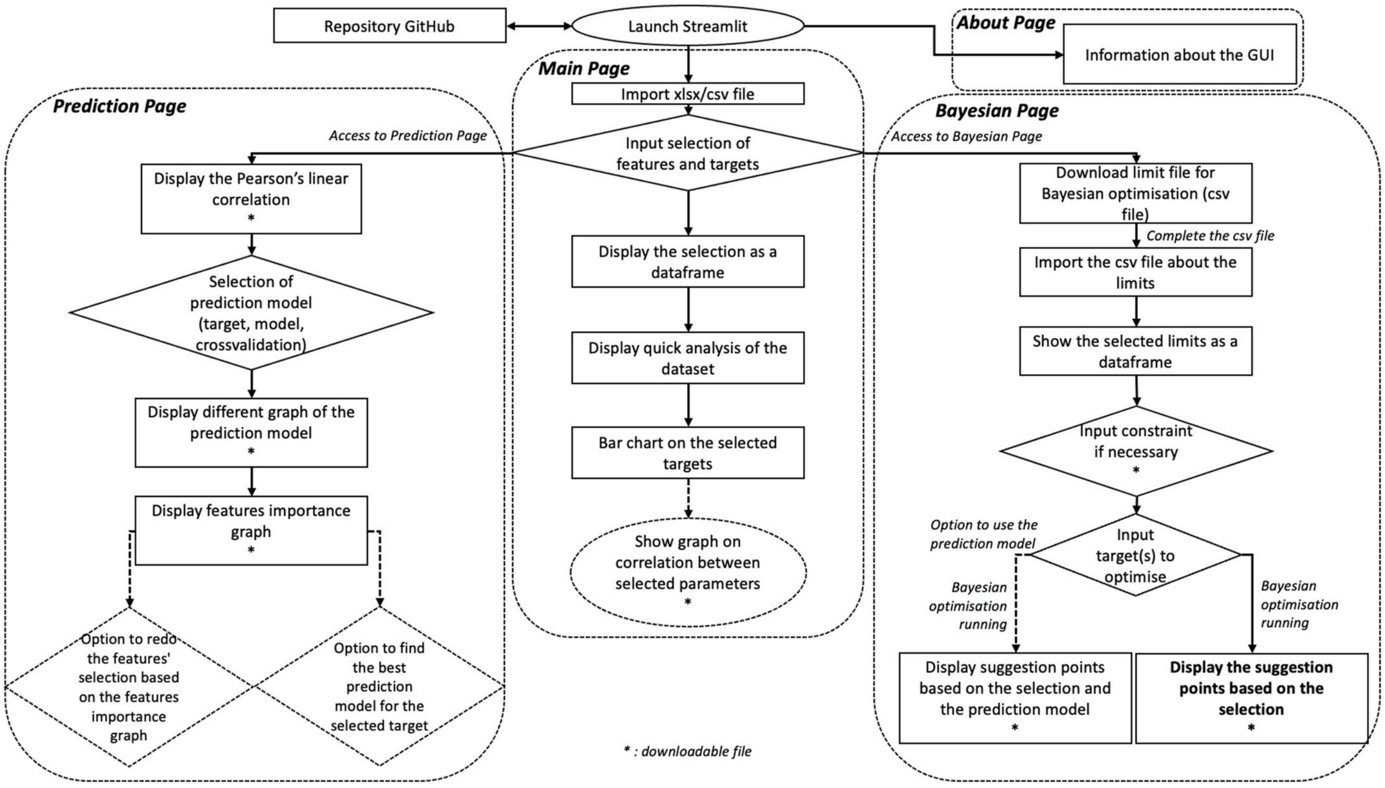
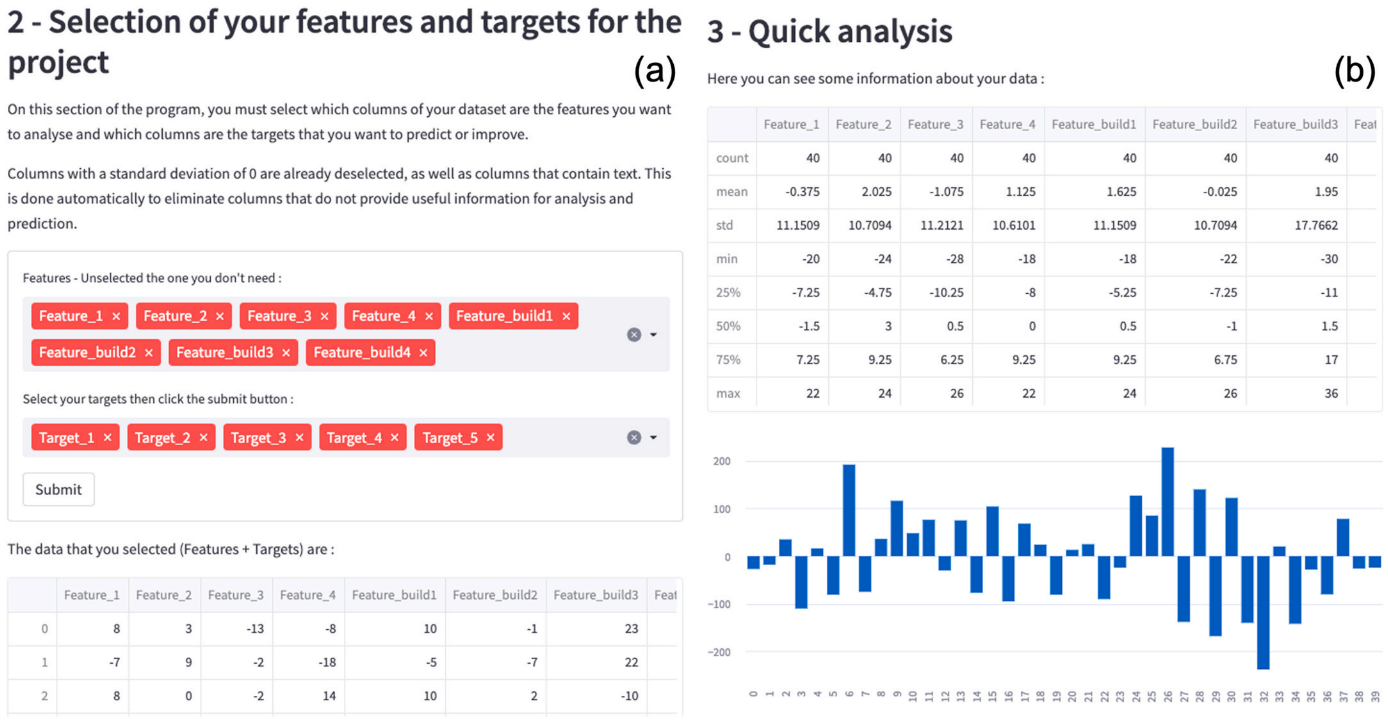

