Alcaráz, Alba; Capilla, Javier; Garcia-Hiernaux, Alfredo; Pérez-Amaral, Teodosio; Valarezo-Unda, Angel

Conference Paper

Environmental Cost Function for Time Series Models: The M4 Competition

ITS 33rd European Conference 2025: "Digital innovation and transformation in uncertain times", Edinburgh, UK, 29th June – 1st July 2025

Provided in Cooperation with:
International Telecommunications Society (ITS)

Suggested Citation: Alcaráz, Alba; Capilla, Javier; Garcia-Hiernaux, Alfredo; Pérez-Amaral, Teodosio; Valarezo-Unda, Angel (2025) : Environmental Cost Function for Time Series Models: The M4 Competition, ITS 33rd European Conference 2025: "Digital innovation and transformation in uncertain times", Edinburgh, UK, 29th June – 1st July 2025, International Telecommunications Society (ITS), Calgary

This Version is available at:
https://hdl.handle.net/10419/331246

Environmental Cost Function for Time Series Models: The M4 Competition*

Alba Alcaráz† Javier Capilla‡ Alfredo Garcia-Hiernaux§
Teodosio Pérez-Amaral¶ Angel Valarezo-Unda‖

Work in progress
June 18, 2025

Abstract

In this work, a cost function is estimated for eight models from the M4 competition. The main objective of the M competitions is to evaluate the accuracy of numerous forecasting models. This study introduces metrics to measure the environmental cost associated with running different time series models during the training and forecasting phases. This approach enables the construction of an environmental cost function that depends on other explanatory variables. Interpretable models help identify key drivers of environmental impact, while more complex machine learning models are used to predict emissions without rerunning the algorithms. The findings contribute to *Green AI* by promoting the evaluation of forecasting models not only by forecasting precision but also by sustainability.

Keywords: Green IA; M-competitions; forecasting; machine learning; sustainability.

*We would like to thank all the participants of the *International Conference on AI & Machine learning* (Kyoto, May 2025) for useful comments.

†Facultad de Matemáticas, Universidad Complutense de Madrid (UCM).
‡Banco de Santander and UCM.
§DANAE and ICAE, UCM. Corresponding author. Email: agarciah@ucm.es.
¶DANAE and ICAE, UCM.
‖DANAE and ICAE, UCM.

1 Introduction

In recent years, we have witnessed significant progress in the field of Artificial Intelligence (AI). This progress has been driven by the development of various tools, such as Large Language Models (LLMs), personalized recommendation systems, and machine learning algorithms, which have been applied in a wide variety of areas. This advancement has led to the use of AI in ways that were unimaginable a decade ago. However, all these developments, driven mainly by large technological companies, have been accompanied by a considerable environmental impact. Especially since 2020, some academic and popular science publications have turned their attention to this problem, establishing a new branch within the AI literature, known as *Green AI*; see, e.g., Anthony et al. (2020), Dhar (2020), Kaack et al. (2021), Schwartz et al. (2020), Strubell et al. (2019), Treviso et al. (2023), Yao et al. (2023). This new field proposes metrics to evaluate the environmental footprint of AI models, analyzes the effectiveness of the algorithms in terms of accuracy and resource sustainability, and recommends optimal practices to find a balance between technological progress and environmental cost. However, further progress is still needed in this domain.

This need is even more evident in the field of time series and their estimation algorithms. The time series literature has tended to focus on the development of models that generate increasingly accurate forecasts (of both, point estimates and confidence intervals), while the environmental impact associated with these methods has received little attention. An example can be found in forecasting competitions such as M-Competitions; see Makridakis et al. (2018); Makridakis (2020); Makridakis et al. (2020). Named after Professor Spyros Makridakis, these competitions are highly prestigious in the field of predictive time series analysis. Particularly, M4 focused on forecasting many time series from various fields such as finance, industry, and economics. M5 expanded the research and evaluation of forecasting methods by proposing a different dataset. Both competitions aimed to evaluate and compare the accuracy of different time series models when applied to real-world data from different domains. They have succeeded in identifying the most effective approaches in time series forecasting. However, they do not consider metrics that allow the environmental impact of the forecasting methods proposed by the participants to be assessed.

In this work, a methodology is developed to estimate the environmental cost in the training and forecasting phase per series for eight models from the M4 competition. In Section 2, all the characteristics of M4 and the estimation methods considered are explained in detail. We include a subsection dedicated to summarizing some classical time series models for a better understanding of the models created by the competitors. Section 3 presents the environmental impact metrics and the explanatory variables that define the cost functions that we estimate later. The different approaches used to estimate these cost functions are also explained. Section 4 shows the results obtained, which can be divided into two parts: (i) interpretability, dedicated to understanding which variables have the greatest influence on environmental cost, and (ii) prediction, whose goal is to choose a procedure that allows us to estimate the cost function accurately and robustly. Finally, Section 5 summarizes the main conclusions of the work and proposes future lines of research.

2 Background

In this section, we present the most relevant aspects of the M4 competition. This includes: (1) a review of the data, (2) the metrics used, and (3) a description of the models replicated in our work.

2.1 The M4 Competition

The M4 competition, the fourth in the M competition series, was held in 2018. The main objective of the M-competitions is to identify the time series models that provide the most accurate forecasts for different types of data. To achieve this goal, the organizers of M4 introduced several innovations compared to previous competitions. They proposed a broader set of time series, included machine learning methods, and introduced prediction intervals in the evaluation process. Due to this last innovation, the competition was divided into two categories: (i) point forecasts, and (ii) 95% confidence intervals forecasts.

2.2 Training Data of the M4 Competition

The dataset consists of 100,000 time series from various fields and periods, as shown in Table 1. The length of the series varies. For example, annual series have between 13 and 835 observations, quarterly series between 16 and 866, monthly series between 42 and 2794, weekly series between 80 and 2597, daily series between 93 and 9919, and hourly series between 700 and 960. The forecast horizon and the frequency of each time series depend on its period, as shown in Table 2. The frequencies were proposed by the competition organizers. They represent the seasonality considered for each period when evaluating the models. Therefore, a frequency of 1 (annual, weekly, and daily series) indicates that no defined seasonality is considered, whereas a frequency other than 1 indicates seasonality every 4 months in the case of quarterly series, 12 months for monthly series, and 24 hours for hourly series. Contestants were encouraged to explore other seasonality patterns and consider them if they find it appropriate.

Period	**Demographic**	**Financial**	**Industrial**	**Macro**	**Micro**	**Other**	**Total**
Annual	1088	6519	3716	3903	6538	1236	**23000**
Quarterly	1858	5305	4637	5315	6020	865	**24000**
Monthly	5728	10987	10017	10016	10975	277	**48000**
Weekly	24	164	6	41	112	12	**359**
Daily	10	1559	422	127	1476	633	**4227**
Hourly	0	0	0	0	0	414	**414**
Total	**8708**	**24534**	**18798**	**19402**	**25121**	**3437**	**100000**

Table 1: Summary of M4 competition time series by period and data type

The data were provided to the contestants divided into 6 `csv` files (one for each period), where each row corresponded to a time series and each column to the different data

Period	Horizon	Frequency
Annual	6	1
Quarterly	8	4
Monthly	18	12
Weekly	13	1
Daily	14	1
Hourly	48	24

Table 2: Forecast horizons and frequencies of M4 series

points in the series. An additional file was provided containing information about some characteristics of each series. All data can be found in the official M4 competition `GitHub` repository[1]. In March 2018, the `M4comp2018` package was created in `R`. It contains a list, denoted M4, corresponding to the competition dataset. Each series has an object structure, and using appropriate commands, one can access the various characteristics of the series, such as period, forecast horizon, length, etc. Some contestants decided to publish their code using this package to enhance reproducibility.

2.3 Model Evaluation Metrics in the M4 Competition

The metric selected to evaluate point forecasts was the Overall Weighted Average (OWA) of two accuracy measures: the symmetric Mean Absolute Percentage Error (sMAPE, Makridakis, 1993) and the Mean Absolute Scaled Error (MASE, Hyndman and Koehler, 2006). The formulæs for these metrics are the following:

$$\text{sMAPE} = \frac{1}{h}\sum_{t=n+1}^{n+h}\frac{|Y_t - \hat{Y}_t|}{\frac{1}{2}(|Y_t| + |\hat{Y}_t|)} \cdot 100\% = \frac{2}{h}\sum_{t=n+1}^{n+h}\frac{|Y_t - \hat{Y}_t|}{|Y_t| + |\hat{Y}_t|} \cdot 100\%, \tag{1}$$

where Y_t is the actual value of the series at time t, $\hat{Y}_t$ is the predicted value at t, h is the forecast horizon, and n is the number of observations in the series.

$$\text{MASE} = \frac{\frac{1}{h}\sum_{t=n+1}^{n+h}|Y_t - \hat{Y}_t|}{\frac{1}{n-m}\sum_{t=m+1}^{n}|Y_t - Y_{t-m}|} \tag{2}$$

where m represents the frequency of the series. The expression for MASE shown in 2.2 differs from the one presented in Hyndman and Koehler (2006), where m is set to 1.

The sMAPE, Equation (1), involves calculating the absolute error for each point in the forecast horizon of a series and dividing it by the average of the absolute actual value and the absolute predicted value. In this way, the error is not influenced by the scale of the data in the series (e.g., if the data has large values, it might result in large errors, and vice versa). Then, the average of the rescaled errors across the forecast horizon is computed

[1] https://github.com/Mcompetitions/M4-methods

and multiplied by 100. The sMAPE can be interpreted as a measure indicating the average deviation of the absolute error in relation to the actual magnitude of the values at each forecast point: the larger the terms in the summation, the worse the forecasts.

The MASE, see Equation (2), takes into account the frequency of the series. For a given series, each term represents the absolute error at point t divided by the average of the seasonal differences. MASE can be defined as the average of the absolute errors divided by the average of the seasonal differences over the forecast horizon. It can be interpreted as a measure indicating how effective a model is, in comparison to the effectiveness that would have been achieved using a seasonal "naïve" model (i.e., assuming the time series will follow the same seasonal pattern as in the past and setting the forecast at t as the observed value at $t-m$). If a term is less than one, it means the proposed model performs better than a seasonal naïve model for that point; if it equals one, they perform the same; and if it is greater than one, the basic seasonal model performs better for that point. As with the previous metric, the higher the values of the summation elements, the worse the predictions provided by the model.

To obtain the competitors scores, sMAPE and MASE are calculated for all series, and the corresponding averages are taken. The average sMAPE is divided by the average sMAPE of the Naïve 2 model (see Table 3 in the next section), programmed by the competition organizers, yielding a relative sMAPE, denoted as sMAPE*. A similar procedure is applied to MASE, yielding a relative MASE: MASE*. Then, OWA is calculated as:

$$OWA = \frac{1}{2}(\text{sMAPE}^* + \text{MASE}^*) \tag{3}$$

Obviously, the higher the OWA, the less accurate the forecasts produced by the model.

A different metric is used to assess the accuracy of the confidence intervals. However, as this paper just focus on point forecast, we will ignore that part.

2.4 Models from the M4 Competition

The M4 competition had 248 participants. However, only 49 valid submissions were received, and not all teams made their model codes public. Among the published codes, some are not reproducible. Below, we explain the models considered in our study. Some models correspond to combinations of different methods, mainly statistical, but also including machine learning. Many of these statistical methods are classic time series models well-known in the literature. Therefore, first, we will briefly review some classical methods that were frequently used in the competition.