Repository GitHub
Launch Streamlit
About Page
Information about the GUI
Main Page
Import xlsx/csv file
Input selection of features and targets
Display the selection as a dataframe
Display quick analysis of the dataset
Bar chart on the selected targets
Show graph on correlation between selected parameters *
Prediction Page
Access to Prediction Page
Display the Pearson's linear correlation *
Selection of prediction model (target, model, crossvalidation)
Display different graph of the prediction model *
Display features importance graph *
Option to redo the features' selection based on the features importance graph
Option to find the best prediction model for the selected target
Bayesian Page
Access to Bayesian Page
Download limit file for Bayesian optimisation (csv file)
Complete the csv file
Import the csv file about the limits
Show the selected limits as a dataframe
Input constraint if necessary *
Option to use the prediction model
Input target(s) to optimise
Bayesian optimisation running
Bayesian optimisation running
Display suggestion points based on the selection and the prediction model *
Display the suggestion points based on the selection *
* : downloadable file

Fig. 1. Flowchart of MADGUI, displaying the mechanism behind each page.

verify its correctness. The most crucial part of the Main page is the selection of features and targets. Features are parameters that can be manipulated during an experiment, while targets are the measured values that the user aims to interpret and optimize. The system cannot automatically determine which columns are features and which are targets. Therefore, users must manually specify this distinction. For this purpose, the GUI displays two multi-select panels (Fig. 2 a). The first panel contains all the column names, excluding columns where the standard deviation is 0 (indicating no change in the value and hence no information). Users are expected to unselect the column names that are targets or irrelevant (e.g., the sample enumeration). In the second panel, the user must select targets, which can be one or multiple. It's important



2 - Selection of your features and targets for the project

On this section of the program, you must select which columns of your dataset are the features you want to analyse and which columns are the targets that you want to predict or improve.

Columns with a standard deviation of 0 are already deselected, as well as columns that contain text. This is done automatically to eliminate columns that do not provide useful information for analysis and prediction.

Features - Unselected the one you don't need :

Select your targets then click the submit button :

The data that you selected (Features + Targets) are :

	Feature_1	Feature_2	Feature_3	Feature_4	Feature_build1	Feature_build2	Feature_build3	Feat
0	8	3	-13	-8	10	-1	23	
1	-7	9	-2	-18	-5	-7	22	
2	8	0	-2	14	10	2	-10	

3 - Quick analysis

Here you can see some information about your data :

	Feature_1	Feature_2	Feature_3	Feature_4	Feature_build1	Feature_build2	Feature_build3	Feat
count	40	40	40	40	40	40	40	
mean	-0.375	2.025	-1.075	1.125	1.625	-0.025	1.95	
std	11.1509	10.7094	11.2121	10.6101	11.1509	10.7094	17.7662	
min	-20	-24	-28	-18	-18	-22	-30	
25%	-7.25	-4.75	-10.25	-8	-5.25	-7.25	-11	
50%	-1.5	3	0.5	0	0.5	-1	1.5	
75%	7.25	9.25	6.25	9.25	9.25	6.75	17	
max	22	24	26	22	24	26	36	

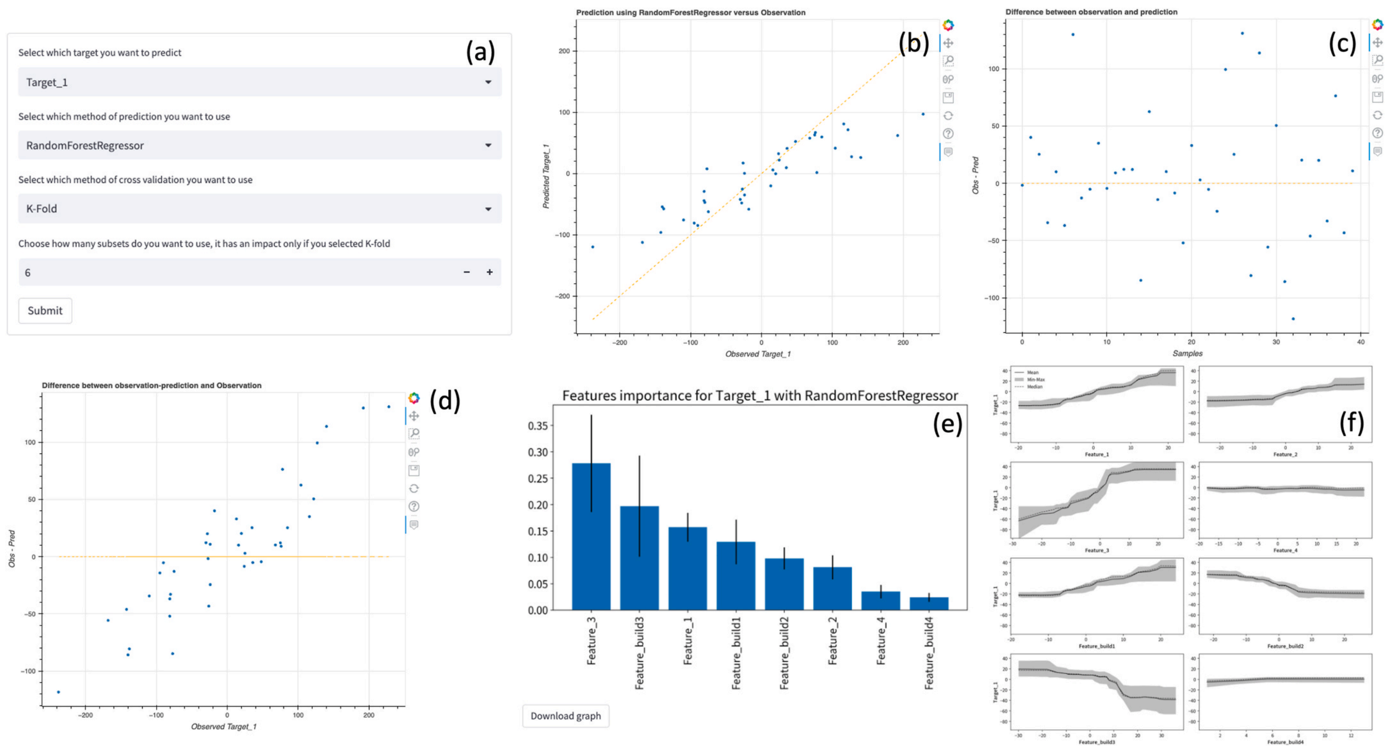
Fig. 2. Demonstration of "Main page" with (a) features and targets selection and (b) quick analysis of the dataset.

to note that a parameter cannot simultaneously be a feature and a target. After the parameter selection, the GUI displays a new data frame containing only the selected parameters, a table for the statistical description (count, mean, min, max, etc. Per column) of the dataset (Fig. 2 b), a histogram of the selected targets as a function of the samples, and a button to display a matrix of feature-to-target scatter plots using Seaborn pair plot function (Fig. 4 a). This latest is optional as it can take a long time to load.

3.2. Prediction page

The GUI automatically displays a Pearson linear correlation between the selected parameters on the Prediction page (Fig. 4 b). The Pearson correlation measures the strength of the linear relationship between two parameters, with values ranging between −1 and 1. This plot is generated using the Seaborn Heatmap function. Values above 0.5 and below −0.5 are displayed to prevent graph overload and enhance readability. A value of 1 signifies an expected linear relationship between two parameters, where an increase in one result in an equal rise in the other. Conversely, a value of −1 indicates an inverse linear relationship, where an increase in one parameter results in a decrease in the other.

The subsequent section is dedicated to prediction. Users can choose from various prediction models such as ElasticNet [8], RandomForestRegressor [18], or XGBRegressor [10], and cross-validation (CV) techniques like leave-one-out [29] or k-fold [30] (Fig. 3 a), all accessible from the Scikit-learn library except for XGBRegressor, which is from the XGBoost library. ElasticNet is a linear regression model that combines the lasso and ridge techniques to regularize regression models. It balances the two regularization types, ensuring a sparse and interpretable solution. RandomForestRegressor is an ensemble learning method that aggregates the results of multiple decision tree models. It creates various decision trees based on bootstrapped training data samples and selects the average prediction of all the trees, making it difficult to interpret. XGBRegressor implements gradient boosting, a powerful ML technique that trains many weak models to work together in a weighted sum to achieve a more accurate prediction. K-fold and leave-one-out are two different CV methods. The main difference is how the data is divided into subsets for training and validation. K-fold CV divides the data into k equally sized folds and repeats the process k times, while leave-one-out CV uses all but one data point as the training set and repeats the process n times, where n is the number of data points in the dataset. Therefore, users can choose an ML model and CV strategy (Fig. 3 a), and the GUI will display the prediction after the training is completed. Multiple graphs are displayed when training an ML model through a given CV strategy is over. The first graph (Fig. 3 b) shows a target's predicted against observed values on test data points only, i.e., data points not seen during the training for a fair evaluation of prediction performance. This lets the user visualize the prediction per sample and assess an ML model's performance. The data of the predicted values can be downloaded in a CSV file. The second graph (Fig. 3 c) represents the gap between observed and predicted values over the samples. Additionally, the third graph reports the same list of residuals but over the observed values (Fig. 3 d). This enables the user to see which samples or range of target properties are difficult to predict. Finally, the two last graphs air the feature importance; Fig. 3 e displays the importance score for all features from the trained ML model. A higher absolute score for a feature reveals its saliency in predicting the target property. This functionality can help reduce the number of features (e.g., ≫ 3) by performing a cut-off on the importance score. However, a feature importance graph and a subsequent cut-off on features are relevant only if the ML model's performance is satisfying, which is at the user's discretion. Fig. 3 f displays the partial dependence of the selected target over each feature, marginalizing the other features. This graph gave us information about the interaction between the target and the features, which can be associated with Fig. 3 e to determine which features are prevalent for the target evolution. Finally, the GUI also allows finding the best prediction model and CV strategy for a selected target. To do so, it calculates and finds the minimum value of the mean absolute error (MAE) between


Select which target you want to predict: Target_1

Select which method of prediction you want to use: RandomForestRegressor

Select which method of cross validation you want to use: K-Fold

Choose how many subsets do you want to use, it has an impact only if you selected K-fold: 6

Fig. 3. "Prediction page" presented in Fig. 1: (a) Prediction model selection, (b) visualization of the prediction, (c) gap between the prediction and observation over the dataset, (d) gap between the prediction and observation over the observation, (e) features importance representation and (f) partial dependence of the target for each feature.