Another feature of the M4 competition was that competitors had access to a set of benchmark models. Accordingly, in the second part, we list these benchmarks proposed by the organizers. Finally, we detail the models we selected to measure environmental impact.

2.4.1 Classical Time Series Models

1. **Naïve**: This consists of using the last observed value of the series as the predicted future value. That is, $\hat{Y}_t = Y_{t-n}$ where n is the number of time steps since the last observed value of the series.

2. **Seasonal Naïve**: The predictions are equal to the last observed value of the series in the same period, $\hat{Y}_t = Y_{t-m \cdot n}$ where m is the frequency of the series and n is the number of time steps since the last observed value.

3. **Random Walk**: This model assumes that each point in a time series is the result of a random step from the previous point:

$$Y_{t+n} = Y_t + \phi(n)$$

where ϕ is a random variable that describes a probability law for taking the next step, and n corresponds to the time interval between steps. If the average of the random steps is zero, one option could be to set the prediction as the last observed value of the series (Naïve model). Sometimes, the average of the random steps may be positive or negative, which is known in the literature as a Random Walk with Drift. In that case, instead of setting the last value of the series as the prediction, let μ be the average of the random steps over n time steps, and the prediction is:

$$\hat{Y}_{t+n} = Y_t + n \cdot \mu$$

4. **ARIMA**: ARIMA models are a combination of three methods: moving average processes (MA), autoregressive models (AR), and the differencing process (transforming a non-stationary series into a stationary one by applying first-order differences). The equation of an ARIMA(p,d,q) model is expressed as:

$$\Delta^d Y_t = \sum_{i=1}^{p} \phi_i \Delta^d Y_{t-i} + \varepsilon_t - \sum_{j=1}^{q} \theta_j \varepsilon_{t-j}$$

where d represents the number of differences that transform the original series into a stationary one; ϕ_i, $i = 1, \ldots, p$, are the coefficients of the AR part; θ_j, $j = 1, \ldots, q$, are the coefficients of the MA part; and ε_t is the error term.

5. **Exponential Smoothing Methods**: Exponential Smoothing (ES) methods consist of forecasting future values as a weighted average of past values. The weights are higher for more recent observations and decrease exponentially for older observations. There are different types of exponential smoothing methods, among which we highlight: SES (Simple Exponential Smoothing), which consists of performing exponential smoothing assuming no trend and applying seasonal adjustments; the Holt-Winters method, which allows for a linear trend and seasonal adjustments; and DES (Damped Exponential Smoothing), which allows for a damped trend (that is, a directional trend whose magnitude decreases over time) and seasonal adjustments. Readers interested in a deeper understanding of how these methods work may refer to Hyndman et al. (2008).

6. **ETS**: ETS model (see Hyndman et al., 2008) encompasses different exponential smoothing algorithms. The letters E, T, and S represent Error, Trend, and Seasonality, respectively. Error can be additive or multiplicative. Trend can be *none*, additive, or additive damped. Seasonality can be *none*, additive, or multiplicative.

2.4.2 Benchmark Models

Table 3 shows the M4 benchmark models. The code for these models can be found in the official `GitHub` repository of the competition.

Method	Description
	Statistical Methods
Naïve 1	Naïve method explained in the previous section.
Naïve S	Seasonal Naïve method explained in the previous section.
Naïve 2	Similar to Naïve 1 but the data are seasonally adjusted, if necessary, by applying a classical multiplicative decomposition. A 90% autocorrelation test is performed to determine whether the data are seasonal.
SES	Explained in the previous section.
Holt	Holt-Winters method explained in the previous section.
Damped	DES method explained in the previous section.
Theta	See Assimakopoulos and Nikolopoulos (2000).
Comb	The simple arithmetic average of SES, Holt, and Damped.
ETS	Explained in the previous section.
ARIMA	Explained in the previous section.
	ML Models
MLP	A perceptron with a very simple architecture and parameterization. Some preliminary trend and seasonal adjustment is applied.
RNN	A recurrent neural network with a very basic architecture and parameterization. Some preliminary trend and seasonal adjustment is applied.

Table 3: Benchmark models of the M4 Competition in the forecasting category

2.4.3 Selected Models from the M4 Competition

A total of eight algorithms are selected (not the top eight ranked) to evaluate the training and prediction cost per series. All of them are implemented in `R`. Below, we briefly explain each of these methods. The code for these models can be found in the official `GitHub` repository of the M4 competition.

Model 69: GROE

Model 69 achieved fifth place in the competition, both in the forecasting category and in the confidence interval construction category. It consists of combining the forecasts of four prediction methods with weights obtained through cross-validation.

First, forecasts (point estimates and confidence intervals) are generated for a time series by applying four models: DOTM, OTM, ETS, and ARIMA. The ETS and ARIMA models are executed using the `forecast` package. The DOTM (optimized dynamic Theta method) and OTM (Optimized Theta Method) models are used from the `forecTheta` package. For more details on the last two methods, refer to Fiorucci et al. (2016). Secondly, cross-validation is performed to assess the quality of each algorithm. Let n be the

number of time steps in the training set for a time series. Let h be the number of points in the forecast horizon. Let $N = n + h$. The authors define:

$$t_1 = \begin{cases} N - h, & \text{if } N - h \geq 5 \\ 5, & \text{otherwise.} \end{cases}$$

Let $m = \left\lfloor \frac{h}{6} \right\rfloor$. Define $t_i = t_{i-1} + m$ for $i = 2, 3, 4, 5, 6$. The loss function used for each individual method is:

$$GROE = \sum_{i=1}^{6} \left(\sum_{j=1}^{\min(h, N - t_i)} g(y_{t_i+j}, \hat{y}_{t_i+j}) \right) \tag{4}$$

where g corresponds to the OWA defined in Equation (3). A score is calculated for each method as the inverse of the GROE function, and the weight of method k is computed by dividing its score by the sum of the scores of all methods, $k = 1, \ldots, 4$. Once the weights of the different methods for a series are calculated, they are used to obtain the forecast horizons.

Model 36: SCUM

SCUM was the sixth-best contribution, both in point forecasting and in interval forecasting. It combines the results of four standard methods: ETS, CES (Complex Exponential Smoothing, see Svetunkov et al., 2022), ARIMA, and DOTM. To apply the first three methods, functions from the *forecast* package are used, and for forecasting with the DOTM method, the *forecTheta* package is used. Given a time series, its forecast horizon and the lower and upper bounds of the 95% interval are calculated using each of the models. The final forecast is the median of the four forecasts produced, and the upper and lower bounds are the median of the upper and lower bounds obtained from the standard methods.

Model 78: THIEF

The THIEF model, the seventh-best contribution in the forecasting category, consists of combining (by summing) different points of a seasonal time series into a single value, resulting in an auxiliary time series of shorter length. For example, a monthly series with seasonality $m = 12$ can be aggregated in groups of $k = 2, 3, 4, 6, 12$. The aggregated frequency will be equal to $\frac{m}{k}$. By aggregating a series using different values of k, multiple forecasts can be created and then combined to produce the forecast of the original series; see Athanasopoulos et al. (2017).

Model 260: Theta Box-Cox

This was the eighth-best model in the competition in the point forecasting category. It did not participate in the interval forecasting category. The steps can be summarized in the following algorithm. Given a series Y:

1. The series is decomposed to estimate the seasonal component and deseasonalized if necessary.

2. A Box-Cox transformation is applied.

3. The Theta method is applied using the `forecast` package in `R`.

4. The Box-Cox transformation is reversed, and the forecasts are reseasonalized (adjusted by multiplying by the seasonal indices found in the decomposition, step 1).

Model 39: Predilab

Model 39 ranked tenth in the point forecasting category and did not compete in the interval forecasting category. It uses a combination of different models to generate the results. The models used are: Seasonal Naïve, Naïve 2, SES, Holt-Winters, DES, 4Theta, and 4Theta-ARMA. The 4Theta model corresponds to a generalization of the classical Theta model (see Spiliotis et al., 2020). The 4Theta-ARMA model is a variant of the 4Theta model that consists of adding an ARMA model to analyze and adjust the residuals of the forecasts generated by the 4Theta model. The auxiliary functions used to generate the forecasts for each of these methods are available in the official `GitHub` repository of the M4 competition. For the methods that match the benchmark models, the functions published by M4 are used.

Given a series Y_t, forecasts are generated using a function created by the competitor called `predilab`. The inputs of the function are the observations of the series Y_t, its forecast horizon, its length, and its frequency. The first thing the function does, given the inputs, is check whether the data in the series are integers. If they are not, it sets the number of decimals to 16. It then normalizes the data. After this, if the frequency of the series is 1 (annual, weekly, or daily), it uses the 4Theta-ARMA model; if it is 4 (quarterly), it also uses the 4Theta-ARMA model; if it is 24 (hourly), it compares the variance of the hourly differences with the variance of the weekly differences, sets the frequency to 24 or 168 depending on the result, and applies the Seasonal Naïve model; and for all other frequencies, it applies the internal function `predilabM4naiveSelect`. This function has the same inputs as `predilab` and works as follows. If the series has fewer than 24 observations, it applies the 4Theta model. Otherwise, for each series, it applies the Seasonal Naïve, Naïve 2, SES, Holt-Winters, DES, 4Theta, and 4Theta-ARMA models. It calculates the in-sample sMAPE of each model, ranks the models from lowest to highest sMAPE, and returns as the forecast the average of the three models with the lowest sMAPE.

Model 005: 4Theta

Model 005 applies a single forecasting algorithm to generate the prediction horizons for each series. It ranked eleventh in the forecasting category, which was the only category in which it participated. The method used is 4Theta. It is a statistical method that, as mentioned in the previous section, generalizes the classical Theta method. The modifications are mainly found in the decomposition structure of the model. Unlike the classical Theta model, it considers both linear and nonlinear trends, allows for adjusting the intensity of the trend, and includes a multiplicative expression. For more information, see Assimakopoulos and Nikolopoulos (2000) and Spiliotis et al. (2020).

Model 251: DOTM

Model 251 participated in both categories of the competition, ranking 13th in point forecasting and 10th in 95% interval construction. The model consists of applying the DOTM method (see Fiorucci et al., 2016) from the `forecTheta` package. The algorithm can be summarized in the following steps. Given a series Y_t:

1. Let n be the number of observations in Y_t:
   (a) If $n \geq 5000$, the training data for series Y_t consist of the observations from point $n - 4999$ to the last point of the series.
   (b) If $n < 5000$, all observations of the series Y_t are considered as training data.
2. The DOTM method is applied using the `forecTheta` package in `R`.

Model 252: ATA-Damped

Competitor 252 ranked 41st in the point forecasting category and 17th in the confidence interval construction category. The model was implemented using the `ATAforecasting` package.