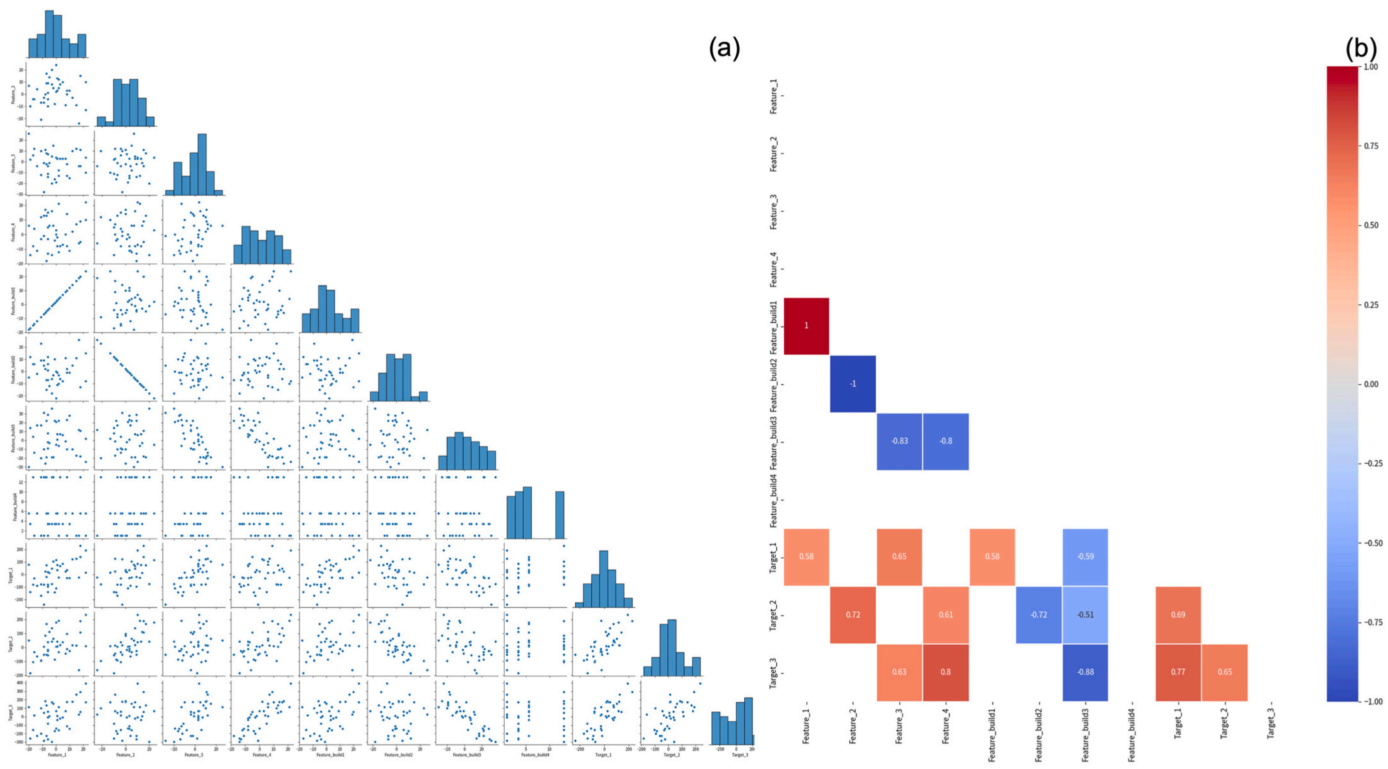
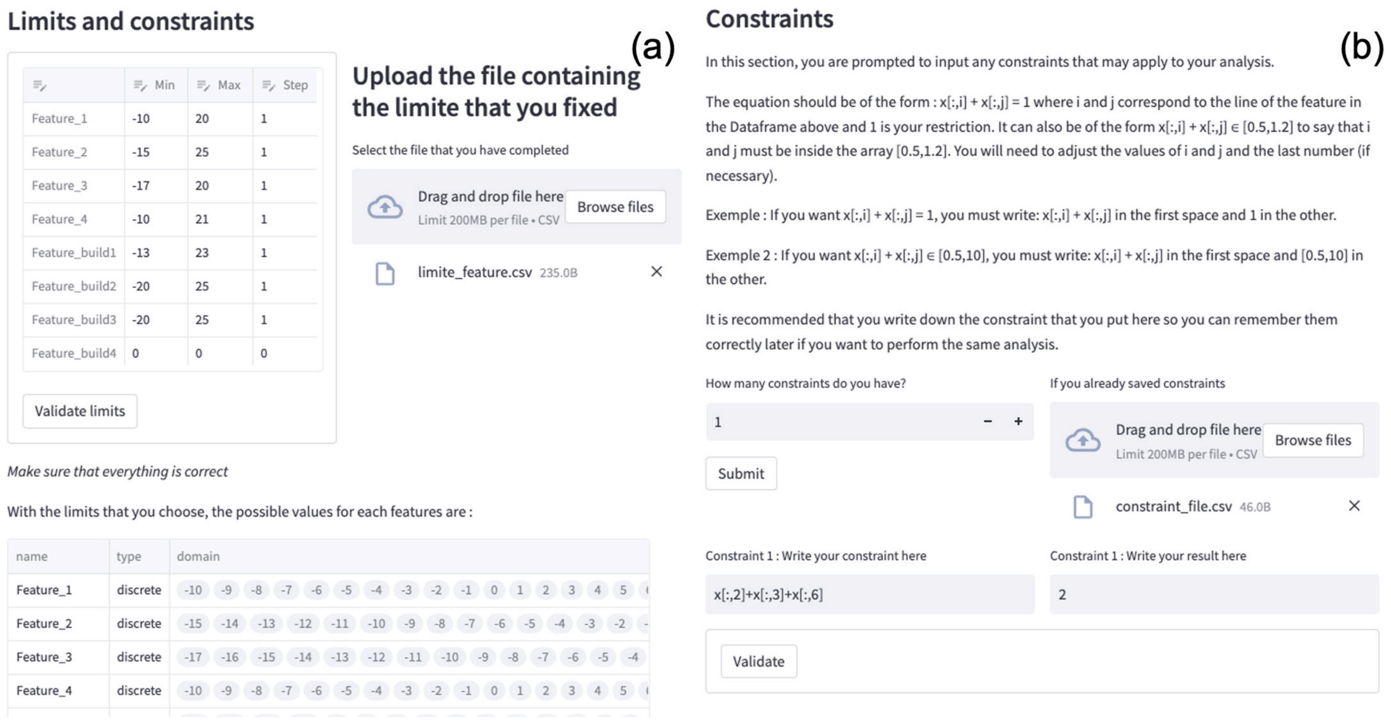


Fig. 4. (a) Seaborn pairplot correlation and (b) Pearson linear correlation using Seaborn heatmap.

actual target values and their predicted counterpart for each couple of ML models and the CV strategy at its disposal.

3.3. Bayesian page

The Bayesian page, the third one, involves the preparation and application of the Bayesian Optimization (BO) algorithm. Initially, the user must complete the limit selection part, represented either by a downloadable CSV file or an editable Dataframe (Fig. 5 a). Both options contain a Dataframe with five columns and as many rows as there are selected features. The first column contains the names of all features, the second and third must be filled with the minimum and maximum values, and the fourth is for the step between each value. The last column is used when the parameters can only take specific values, for example, one or



Limits and constraints

(a)

	Min	Max	Step
Feature_1	-10	20	1
Feature_2	-15	25	1
Feature_3	-17	20	1
Feature_4	-10	21	1
Feature_build1	-13	23	1
Feature_build2	-20	25	1
Feature_build3	-20	25	1
Feature_build4	0	0	0

Validate limits

Upload the file containing the limite that you fixed

Select the file that you have completed

Drag and drop file here Limit 200MB per file • CSV Browse files

limite_feature.csv 235.0B

Make sure that everything is correct

With the limits that you choose, the possible values for each features are :

name	type	domain
Feature_1	discrete	-10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5
Feature_2	discrete	-15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2
Feature_3	discrete	-17 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4
Feature_4	discrete	-10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5

Constraints

(b)

In this section, you are prompted to input any constraints that may apply to your analysis.

The equation should be of the form : x[:,i] + x[:,j] = 1 where i and j correspond to the line of the feature in the Dataframe above and 1 is your restriction. It can also be of the form x[:,i] + x[:,j] ∈ [0.5,1.2] to say that i and j must be inside the array [0.5,1.2]. You will need to adjust the values of i and j and the last number (if necessary).

Exemple : If you want x[:,i] + x[:,j] = 1, you must write: x[:,i] + x[:,j] in the first space and 1 in the other.

Exemple 2 : If you want x[:,i] + x[:,j] ∈ [0.5,10], you must write: x[:,i] + x[:,j] in the first space and [0.5,10] in the other.

It is recommended that you write down the constraint that you put here so you can remember them correctly later if you want to perform the same analysis.

How many constraints do you have?: 1

Submit

If you already saved constraints

Drag and drop file here Limit 200MB per file • CSV Browse files

constraint_file.csv 46.0B

Constraint 1 : Write your constraint here: x[:,2]+x[:,3]+x[:,6]

Constraint 1 : Write your result here: 2

Validate

Fig. 5. "Bayesian page" presented in Fig. 1: Determination of (a) the limits and (b) constraints initialization.

multiple values that are not evenly separated (e.g., 1,3.4,5.6,13). When the data frame is completed, the user must either upload the CSV file or validate the editable data frame. Then, the GUI will display the selection in another data frame, which will be the space S of possibilities for the BO.

The user can download the editable data frame after completion, allowing the user to keep the limits chosen for the next iteration. Another crucial part of this page is the constraint part (Fig. 5 b); in this section, the user can instruct the BO algorithm to follow some rules between different parameters when suggesting new points. For example, if the goal is to optimize a chemical composition, it must follow stoichiometry so the user can tell that the sum of the column must be equal to a specific number. In Table 1, Molecule A + Molecule B + Molecule C must equal 1. In the constraint section, it must be written as x[:,0] + x[:,1] + x[:,2] = 1, where 0, 1 and 2 correspond to the column's index. It is the only part of the GUI where the user must write mathematical equations.

After the constraint selection, the BO part begins. The user selects one up to three targets to optimize, selects for each target to minimise or maximize it and can change the ratio between them (Fig. 6 a). By default, it is 50-50 (for two targets) or 34-33-33 (for three targets). The user can choose how many suggestions points are needed (between 1 and 20). It depends on the user's limitation (resources, time required, machine availability or capacities …). When the selection is complete by clicking a button, it launches the GpyOpt Bayesian optimization. The GpyOpt parameters selected are Expected Improvement (EI) for the acquisition type, lbfgs (L-BFGS) for the acquisition type optimizer, Gaussian Process (GP) for the model type, the domain is the CSV file about the limits and constraints are the constraints that the user has entered. The result of the Bayesian optimization will then be displayed in a data frame and can be downloaded as a CSV file (Fig. 6 b).

The last part of this page is the Bayesian optimization using the prediction model. In this part, the user can use a prediction model to initialize the Bayesian optimization (Fig. 6 c). The algorithm replaces the initial data from the dataset with a hundred values generated with the prediction model. Using the prediction model to initialize the algorithm helps to unbias the dataset. We suggest using this only when the prediction model is performing well. Otherwise, it is not recommended.

The number of iterations required for optimization can vary depending on the research subject, the number of parameters, and the number of new samples per iteration. Typically, 3–4 iterations with 5–10 samples per iteration are performed before reaching the optimum [12–15], though this can vary based on the specific problem.