The ATA-Damped model is similar to the Damped Exponential Smoothing model (see Yapar et al., 2018, 2019). The model is defined in both additive and multiplicative forms. The notation typically used is $ATA_{add}(p, q, \phi)$ for the additive model and $ATA_{mult}(p, q, \phi)$ for the multiplicative model, where p is the smoothing parameter for the level, q is the smoothing parameter for the trend, and ϕ is the damping coefficient for the trend. The algorithm used by the competitor can be summarized as follows. Given a series Y_t:

1. The frequency of series Y_t is determined.
   (a) If the frequency is annual, the model $ATA_{add}(p, 1, \phi)$ is applied with $\phi \in \{0.05, 0.10, 0.15, \ldots, 1\}$ to obtain the forecasts and the 95% interval bounds.
   (b) If the frequency is not annual, the models $ATA_{add}(p, 1, \phi)$, $ATA_{mult}(p, 1, \phi)$, $ATA_{add}(p, q, \phi)$ and $ATA_{mult}(p, q, \phi)$ are applied with $\phi \in \{0.85, 0.90, 0.95, 1\}$. The forecasts and interval bounds are returned as the arithmetic mean of the estimates and the bounds from each model.

3 Methodology

This section presents the methodology followed to address the main objectives of the study, which can be summarized in the following two points. First, to estimate a cost function that allows us to study the influence of different variables on the environmental cost generated when training various time series models and providing point forecasts and confidence intervals. For its estimation, interpretable models will be applied that allow us to understand the relationship between the explanatory variables and the target variable. Second, to estimate a cost function whose main purpose is to accurately predict CO_2 emissions, given the value of different explanatory variables. Machine learning algorithms

such as Random Forest (RF) and XGBoost (XG) will be used, and several metrics will be considered to evaluate the quality of the predictions.

The developed methodology can be summarized in the following steps:

1. Measure the environmental cost per series of the algorithms explained in the previous section during the training and prediction phases, using the M4 competition series set. Since the number of series in the M4 competition is large and some models considered are complex, a Simple Random Sample (SRS) will be taken according to the weights of each frequency in the M4 competition. The sample size depends on the algorithm's runtime, estimated from initial experiments. The environmental cost will be measured in CO_2 emissions (kg) by means of the `Eco2AI` library in `Python`.

2. Define the model and the variables that will be used to explain the environmental cost.

3. Build interpretable models that allow us to study how the explanatory variables influence the environmental cost.

4. Use different machine learning models to better predict the environmental cost based on the explanatory variables.

3.1 Metrics for Measuring Environmental Impact

In recent years, some tools have been developed to measure the environmental impact of AI models (Heguerte et al., 2023, see). Among them, it is worth to mention `Green-algorithms`, `CodeCarbon`, `Experiment-Impact-CarbonTracker`, `Eco2AI`, `MLCO2`, `CarbonTracker` and `Cumulator`.

In our study we use the `Eco2AI` library in `Python` (Budennyy et al., 2022). The main reasons for using `Eco2AI` are that it has been widely used in the literature, is easy to implement, and allows selecting the specific process of the algorithm to be measured. The functioning of `Eco2AI` is straightforward. First, an object is imported, which we denote as *Tracker*, and initialized at the line of code chosen by the user. The *Tracker* will store information on environmental cost variables from the lines of code that follow, in `csv` format. In particular, the variables for which information is recorded are: Project name, Experiment description, Start time, Duration (s), Energy consumption (kWh), CO_2 emissions (kg), Computer name, GPU name, Operating system, and Country from which the experiment is launched. Finally, the *Tracker* is closed at a line of code chosen by the user. The information on the above variables is saved in a `csv` file, which is by default called *emission.csv*. When another *Tracker* is opened, the previous process is repeated, and the results are stored in the next line of *emission.csv*, and so on.[2]

[2]Since `Eco2AI` is a `Python` library and the algorithms considered are programmed in R, Appendix 1 shows the procedure followed to load this library in `R`. For details on the dependencies of `Eco2AI` (dependent libraries that must be installed, required versions of the dependent libraries, etc.), see https://github.com/sb-ai-lab/Eco2AI.

3.2 Estimation of the Cost Function

Being i the i-th series of the M4 competition, we define:

$$c_i = f\big(a_i, e_i, ne_i, pt_i, pl_i, npl_i, n_i, fq_i, p_i, np_i, t_i, cpu_i\big) \tag{5}$$

where c_i is the cost in terms of CO_2 emissions (kg) corresponding to the training of a model and prediction of a given horizon for series i. We make the cost of a series dependent on ex-ante variables that can be classified into three groups composed of: (i) variables that depend on the method used by the competitor to generate their forecasts; (ii) variables that depend on the characteristics of the series; and (iii) a single variable, which captures the characteristics of the computer used to measure the emissions.

The following variables depend on the forecasting method used:

- a_i Forecasting algorithm for series i.
- e_i Whether the algorithm for series i corresponds to an ensemble of models.
- ne_i Number of standard methods used in the algorithm for series i.
- pt_i Type of forecast returned by the algorithm for series i (point forecast, confidence interval, or both).
- pl_i Whether the algorithm for series i is programmed in parallel or sequentially.
- npl_i Number of cores used to run model a_i.

The following variables depend on the characteristics of the series:

- n_i Length of series i.
- fq_i Frequency (seasonality) of series i (as considered by the competition organizers).
- p_i Period of series i.
- np_i Number of forecasts for series i.
- t_i Type of data in series i.

The following variable reflects the characteristics of the computer used to measure the per-series costs of the different methods:

- cpu_i Processor used to measure the cost of series i.

As previously mentioned, f will be estimated using different approaches. To try to understand the relationship between the explanatory variables and the target variable, a linear regression and a regression tree model will be used. The linear regression model

allows us to obtain a linear version of Equation (5). The coefficients of the regressors will show the linear relationship between the independent variables and the variable to be predicted. The regression tree does not allow for an explicit representation of that equation, as it is based on a hierarchical decision structure. At each node, conditions are imposed on the independent variables, and predictions are made at the final leaves of the tree based on these conditions. We will build a shallow tree in order to identify the most influential explanatory variables on the endogenous variable.

Likewise, the XGBoost and Random Forest (RF) models do not allow for an explicit representation of Equation (5). The XGBoost model, in very general terms, consists of building trees sequentially, so that each subsequent tree corrects the errors of the previous one. To do this, the new tree is fitted to the residuals of the previous tree using the gradient of a loss function, which is usually based on a known error metric. The model is updated by incorporating the newly fitted tree according to a learning rate. In turn, RF, in very broad terms, consists of generating bootstrap samples from the original dataset and constructing a regression tree for each sample. The final prediction is the arithmetic mean of the predictions generated by each tree. To prevent the trees from being highly correlated, each one is built using a random subset of features and a partial subset of data.

4 Results

This section presents the main results of the study according to the interpretability and prediction objectives outlined. First, some preprocessing tasks performed on the initial dataset will be described.

4.1 Database Construction and Preprocessing

As explained in Section 3, the first step consists of building a database that collects information on the per-series cost of the different models presented in Section 2. The selected models were executed using two different computers: *Apple M2/1 device(s), TDP:100.1*, referred to as Apple; and *Intel(R) Core(TM) i7-9700T CPU @ 2.00GHz/1 device(s), TDP:35.0*, referred to as Intel. The number of series executed per model was selected based on initial experiments used to estimate algorithm runtimes. Once the number of series per model was defined, a SRS was drawn according to the weights of each frequency in the competition.

Some algorithms were not fully reproducible. Model 39 (Predilab) could not run hourly series, and model 78 (THIEF) was only able to run annual, quarterly, and monthly series. Specifically, Table 4 in Appendix 3 provides information on how many series were run per model and frequency, as well as the computer used to reproduce each model. During the cost measurement process, several noteworthy issues were observed. First, in models that parallelize, results were not correctly saved. Appendix 2 explains in detail the main reasons for this issue. On the other hand, whether a model is parallelized does not affect the per-series cost under the assumption of core independence, because the

training and forecasting process for a given series is executed on the same core. While the total runtime of the model is shorter, the runtime per series remains the same. The difference is that, during the time in which the model is being trained, and the forecast horizon is being generated for series i, the same is also being done for series $j, k, w, \ldots$. Therefore, the variables related to parallelization (p_i and np_i) are removed from Equation 5. Another observation is that `Eco2AI` includes in its cost metrics the time, kWh, and CO_2 emissions generated when opening the *emission.csv* file after finishing an experiment. This creates a new challenge, as the time to read (and write) this file increases with its size, introducing a non-constant bias. If the experiment has a long duration, this bias may be negligible. However, since this study measures cost per series, the experiments performed are numerous and generally of short duration. Thus, it is necessary to estimate the bias introduced for each series. Appendix 2 explains the process implemented to correct the cost metrics.

Using the information collected by `Eco2AI` after these corrections, and according to the ex-ante variables defined in Section 3.2, a dataset with a total of 202,717 observations is generated. Although the variable measuring experiment duration will not be used—since it is not ex-ante—it is interesting to analyze how it relates to CO_2 emissions.[3]

One observation where CO_2 emissions were negative was removed, as it was considered a measurement error, along with four other observations identified as potential outliers. While more refined techniques, such as a residual analysis, could be used to detect outliers, these four cases represent a negligible proportion of the sample size, and were therefore removed without applying such methods. At this stage, the dataset consists of 202,712 observations and 12 columns: CO_2 emissions (target variable) and the variables defined in Equation (5) (excluding those related to parallelization, which, as previously justified, were discarded). Appendix 3 (Tables 5 and 6) provides descriptive statistics of the target variable and average cost per model. Notably, models based on a single forecasting algorithm (DOTM, Theta Box-Cox, and 4Theta) seem, at first glance, to consume less than those composed of multiple standard methods, with SCUM being the only exception.

Figures 7-14 in Appendix 3 show the environmental cost as a function of the number of series observations, broken down by frequency, for each model. The variability in cost for series with similar lengths depends on both the frequency and the model. Although some frequencies, such as monthly, daily, or hourly, tend to show higher variability for series of similar lengths, this is also influenced by the model used. For series of similar lengths, greater variability tends to appear in seemingly more complex models compared to simpler ones.

Figures 15-22 (Appendix 3) present the cost per frequency for each model. Again, while some frequencies (monthly, daily, or hourly) tend to exhibit greater variability in emissions, the outcome is again tied to the forecasting model. Figure 23 shows the emissions for each of the models considered. The more complex models generally display greater variability in the variable of interest.

Figure 24 in Appendix 3 shows the matrix of linear correlations for our dataset, where the qualitative categorical variables (Forecasting algorithm, Period, Prediction type, En-

[3]Figure 6 in Appendix 3 shows the relationship between CO_2 emissions and experiment duration. This relationship appears to be nonlinear, with an increasing trend.

semble -whether the model consists of one or more standard forecasting methods- Series data type, and Processor used) were converted into binary dummy variables. For each, as many dummies were created as there were possible values, and one was removed to avoid issues of perfect multicollinearity. In particular, ATA-Damped, Daily, Point forecasting, No ensemble (a single standard method), Other (for data type), and Intel were removed. Additionally, we build the polychoric correlation matrix for the binary dummy variables, which can be found in Figure 25 in Appendix 3. We highlight the strong correlation between the Forecast horizon and Frequency variables. This is due to the fact that the frequency (seasonality) considered for each period is fixed by the competition organizers. Likewise, the forecast horizon established for each period is also fixed. Thus, annual series always have frequency 1 (no seasonality) and a forecast horizon of six steps; quarterly series always have seasonality every 4 months and a forecast horizon of 8 steps, etc. We remove then both, the horizon and frequency variables and keep only the dummies corresponding to the period.

On the other hand, the dummy variable associated with whether the model uses one or several standard methods is highly correlated with the variable that captures the number of forecasting methods used per model. This was expected since, if there is no ensemble, the number of methods will be one, and if there is an ensemble, this number will be different from one. Consequently, we decided to remove the ensemble dummy variable and retain only the variable indicating the number of methods per model.

4.2 Interpretability of the Explanatory Variables

The linear regression model is then defined as:

$$\begin{aligned}\log c_i = \beta_0 &+ \beta_1 \cdot n_i + \beta_2 \cdot ne_i + \beta_3 \cdot A_i + \beta_4 \cdot T_i + \beta_5 \cdot M_i + \beta_6 \cdot S_i + \beta_7 \cdot H_i + \beta_8 \cdot 4THETA_i \\ &+ \beta_9 \cdot Predilab_i + \beta_{10} \cdot DOTM_i + \beta_{11} \cdot ThetaBoxCox_i + \beta_{12} \cdot SCUM_i + \beta_{13} \cdot GROE_i \\ &+ \beta_{14} \cdot THIEF_i + \beta_{15} \cdot pt_i + \beta_{16} \cdot cpu_i + \beta_{17} \cdot Micro_i + \beta_{18} \cdot Macro_i + \beta_{19} \cdot Finance_i \\ &+ \beta_{20} \cdot Industria_i + \beta_{21} \cdot Demografico_i + \varepsilon_i\end{aligned} \tag{6}$$

for $i = 1, \ldots, 202712$, where:

- n_i, ne_i, cpu_i, and pt_i were defined in Equation (5). According to the preprocessing step, cpu_i equals 0 if the processor used to execute series i is Intel, and 1 if it is Apple; and pt_i equals 0 if the model running series i competes in the point forecasting category and 1 if it competes in both.
- Regressors 3-7 take the value 0 or 1 depending on the period. A_i equals 1 if series i is annual and 0 otherwise; T_i equals 1 if it is quarterly and 0 otherwise; M_i equals 1 if it is monthly; S_i equals 1 if weekly; and H_i equals 1 if hourly. If the series is daily (reference category), all these regressors take the value 0.
- Regressors 8-14 take the value 0 or 1 depending on the model. If series i was run using the model named in the regressor, it takes the value 1; otherwise, 0. If the model is ATA-Damped (reference model), all previous variables take the value 0.

- Regressors 17-21 take the value 0 or 1 depending on the data type of series i. If the series corresponds to the data type named in the regressor, it takes the value 1; otherwise, 0. If the data type is Other (reference), all these variables take the value 0.

The model defined in Equation (6) was trained on the full sample. The R^2 is equal to 0.834, and so the model's regressors explain 83.4% of the variability in the dependent variable. A priori, our hypothesis was that the variables related to data type and prediction type would not be significant. For the data type, this seems justifiable. For the prediction type variable, some clarification is warranted. Competitors that participate in both categories usually use the same algorithm to generate both point forecasts and intervals; and when a series is executed, the resulting object typically includes, by default, the forecast horizons, error metrics, confidence intervals at specific levels, etc. Moreover, most of the algorithms that competed in both categories were run on the Intel processor, meaning there's a strong negative correlation with the Processor dummy variable. The p-values of the regressors associated with these variables do not reflect the overall significance of the variable, but rather the significance relative to the reference category. We discard the regressors reflecting the data type and prediction type and define, for $i = 1, \ldots, 202712$:

$$\begin{aligned}\log c_i = \beta_0 + \beta_1 \cdot n_i + \beta_2 \cdot ne_i + \beta_3 \cdot A_i + \beta_4 \cdot T_i + \beta_5 \cdot M_i + \beta_6 \cdot S_i + \beta_7 \cdot H_i + \beta_8 \cdot 4THETA_i \\ + \beta_9 \cdot Predilab_i + \beta_{10} \cdot DOTM_i + \beta_{11} \cdot ThetaBoxCox_i + \beta_{12} \cdot SCUM_i + \beta_{13} \cdot GROE_i \\ + \beta_{14} \cdot THIEF_i + \beta_{15} \cdot cpu_i + \varepsilon_i\end{aligned} \tag{7}$$

The R^2 of the previous model reach 83.3%, very similar to model represented in Equation (6). The number of observations and the number of standard methods contribute positively to cost. The cost of running the series on the Apple processor is, on average, 48.967% lower ($\exp(-0.6727) - 1 \approx -0.48967$) than running them on an Intel processor, holding the rest of the variables constant. The average cost for annual, quarterly, and monthly series is 67.453%, 38.676%, and 1.823% lower than for daily series, respectively (in all cases, holding other variables constant). Similarly, the average cost for hourly and weekly series is 130.735% and 9.057% higher than for daily series, respectively. When the model is DOTM, SCUM, or Predilab, the average cost is lower by 92.084%, 77.476%, and 31.395%, respectively, compared to the ATA-Damped model. Conversely, it is higher when the model corresponds to Theta Box-Cox, 4Theta, GROE, and THIEF by 1.990%, 3.572%, 212.646%, and 9.604%, respectively.

Figure 1 shows the most influential features in CO_2 emissions, obtained using a regression tree of depth four. The target variable is again considered on a logarithmic scale. The model was trained on the full sample. The most important variable for emissions, according to the tree, is whether the series was run with the DOTM model (the model with the lowest average emissions). Second is whether the series was run with the GROE model (the one with the highest average emissions). Third is whether the period of the series is annual (the period with the lowest average emissions). Fourth is the processor used for the experiment; fifth is whether the model corresponds to SCUM (second-lowest emissions); sixth is the length of the series; and seventh is the number of standard meth-

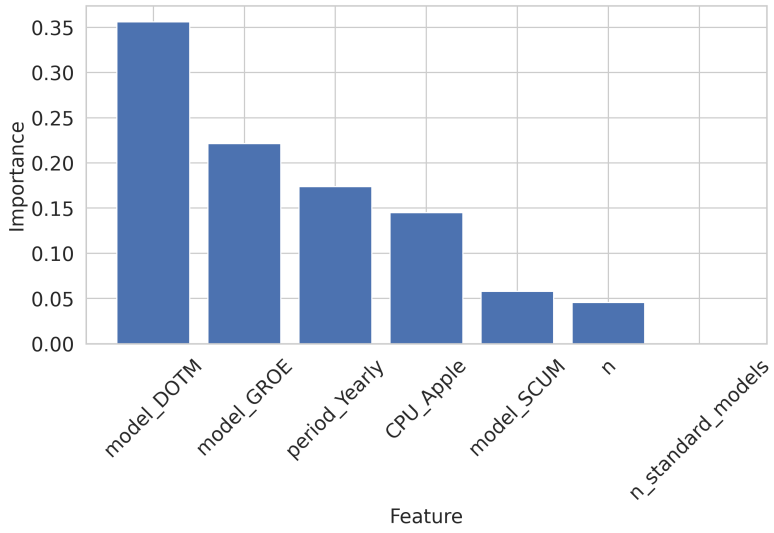


Figure 1: Most important features

ods used in the forecasting algorithm. In short, the most influential variables in the cost are the model used and the series period. Among the models, DOTM and GROE are the most decisive; among the periods, the annual period indicator is most influential.

4.3 Prediction: Estimating CO_2 Emissions

Once the variables with the most impact on environmental cost have been identified, the focus shifts to finding a model capable of providing accurate and robust predictions. Developing a method to estimate CO_2 emissions per series for the selected models is of great importance, as it allows evaluation of the environmental cost for the full M4 series set without needing to execute the competitors algorithms for each series.

Table 7 (Appendix 3) presents the average value of various metrics (RMSE, MAE, and R^2) across different training and test sets obtained by four different models: the linear regression model and regression tree from the previous section, as well as an XGBoost model and a RF model. In all cases, the target variable is log-transformed. The table also includes the values of the hyperparameters used for each model.[4]

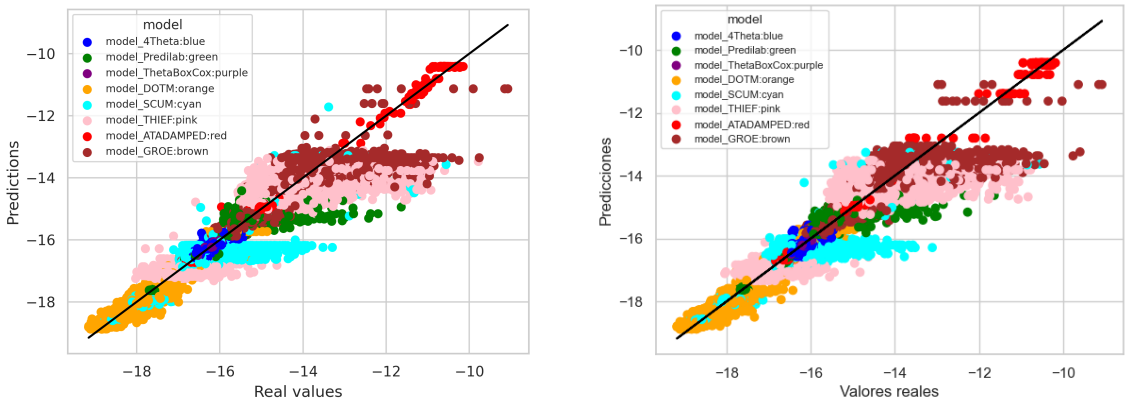
To compute the above metrics, the dataset is randomly split into a training set (80% of the sample) and a test set (remaining 20%). Each model is fitted on the training set and used to predict both the training and test sets. To reduce the impact of a specific sample split, this procedure is repeated five times, and the reported result is the average value of the evaluation metrics (common cross-validation exercise). The standard deviation for each metric is also calculated and is close to 0 in all cases. This indicates that, across all sample splits, the metric values are practically identical.

According to Table 7, XGBoost and Random Forest show considerably lower error metrics on average and higher R^2 values compared to the Linear Regression and Regression Tree

[4]If a hyperparameter is not mentioned, the default value from the corresponding libraries is used: `Scikit-learn` (for Regression Tree and RF) and `XGBoost` (for XG) in `Python`.

models. In all models, the difference between the average training and test scores is similar, so there appears to be no overfitting. RF and XG have very similar evaluation metrics, with RF being slightly better overall.

Table 8 (Appendix 3) shows various evaluation metrics conditional on the model. For its calculation, the training and test datasets were partitioned in the same proportions as before. It is interesting to observe that, indeed, the error metrics are similar across competition algorithms for both models, except for ATA-Damped, where RF seems to better capture the data patterns. Figure 2 shows how both models perform. The X-axis corresponds to the true values of the observations, while the Y-axis shows the predictions made by RF (left) and XG (right). Ideally, the points should lie along the line $y = x$. Once again, the plots show that RF predicts better than XG for ATA-Damped (in red).