4. Illustrative examples

The following illustrations are an example of MADGUI using the data from a CSV file that is available in the GitHub repository of MADGUI. This dataset was built using data randomly generated using Scikit-learn's function *sklearn. datasets.make_regression,* and data was built in-house to display some of MADGUI's functionality. The code used to create this dataset is also available on the GitHub repository. Figs. 2–6 are screenshots of the GUI and display the different parts of the software shown in the flowchart in Fig. 1. The goal of those figures is to display the way to use MADGUI, its user-friendliness and its different functionalities.

Fig. 2 displays the selection part on the "Main Page", where the user must determine features and targets and display the selected input with a quick data analysis.

Fig. 3 demonstrates the prediction page by selecting the prediction model (ElasticNet, RandomForestRegressor or XGBRegressor) and the cross-validation method (Leave-One-Out or K-Fold). MADGUI automatically displays several graphs that give information about the feature's prediction or importance.

Fig. 4 combines two graphs in the "Main Page" for Fig. 4 a and the "Prediction Page" for Fig. 4 b. Both charts show the relationship between each entry of the dataset. Fig. 4 displays a scatter plot of the Seaborn pair plot correlation, while Fig. 4 b displays numerical values corresponding to the measures of the linear relationship between two parameters. We can see the linear correlation between several parameters in these graphs.

Fig. 5 is the selection of the space S for the Bayesian Optimization by determining the limits for each feature and the possible correlation between them via the constraint.

Fig. 6 displays the target selection for the Bayesian optimization; the user can choose up to 3 targets. After the BO, MADGUI will display a data frame that can be downloaded—the user also can use Bayesian Optimization with a prediction model. The difference is the data provided for initialization of the BO will be generated with the prediction model.

Table 1
Example of constraints.

Column 0	Column 1	Column 2	Column 3
Feature A	**Feature B**	**Feature C**	**Target value**
0.5	0.25	0.25	1.33
0.4	0.3	0.3	2.5
0.8	0.15	0.05	2.2

5. Impact

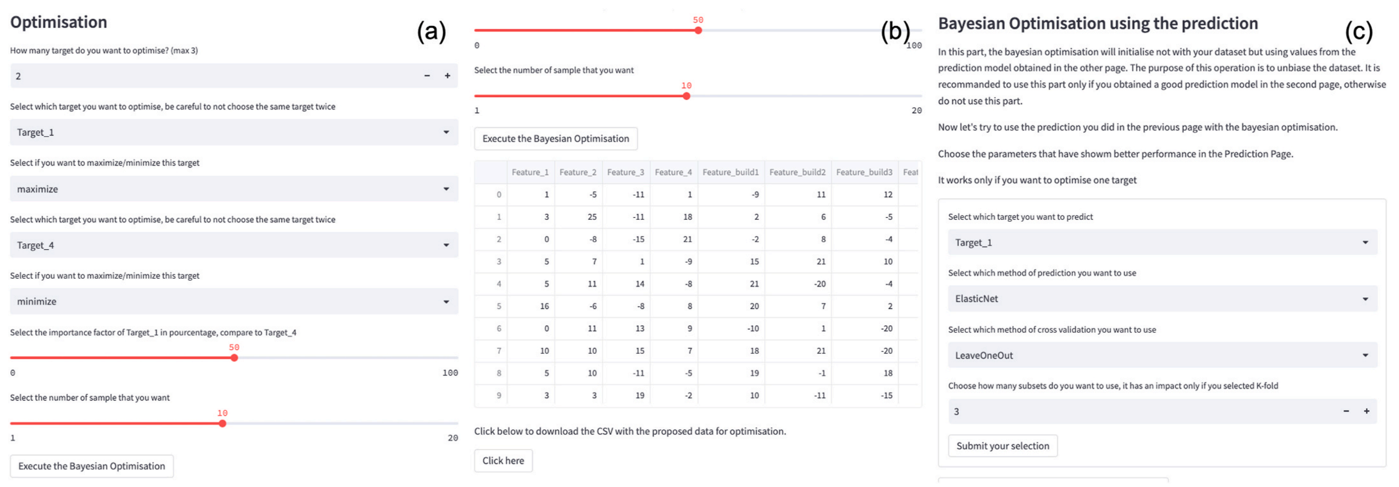
Overall, the MADGUI has the potential to significantly benefit researchers in various fields due to its user-friendly interface and the implementation of BO. Using BO allows researchers to explore beyond their existing knowledge, thus avoiding random points and saving resources. With just a few samples and iterations, optimal experimental parameters can be efficiently identified [12–15], making it a valuable tool for those working with companies or laboratories to improve processes or compositions. Our team has already applied this software in collaboration with industrial partners to improve process or composition. We envision the research community will widely adopt this powerful yet intuitive GUI to streamline their optimization process.

6. Conclusions

This work introduces MADGUI, a user-friendly and accessible GUI that leverages Bayesian Optimization (BO) for process, composition, and performance optimization across various applications. Through the integration of the Streamlit library, MADGUI bypasses the need for programming knowledge, allowing researchers, engineers, and data scientists to implement prediction models and BO seamlessly within their workflow. This innovation notably reduces the time, resources, and technical expertise typically required for experiments, making advanced optimization techniques available to a broader audience.

MADGUI democratizes data analysis and optimization access by offering a comprehensive, no-code platform where users can quickly navigate data selection, predictive modelling, and parameter optimization within a unified framework. BO is at the core of MADGUI, providing a focused approach to experimental design that minimizes iterative trials and enhances the efficiency of reaching optimal solutions. This efficiency makes MADGUI especially valuable in applications that traditionally require extensive trial-and-error testing, as it facilitates rapid convergence on improved experimental outcomes.

While MADGUI was initially developed with material science applications in mind, its flexibility enables use across various scientific fields, wherever experimental parameters can be optimized. Furthermore, MADGUI's web-based, real-time accessibility positions it as a versatile tool for collaborative work and broader deployment. Potential future developments include adding advanced ML algorithms and alternative optimization methods, such as conformal prediction, to further expand its scope and application.