(a) Predictions using RF

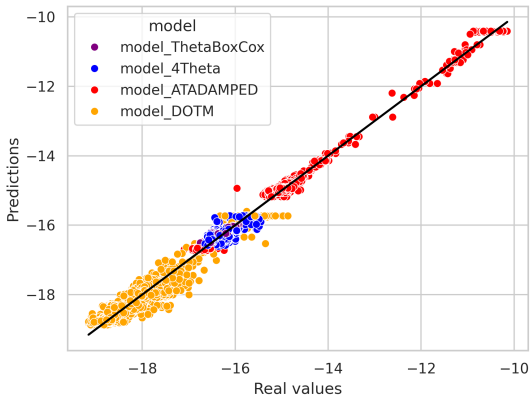
(b) Predictions using XG

Figure 2: Predictions for each model (RF and XG)

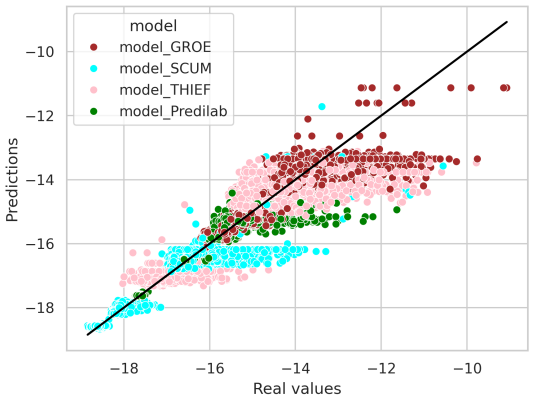
Another interesting result is the large variation in prediction quality across models. Figure 3 presents two plots (only for RF), where, as in Figure 2, the X-axis shows the actual values and the Y-axis the model predictions. We see that prediction quality for DOTM, Theta Box-Cox, 4Theta, and ATA-Damped (right plot) is better than for SCUM, Predilab, GROE, and THIEF (left plot). DOTM, Theta Box-Cox, and 4Theta are seemingly simpler algorithms than SCUM, Predilab, GROE, and THIEF, as they consist of a single forecasting method. It seems, therefore, that cost variability for series with similar lengths and belonging to the same period is greater for the more complex models than for the simpler ones.

Figure 4 shows the prediction accuracy of SCUM, Predilab, GROE, and THIEF for different periods. We observe that accuracy is higher for annual and quarterly periods compared to monthly. The fit for Predilab in weekly and daily series (recall that cost could not be measured for hourly series in Predilab) also shows better predictions than for monthly. The weekly and daily predictions for GROE also appear better than monthly and hourly predictions (GROE's hourly predictions correspond to the brown points at the top right in plot (c) of Figure 4). This is likely due to the fact that some periods, like annual or quarterly, show lower cost variability compared to others, such as monthly or hourly.

In summary, RF and XG show similar results in predicting environmental cost. RF

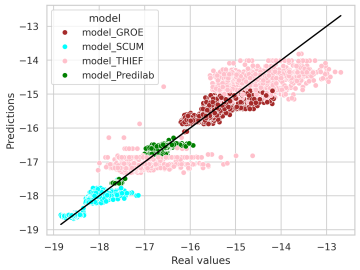


(a) Predictions for DOTM, Theta Box-Cox, 4Theta, and ATA-Damped

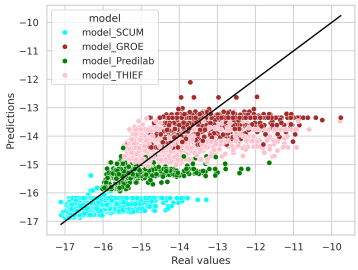


(b) Predictions for Predilab, THIEF, SCUM, and GROE

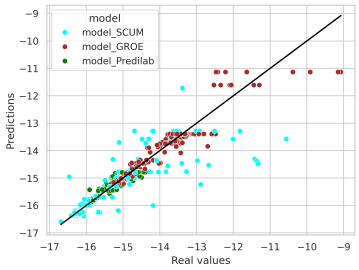
Figure 3: Predictions for each model (RF)



(a) Annual and Quarterly



(b) Monthly



(c) Weekly, Daily, and Hourly

Figure 4: Predictions for Predilab, THIEF, SCUM, and GROE by period

is selected due to its superior performance on the ATA-Damped model. On the other hand, the quality of the predictions in estimating CO_2 emissions is conditioned by the forecasting model and the series period, as there tends to be greater variability in the target variable depending on the complexity of the model used to generate the prediction horizons and on the series period.

5 Conclusions and Future Research

This study calculated the environmental cost during the training and prediction phases for various time series using eight models from the M4 competition. Based on the generated results, several machine learning models were used to estimate a cost function with two goals: to study the explanatory variables that most influence CO_2 emissions, and to develop an algorithm that allows predicting the environmental cost of processing series during the training and forecasting stages for the selected competition methods. For the first goal, interpretable algorithms such as a Linear Regression model and a Regression Tree were used. For the second, more complex predictive algorithms such as Random Forest and XGBoost were employed, aiming to better capture nonlinear relationships in the data.

The Linear Regression model and Regression Tree show how the explanatory variables relate to cost, and to what extent. The time series forecasting model and the series period are the most influential variables on CO_2 emissions. Among the competition models, DOTM (the model with the lowest average cost for the sample considered from the M4 dataset) and GROE (the model with the highest average cost for the sample) have the greatest impact on emissions. Regarding the series period, the annual frequency shows the highest influence.

With the Random Forest and XGBoost models, we have been able to construct a model that estimates CO_2 emissions based on the values of various explanatory variables. Since these variables are ex-ante, the model can be used to estimate the environmental cost for series not included in the training sample. This application may be useful for ranking models according to, for example, their average cost per series, enabling an evaluation not only in terms of accuracy (as done in M4) but also in terms of environmental cost—without needing to execute them. It has been shown that the predictive model performs well for less complex methods and less accurately for more complex ones. This is explained by the greater variability in environmental cost as model complexity increases. The fit of the cost function also depends on the series period. For example, it was observed that, in general, for complex models, the prediction quality is much higher for annual or quarterly series than for monthly ones. This is explained by the high-cost variability in certain periods (such as monthly or hourly) compared to others (such as annual or quarterly).

Future research will aim to improve prediction quality for the more complex models. For this, additional explanatory variables will be considered to better capture the complexity of the forecasting method used. The variability of cost in CO_2 emissions will be addressed using various approaches. Alternative transformations to the logarithmic one will be tested to correct variability, a refined outlier analysis will be conducted, experiments will be repeated using different emissions libraries such as `CodeCarbon` to detect possible measurement errors, and the possibility of building a cost function for each model will be evaluated. Additionally, the new tool `ecoRETINA` will be used for specifying and selecting the best cost models, in order to improve both their interpretability and predictive power.

References

Anthony, L. F. W., Kanding, B., and Selvan, R. (2020). Carbontracker: Tracking and predicting the carbon footprint of training deep learning models.

Assimakopoulos, V. and Nikolopoulos, K. (2000). The theta model: a decomposition approach to forecasting. *International Journal of Forecasting*, 16(4):521–530.

Athanasopoulos, G., Hyndman, R. J., Kourentzes, N., and Petropoulos, F. (2017). Forecasting with temporal hierarchies. *European Journal of Operational Research*, 262(1):60–74.

Budennyy, S. A., Lazarev, V. D., Zakharenko, N. N., Korovin, A. N., Plosskaya, O. A., Dimitrov, D. V. E., Akhripkin, V. S., Oseledets, I. V., Barsola, I. S., Egorov, I. V., Kosterina, A. A., and Zhukov, L. E. E. (2022). Eco2ai: Carbon emissions tracking of machine learning models as the first step towards sustainable ai. In *Doklady Mathematics*, volume 106, pages S118–S128, Moscow. Pleiades Publishing.

Dhar, P. (2020). The carbon impact of artificial intelligence. *Nature Machine Intelligence*, 2(8):Article 8.

Fiorucci, J. A., Pellegrini, T. R., Louzada, F., Petropoulos, F., and Koehler, A. B. (2016). Models for optimising the theta method and their relationship to state space models. *International Journal of Forecasting*, 32(4):1151–1161.

Heguerte, L. B., Bugeau, A., and Lannelongue, L. (2023). How to estimate carbon footprint when training deep learning models? a guide and review.

Hyndman, R., Koehler, A. B., Ord, J. K., and Snyder, R. D. (2008). *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688.

Kaack, L. H., Donti, P. L., Strubell, E., Kamiya, G., Creutzig, F., and Ronick, D. (2021). Aligning artificial intelligence with climate change mitigation. *Nature Climate Change*.

Makridakis, S. (1993). Accuracy measures: theoretical & practical concerns. *International Journal of Forecasting*, 9(4):527–529.

Makridakis, S. (2020). The m5 competition: A practitioners view. *International Journal of Forecasting*, 36(1):7–11.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2018). The m4 competition: Results, findings, conclusion, and way forward. *International Journal of Forecasting*, 34(4):802–808.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2020). The m4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1):54–74.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. (2020). Green ai. *Communications of the ACM*, 63(12):54–63.

Spiliotis, E., Assimakopoulos, V., and Makridakis, S. (2020). Generalizing the theta method for automatic forecasting. *European Journal of Operational Research*, 284(2):550–558.

Strubell, E., Ganesh, A., and McCallum, A. (2019). Energy and policy considerations for deep learning in nlp.

Svetunkov, I., Kourentzes, N., and Ord, J. K. (2022). Complex exponential smoothing. *Naval Research Logistics (NRL)*, 69(8):1108–1123.

Treviso, M., Lee, J. U., Ji, T., Van Aken, B., Cao, Q., Ciosici, M. R., Hassid, M., Heafield, K., Hooker, S., Raffel, C., Martins, P. H., Martins, A. F. T., Forde, J. Z., Milder, P., Simpson, E., Slonim, N., Dodge, J., Strubell, E., Balasubramanian, N., and et al. (2023). Efficient methods for natural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 11:826–860.

Yao, Z., Lum, Y., Johnston, A., Mejia-Mendoza, L. M., Zhou, X., Wen, Y., Aspuru-Guzik, A., Sargent, E. H., and Seh, Z. W. (2023). Machine learning for a sustainable energy future. *Nature Reviews Materials*, 8(3):Article 3.

Yapar, G., Capar, S., Selamlar, H. T., and Yavuz, I. (2018). Modified holt's linear trend method. *Hacettepe Journal of Mathematics and Statistics*, 47(5):1394–1403.

Yapar, G., Selamlar, H. T., Capar, S., and Yavuz, I. (2019). Ata method. *Hacettepe Journal of Mathematics and Statistics*, pages 1–7.

Appendix

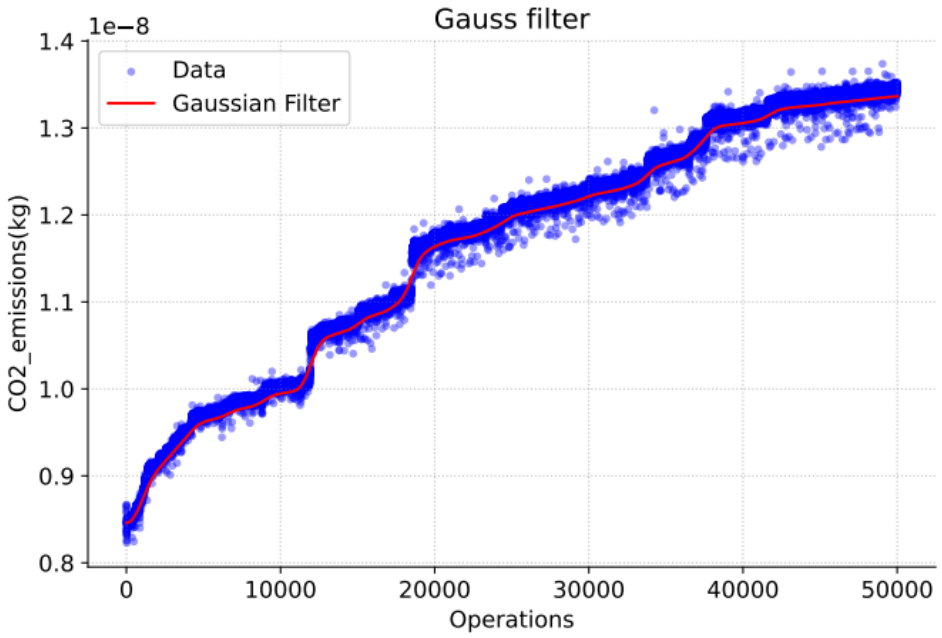
A1. Eco2AI in R

```
# Using Eco2AI in R

# Install reticulate and import the "reticulate" library

install.packages("reticulate")
library(reticulate)

# Install Eco2AI
py_install("eco2ai")

# Import the Tracker
Tracker <- import("eco2ai", convert = FALSE)$Tracker
track <- Tracker(project_name="Your_project_name",
experiment_description="Your_experiment_description")

# Use it to measure emissions from the desired code
track$start()

#Your code

track$stop()
```

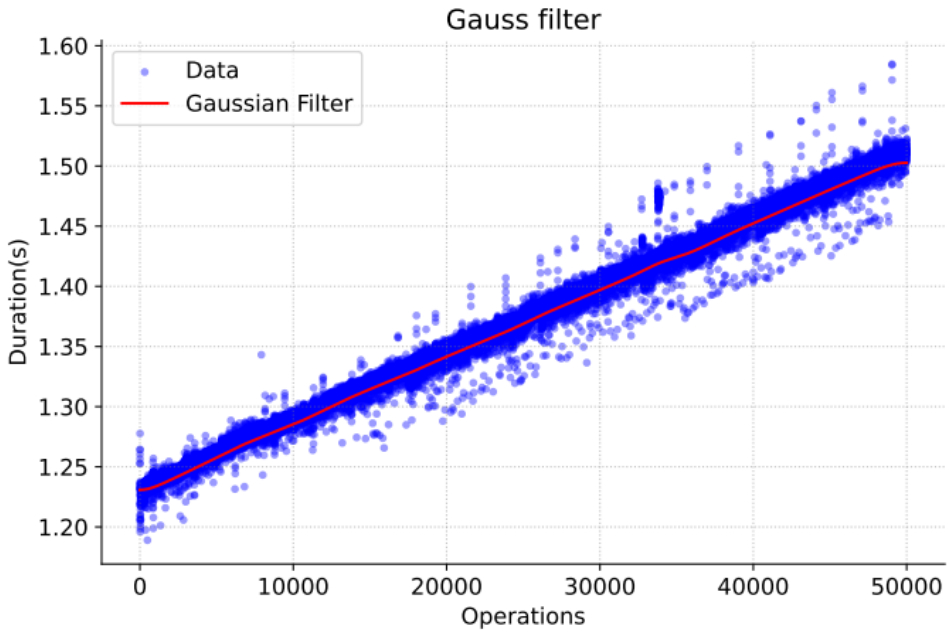
A2. Issues with `Eco2AI`

- **`Eco2AI` and parallel programming methods.**
  `Eco2AI` encounters problems when measuring emissions for algorithms that use parallel programming. The internal structure of the library can be accessed at https://github.com/sb-ai-lab/Eco2AI. The main issue lies in the *data* folder. When the *Tracker* is initialized, information is saved in a file called *config.txt*. The problem is that the *config* file is shared among all *Trackers*, even if a separate one is defined for each core. This results in errors that prevent proper information capture. For example, if emissions for one experiment are being measured and, within the same process, another experiment starts recording, the *config.txt* file will store the name of the second experiment but the data from the one that finishes first. This leads to information loss, incorrect storage, or even errors if two *Trackers* attempt to access *config.txt* simultaneously.

- **Correction of bias in environmental cost metrics.**
  Various tests were conducted to evaluate the measurements provided by `Eco2AI`. One of them investigated whether the environmental impact metrics include the cost of reading the `csv` file used to write the library's variable information. The result showed that, indeed, the metrics are contaminated by the impact of reading and writing this results file. Furthermore, the bias increases as the size of the `csv` grows (since reading/writing large files costs more than small ones). After several simulations, a Polynomial Regression model was used to estimate the bias generated based on the number of rows (or file size in bytes) of the experiment results file, for the variables Duration, Energy Consumption (kWh), and CO_2 Emissions (kg). Finally, the previously estimated bias was subtracted from the initial measurements to obtain environmental cost metrics cleaned of read/write effects. Figure 5 shows this impact and the estimated values conditioned on the number of file rows.



(a) Bias estimation in CO_2 emissions



(b) Bias estimation in experiment duration

Figure 5: Bias estimation in environmental impact metrics

A3. Tables and Figures

	Annual	Quarterly	Monthly	Weekly	Daily	Hourly	Total	Computer
DOTM	5290	5520	11040	83	972	95	23000	Intel
SCUM	5520	5760	11520	86	1014	99	23999	Apple
Theta Box-Cox	5750	6000	12000	90	1057	103	25000	Apple
4Theta	9200	9600	19200	144	1691	166	40001	Apple
Predilab	4600	4800	9600	72	845	-	19917	Apple
THIEF	5520	5760	11520	-	-	-	22800	Intel
ATA-Damped	5290	5520	11040	83	972	95	23000	Intel
GROE	5750	6000	12000	90	1057	103	25000	Apple

Table 4: Number of series and computer used for each model

Statistics	Value
Mean (μ)	$3.301215 \cdot 10^{-7}$
Standard deviation (σ)	$1.747820 \cdot 10^{-6}$
Minimum (min)	$8.242022 \cdot 10^{-10}$
25th Percentile (25%)	$5.442844 \cdot 10^{-8}$
Median (50%)	$8.541018 \cdot 10^{-8}$
75th Percentile (75%)	$2.503835 \cdot 10^{-7}$
Maximum (max)	$1.144927 \cdot 10^{-4}$

Table 5: Descriptive statistics of CO_2 emissions (kg)

Model	CO_2 Emissions (kg)
DOTM	$1.550643 \cdot 10^{-8}$
SCUM	$7.104638 \cdot 10^{-8}$
ThetaBoxCox	$8.054801 \cdot 10^{-8}$
4Theta	$8.289471 \cdot 10^{-8}$
Predilab	$1.471924 \cdot 10^{-7}$
THIEF	$6.813332 \cdot 10^{-7}$
ATADAMPED	$8.133756 \cdot 10^{-7}$
GROE	$8.943351 \cdot 10^{-7}$

Table 6: Average CO_2 emissions (kg) per model and series

Model	Set	RMSE	MAE	R^2	σ RMSE	σ MAE	σ R^2
Linear Regression	Training	0.5587	0.3912	0.8335	0.000673	0.000341	0.000364
	Test	0.5608	0.3922	0.8321	0.002684	0.001164	0.001473
Regression Tree	Training	0.6066	0.3893	0.8037	0.000684	0.000549	0.000372
(max_depth = 4,	Test	0.6069	0.3898	0.8033	0.002787	0.002150	0.001500
min_samples_leaf = 100)							
XGBoost	Training	0.3124	0.1579	0.9479	0.000285	0.000162	0.000115
(eta = 0.1,	Test	0.3181	0.1616	0.9460	0.002084	0.000868	0.000578
n_estimators = 300,							
max_depth = 14)							
Random Forest	Training	0.3076	0.1561	0.9495	0.000500	0.000163	0.000174
(n_estimators = 300,	Test	0.3161	0.1608	0.9466	0.002033	0.000785	0.000627
max_depth = 14)							

Table 7: Average evaluation metrics

Model	**RF**			**XG**		
	RMSE	**MAE**	**R2**	**RMSE**	**MAE**	**R2**
DOTM	0.182681	0.136449	0.876563	0.187083	0.138626	0.870420
SCUM	0.357418	0.192964	0.880681	0.352372	0.188608	0.885448
Theta Box-Cox	0.042166	0.029185	0.920766	0.040791	0.027316	0.924703
4Theta	0.085146	0.051944	0.809032	0.084019	0.050812	0.808860
Predilab	0.283632	0.145013	0.908763	0.253008	0.139464	0.927551
THIEF	0.633477	0.454762	0.711182	0.637067	0.460388	0.707761
ATA-Damped	0.090898	0.068355	0.992256	0.097589	0.070595	0.990665
GROE	0.465232	0.292579	0.737699	0.458715	0.290600	0.753879