Optimisation

How many target do you want to optimise? (max 3)

2

Select which target you want to optimise, be careful to not choose the same target twice

Target_1

Select if you want to maximize/minimize this target

maximize

Select which target you want to optimise, be careful to not choose the same target twice

Target_4

Select if you want to maximize/minimize this target

minimize

Select the importance factor of Target_1 in pourcentage, compare to Target_4

50 (0–100)

Select the number of sample that you want

10 (1–20)

Execute the Bayesian Optimisation

	Feature_1	Feature_2	Feature_3	Feature_4	Feature_build1	Feature_build2	Feature_build3	Feat
0	1	-5	-11	1	-9	11	12	
1	3	25	-11	18	2	6	-5	
2	0	-8	-15	21	-2	8	-4	
3	5	7	1	-9	15	21	10	
4	5	11	14	-8	21	-20	-4	
5	16	-6	-8	8	20	7	2	
6	0	11	13	9	-10	1	-20	
7	10	10	15	7	18	21	-20	
8	5	10	-11	-5	19	-1	18	
9	3	3	19	-2	10	-11	-15	

Click below to download the CSV with the proposed data for optimisation.

Click here

Bayesian Optimisation using the prediction

In this part, the bayesian optimisation will initialise not with your dataset but using values from the prediction model obtained in the other page. The purpose of this operation is to unbiase the dataset. It is recommanded to use this part only if you obtained a good prediction model in the second page, otherwise do not use this part.

Now let's try to use the prediction you did in the previous page with the bayesian optimisation.

Choose the parameters that have showm better performance in the Prediction Page.

It works only if you want to optimise one target

Select which target you want to predict

Target_1

Select which method of prediction you want to use

ElasticNet

Select which method of cross validation you want to use

LeaveOneOut

Choose how many subsets do you want to use, it has an impact only if you selected K-fold

3

Submit your selection

Fig. 6. "Bayesian page" presented in Fig. 1: (a) Targets determination, (b) optimization result and (c) Bayesian optimization with prediction model for initialization.

In summary, MADGUI represents a significant advancement in user-friendly experimental optimization, providing researchers with a tool that not only accelerates the experimental process but also reduces associated costs. Its impact on scientific discovery and process improvement is expected to be substantial, as it brings the advantages of active learning and Bayesian optimization to a broad user base.

CRediT authorship contribution statement

Christophe Bajan: Writing – original draft, Software, Conceptualization. **Guillaume Lambard:** Writing – review & editing, Validation, Supervision, Software, Project administration, Conceptualization.

Declaration of generative AI and AI-assisted technologies in the writing process

While preparing this work, the author(s) used Grammarly to improve readability and language. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

Funding

This research received no specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Independent testing by Dr. Juan M. Ugalde.

MADGUI was reviewed by Dr. Juan M. Ugalde an outside reviewer. For this review Dr. Juan M. Ugalde used the executable downloaded for our website https://madgui.odoo.com/our-services with the demonstration dataset from our GitHub https://github.com/Lambard-ML-Team/MADGUI/tree/main/MADGUI/Data/Demonstration.

After his review, Dr. Juan M. Ugalde declare.

"MADGUI - Multi-Application Design GUI", is a software package meant to provide the user with an intuitive platform to carry on Bayesian Optimization. Herein, the tool was used in a Macintosh environment, but other versions are also available and compatible with platforms such as Windows.

In regards of the software implementation, the user is guided through several steps: (i) the data upload, (ii) the prediction and (iii) the Bayesian optimization.

During the first step, the tool provides a clear explanation of the data format that must be implemented. More precisely, it details the variable names which must be used to incorporate the model inputs/features and outputs. Once selected, the user data is displayed to verify its adequacy, the user is also able to exclude features deemed not necessary for the next steps. At the end of this initiation, a quick analysis is performed; it displays the features distributions details (mean, std) as well as correlation plots.

The prediction is carried on in the second step, where the user is able to choose the target to predict as well as the prediction and cross validation methods. Executing the program results in a set of plots displaying the prediction results against the observation points as well as importance plots highlighting the relationship between the predicted target and each feature.

Finally, the Bayesian optimization is done in the third and final step. The user must input the features/targets limits and constraints, if applicable. Thereafter, the results are displayed in a table, these correspond to each parameter suggested values. Subsequently, said values would be measured/assessed by the user and the results would be added to the initial dataset in an iterative process to improve the accuracy of his estimations.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.chemolab.2025.105323.

Data availability

The data used in the article are not real value, just fabricated value for the use of explanation and to avoid linking the software to a specific field. The data and the code are accessible on GitHub.

References

[1] G. Wu, M. Sun, Machine learning for accelerating the discovery of high-performance donor/acceptor pairs in non-fullerene organic solar cells, npj Comput. Mater. 1 (6) (2020), https://doi.org/10.1038/s41524-020-00388-2.

[2] Caro Csányi Deringer, Machine learning interatomic potentials as emerging tools for materials science, Adv. Mater. 46 (31) (2019) 1902765, https://doi.org/10.1002/adma.201902765.

[3] Florea Rodrigues, Diamond Oliveira, Oliveira, Big data and machine learning for materials science, Discov Mater 1 (1) (2021), https://doi.org/10.1007/s43939-021-00012-0.

[4] Zhang Chen, Zhou, Machine learning: accelerating materials development for energy storage and conversion, InfoMat 3 (2) (2020) 553–576, https://doi.org/10.1002/inf2.12094.

[5] Choudhary Gupta, Campbell Tavazza, Choudhary Liao, et al., Cross-property deep transfer learning framework for enhanced predictive analytics on small materials data, Nat. Commun. 1 (12) (2021), https://doi.org/10.1038/s41467-021-26921-5.