Table 8: Evaluation metrics in RF and XG

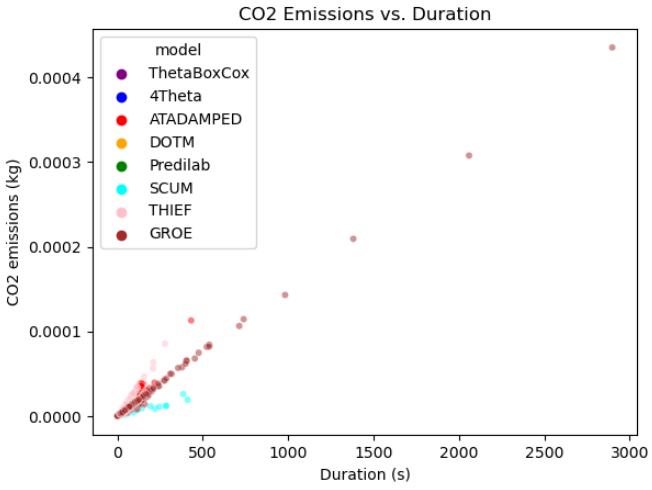


Figure 6: CO_2 emissions vs Duration

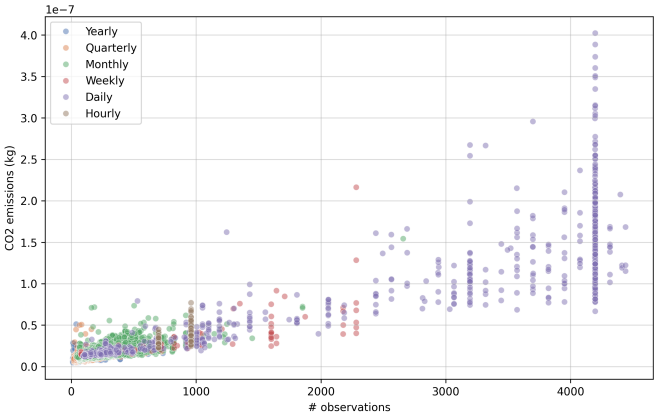


Figure 7: CO_2 emissions for different series lengths in the DOTM model

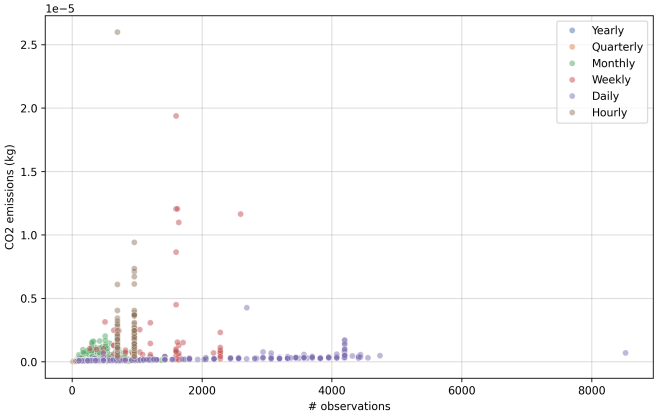


Figure 8: CO_2 emissions for different series lengths in the SCUM model

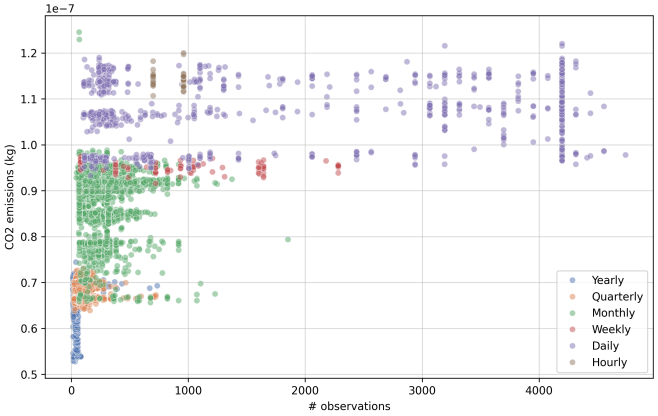


Figure 9: CO_2 emissions for different series lengths in the Theta Box-Cox model

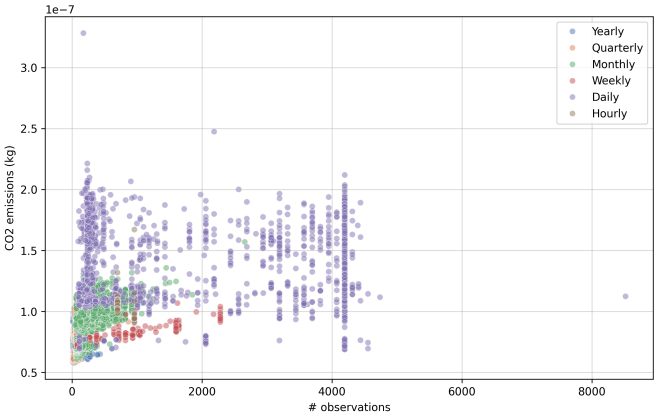


Figure 10: CO_2 emissions for different series lengths in the 4Theta model

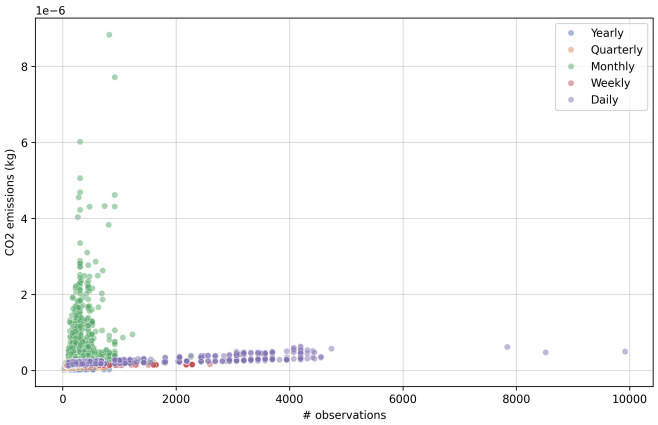


Figure 11: CO_2 emissions for different series lengths in the Predilab model

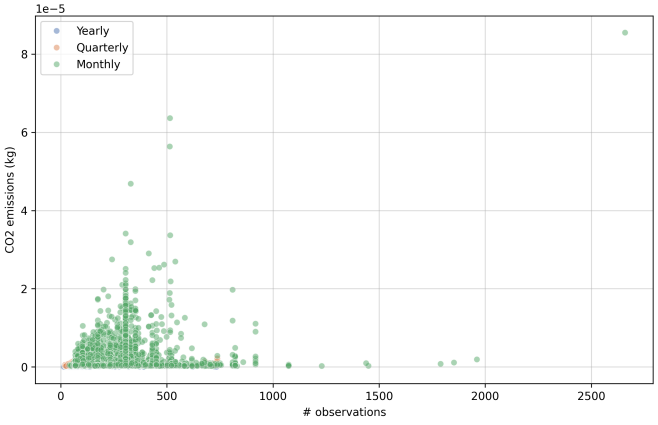


Figure 12: CO_2 emissions for different series lengths in the THIEF model

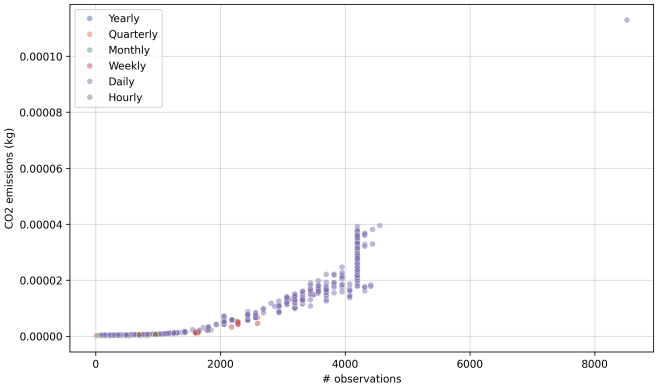


Figure 13: CO_2 emissions for different series lengths in the ATA-Damped model

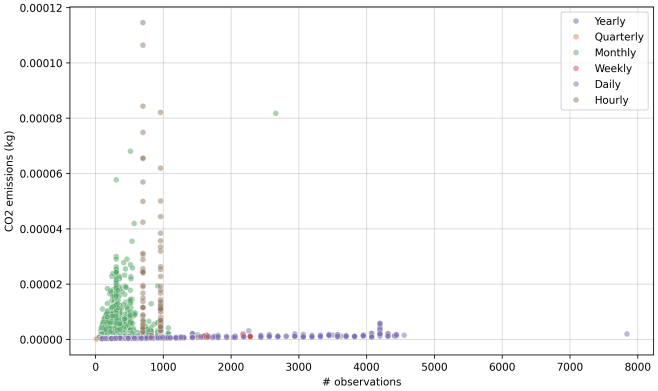


Figure 14: CO_2 emissions for different series lengths in the GROE model

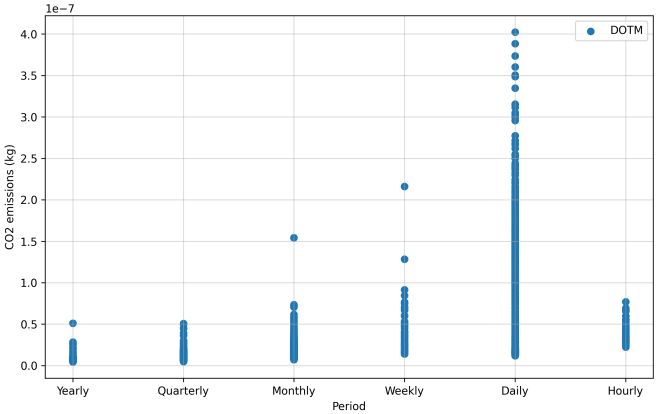


Figure 15: CO_2 emissions for different periods in the DOTM model

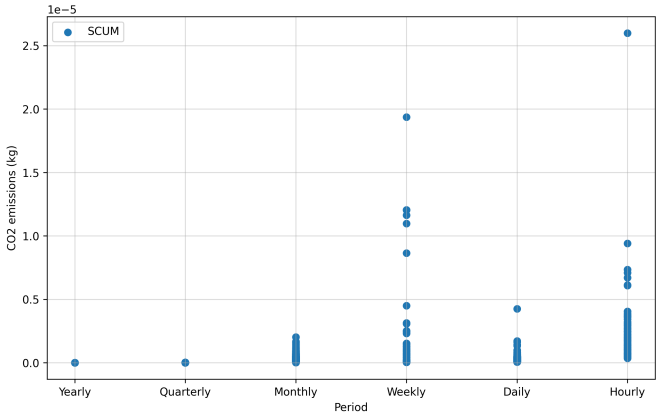


Figure 16: CO_2 emissions for different periods in the SCUM model

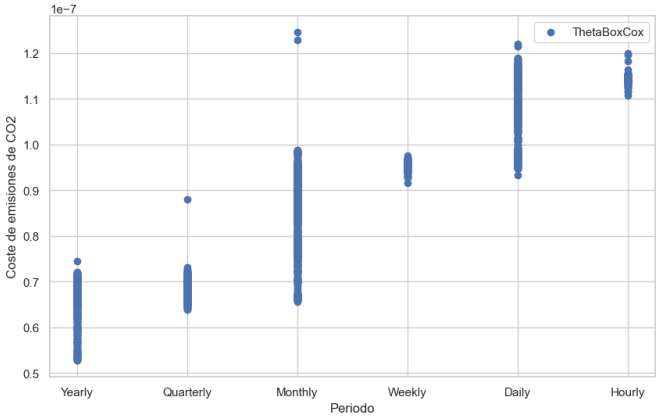


Figure 17: CO_2 emissions for different periods in the Theta Box-Cox model

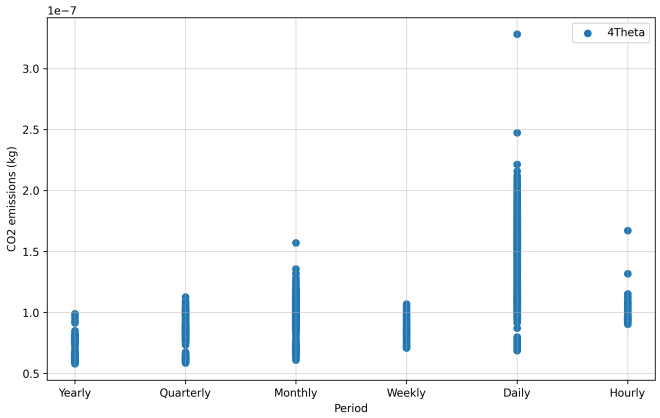


Figure 18: CO_2 emissions for different periods in the 4Theta model

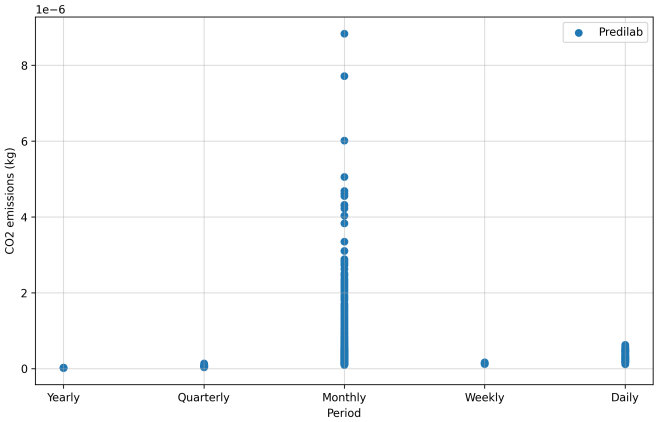


Figure 19: CO_2 emissions for different periods in the Predilab model

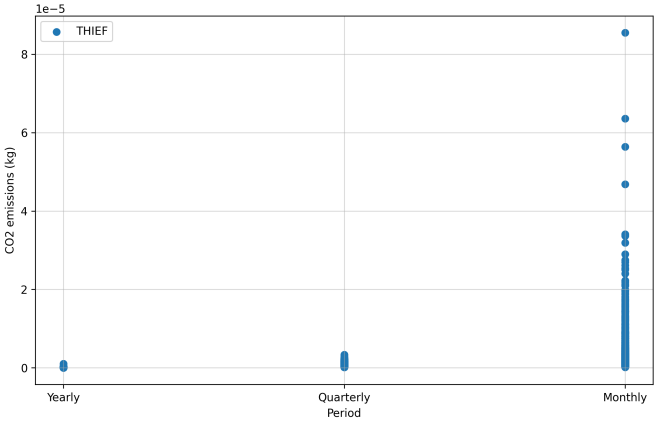


Figure 20: CO_2 emissions for different periods in the THIEF model

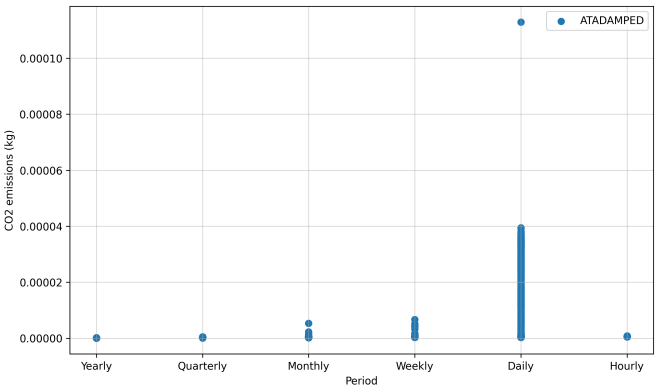


Figure 21: CO_2 emissions for different periods in the ATA-Damped model

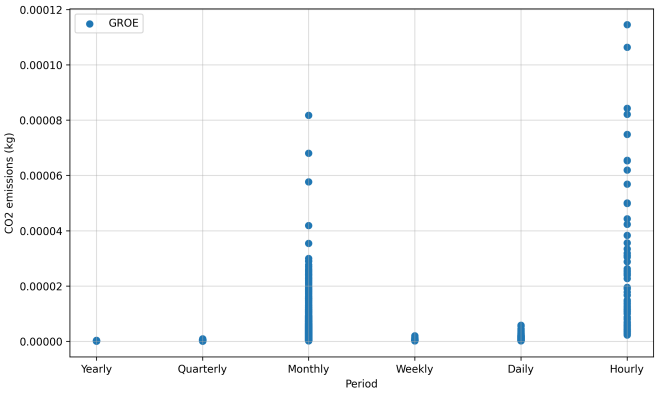


Figure 22: CO_2 emissions for different periods in the GROE model

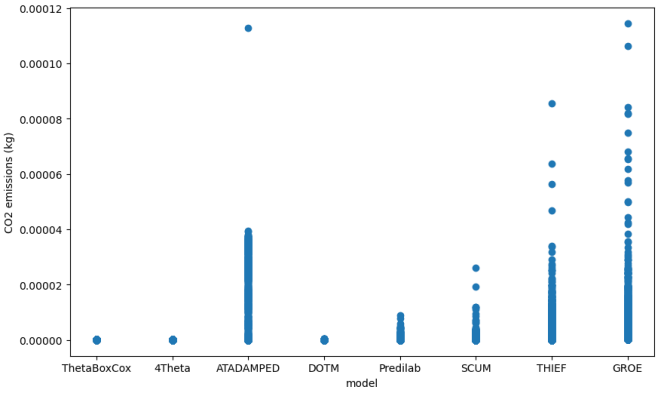


Figure 23: CO_2 emissions of the studied models

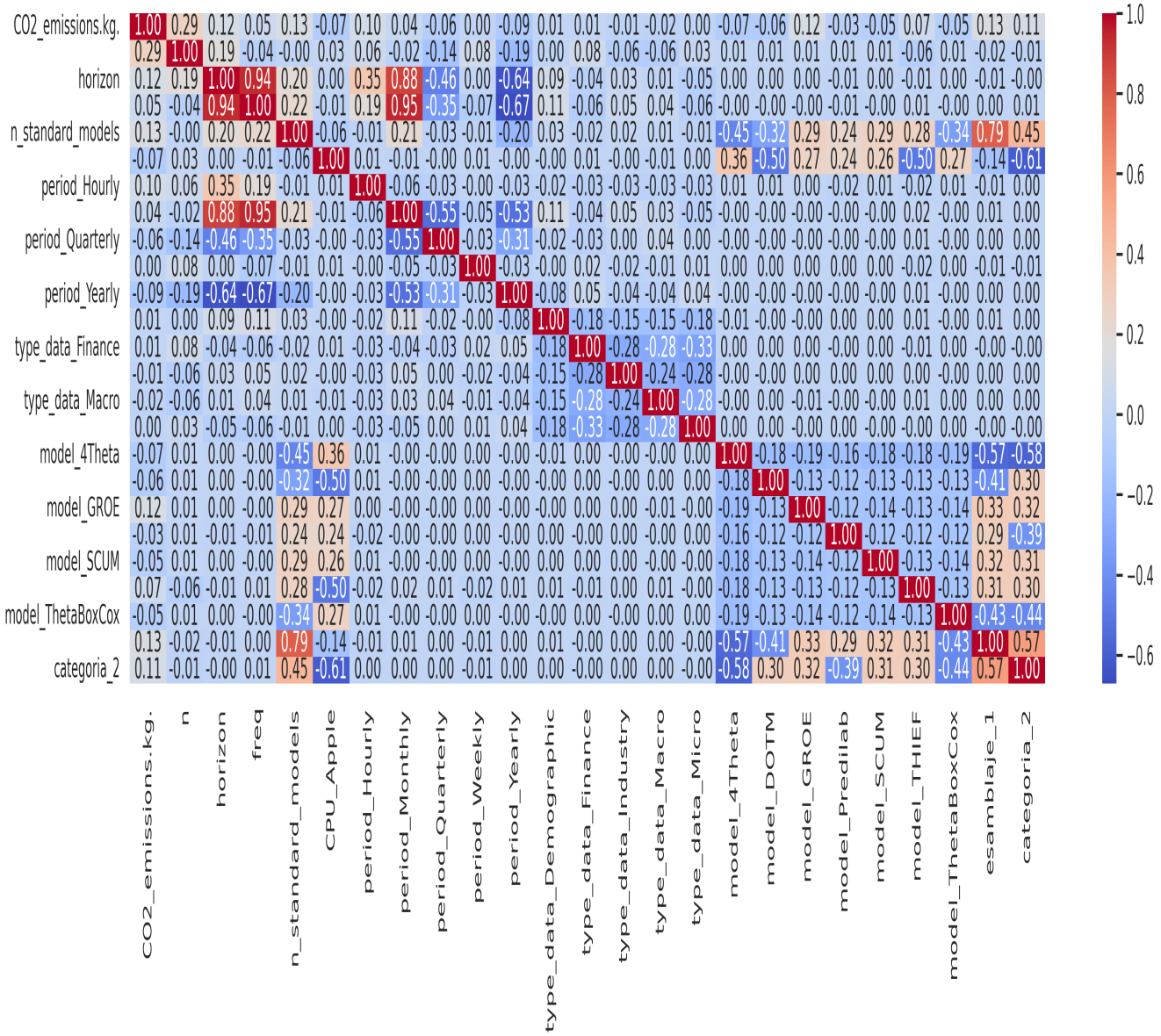


Figure 24: Matrix of linear correlations

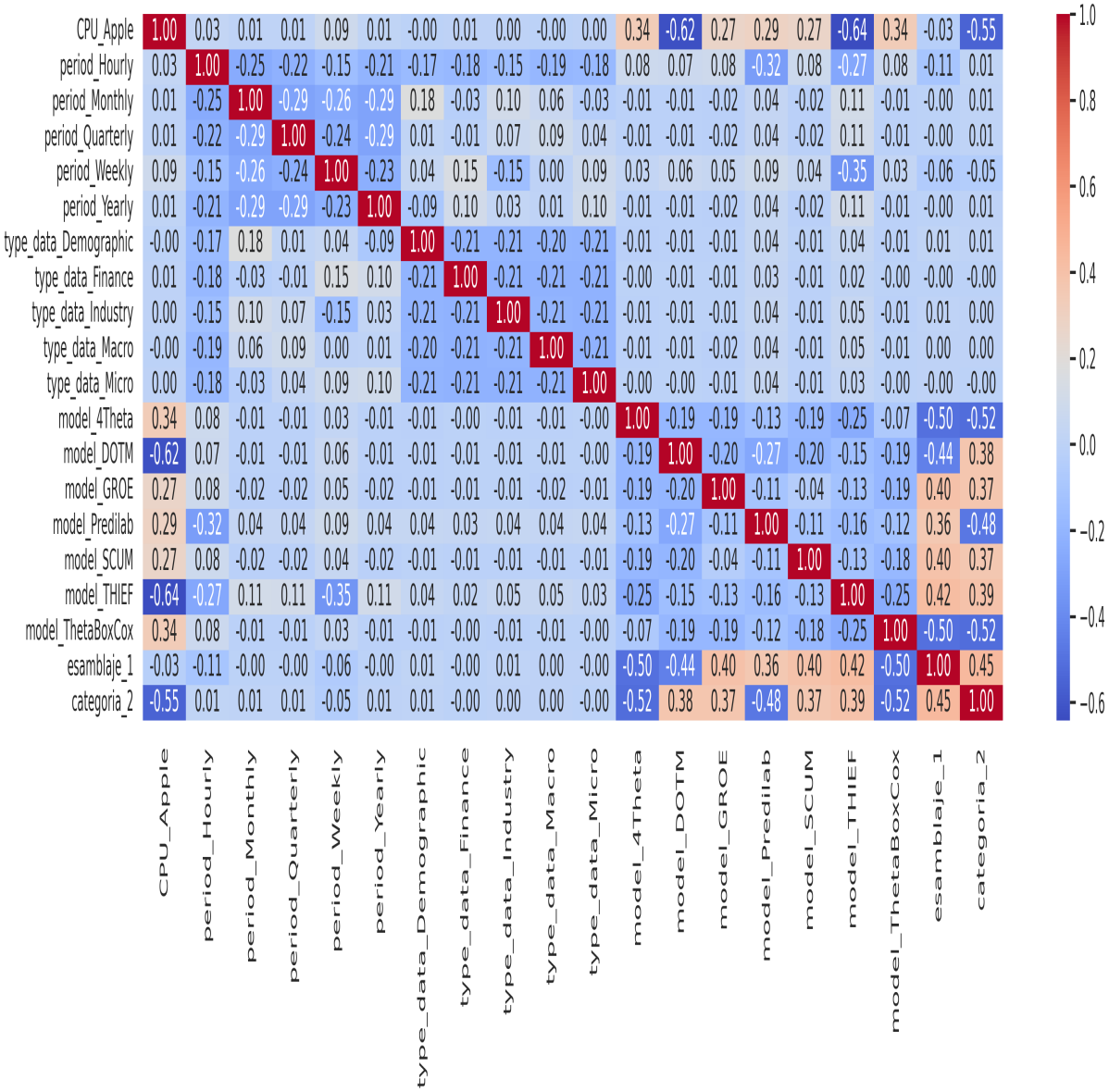


Figure 25: Matrix of polychoric correlations