[6] C. Bajan, G. Lambard, Madgui, Version 1.0.0, 2023. https://github.com/Lambard-ML-Team/MADGUI.

[7] Open-source Python library, "Streamlit, A Faster Way to Build and Share Data Apps", https://streamlit.io.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (Oct) (2011) 2825–2830.

[9] M. Waskom, O. Botvinnik, D. O'Kane, et al., Mwaskom/Seaborn: v0.8.1 (September 2017), Zenodo, 2017, https://doi.org/10.5281/zenodo.883859.

[10] T. Chen, C. Guestrin, XGBoost: a scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2016, pp. 785–794, https://doi.org/10.1145/2939672.2939785.

[11] J. González, Z. Dai, GPyOpt: a Bayesian optimization framework in Python. https://github.com/SheffieldML/GPyOpt, 2016.

[12] S. Pruksawan, G. Lambard, S. Samitsu, K. Sodeyama, M. Naito, Prediction and optimization of epoxy adhesive strength from a small dataset through active learning, Sci. Technol. Adv. Mater. 20 (1) (2019) 1010–1021, https://doi.org/10.1080/14686996.2019.1673670.

[13] A.S. Nugraha, G. Lambard, J. Na, M.S.A. Hossain, T. Asahi, W. Chaikittisilp, Y. Yamauchi, Mesoporous trimetallic PtPdAu alloy films toward enhanced electrocatalytic activity in methanol oxidation: unexpected chemical compositions discovered by Bayesian optimization, J. Mater. Chem. A 8 (27) (2020) 13532–13540, https://doi.org/10.1039/D0TA04096G.

[14] S. Matsuda, G. Lambard, K. Sodeyama, Data-driven automated robotic experiments accelerate discovery of multi-component electrolytes for rechargeable Li–O2 batteries, Cell Reports Physical Science 3 (4) (2022) 100832, https://doi.org/10.1016/j.xcrp.2022.100832.

[15] G. Lambard, T. Sasaki, K. Sodeyama, T. Ohkubo, K. Hono, Optimization of direct extrusion process for Nd-Fe-B magnets using active learning assisted by machine learning and Bayesian optimization, Scripta Mater. 209 (2022) 114341, https://doi.org/10.1016/j.scriptamat.2021.114341.

[16] E. Brochu, V.M. Cora, N. De Freitas, A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning, 2010. Preprint at arXiv. arXiv:1012.2599.

[17] D. Packwood, Theory of bayesian optimization, in: Bayesian Optimization for Materials Science, Springer, 2017, pp. 11–28, https://doi.org/10.1007/978-981-10-6781-5_2.

[18] L. Breiman, Mach. Learn. 1 (45) (2001) 5–32, https://doi.org/10.1023/a:1010933404324.

[19] D.R. Jones, M. Schonlau, W.J. Welch, Efficient global optimization of expensive black-box functions, J. Global Optim. 4 (13) (1998) 455–492, https://doi.org/10.1023/a:1008306431147.

[20] T. Head, M. Kumar, H. Nahrstaedt, G. Louppe, I. Shcherbatyi, Scikit-Optimize/Scikit-Optimize (v0.8.1), Zenodo, 2020, https://doi.org/10.5281/zenodo.4014775.

[21] J. Jiménez-Luna, J. Ginebra, pyGPGO: bayesian optimization for Python, J. Open Source Softw. 2 (2017) 431, https://doi.org/10.21105/joss.00431.

[22] F. Nogueira, Bayesian Optimization: open source constrained global optimization tool for Python. https://github.com/fmfn/BayesianOptimization, 2014.

[23] Y. Tian, M.K. Luković, T. Erps, M. Foshey, W. Matusik, AutoOED: automated optimal experiment design platform, arXiv preprint arXiv:2104.05959 (2021).

[24] A. Ishii, R. Kamijyo, A. Yamanaka, A. Yamamoto, BOXVIA: bayesian optimization executable and visualizable application, SoftwareX 18 (2022) 101019, https://doi.org/10.1016/j.softx.2022.101019.

[25] Streamlit community cloud, Streamlit cloud share. https://streamlit.io/cloud, 2023.

[26] W. McKinney, Pandas: a foundational Python library for data analysis and statistics, Python for high performance and scientific computing 14 (9) (2011) 1–9, https://doi.org/10.1109/MCSE.2007.53.

[27] C.R. Harris, K.J. Millman, S.J. van der Walt, et al., Array programming with NumPy, Nature 585 (2020) 357–362, https://doi.org/10.1038/s41586-020-2649-2.

[28] J.D. Hunter, Matplotlib: a 2D graphics environment, Comput. Sci. Eng. 9 (3) (2007) 90–95, https://doi.org/10.1109/MCSE.2007.55.

[29] B. Efron, The jackknife, the bootstrap and other resampling plans. https://doi.org/10.1137/1.9781611970319, 1982.

[30] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the gap statistic, J. Roy. Stat. Soc. B 2 (63) (2001) 411–423, https://doi.org/10.1111/1467-9868.00293.

[31] V. Duros, J. Grizou, W. Xuan, Z. Hosni, D.-L. Long, H.N. Miras, L. Cronin, Angew. Chem. Int. Ed. 56 (2017) 10815, https://doi.org/10.1002/anie.201705721.

[32] J. Snoek, H. Larochelle, R.P. Adams, Practical bayesian optimization of machine learning algorithms, Adv. Neural Inf. Process. Syst. 25 (2012).

[33] R. Lam, M. Poloczek, P. Frazier, K.E. Willcox, Advances in bayesian optimization with applications in aerospace engineering, in: 2018 AIAA Non-deterministic Approaches Conference, 2018, https://doi.org/10.2514/6.2018-1656.

[34] Aldair E. Gongora, et al., A Bayesian experimental autonomous researcher for mechanical design, Sci. Adv. 6 (2020) eaaz1708, https://doi.org/10.1126/sciadv.aaz1708.