engineering proceedings

MDPI

Proceeding Paper

Evaluation of Heuristics for Taken's Theorem Hyper-Parameters Optimization in Time Series Forecasting Tasks †

Rodrigo Hernandez-Mazariegos [1], Jose Ortiz-Bejar [2,*] and Jesus Ortiz-Bejar [1]

[1] Facultad de Ciencias Físico-Matemáticas "Mat. Luis Manuel Rivera Gutiérrez", UMSNH, Avenida Universidad 100, Villa Universidad, 58060 Morelia, Michoacán, Mexico; 1301441a@umich.mx (R.H.-M.); jesus.ortiz@umich.mx (J.O.-B.)

[2] División de Estudios de Posgrado de la Facultad de Ingeniería Eléctrica, UMSNH, Building "Ω2" Ciudad Universitaria, Francisco J. Múgica S/N, 58030 Morelia, Michoacán, Mexico

* Correspondence: jose.ortiz@umich.mx

† Presented at the 9th International Conference on Time Series and Forecasting, Gran Canaria, Spain, 12–14 July 2023.

Abstract: This study compares three methods for optimizing the hyper-parameters m (embedding dimension) and τ (time delay) from Taken's Theorem for time-series forecasting to train a Support Vector Regression system (SVR). Firstly, we use a method which utilizes Mutual Information for optimizing τ and a technique referred to as "Dimension Congruence" to optimize m. Secondly, we employ a grid search and random search, combined with a cross-validation scheme, to optimize m and τ hyper-parameters. Lastly, various real-world time series are used to analyze the three proposed strategies.

Keywords: Taken's Theorem; time-series; SVR forecasting; mutual information; dimension congruence; random search; grid search

check for updates

Citation: Hernández-Mazariegos, R.; Ortiz-Bejar, J.; Ortiz-Bejar, J. Evaluation of Heuristics for Taken's Theorem Hyper-Parameters Optimization in Time Series Forecasting Tasks. *Eng. Proc.* **2023**, *39*, 71. https://doi.org/10.3390/engproc2023039071

Academic Editors: Ignacio Rojas, Hector Pomares, Luis Javier Herrera, Fernando Rojas and Olga Valenzuela

Published: 10 July 2023

1. Introduction

Several complex phenomena are often modeled as a sequence of states. This sequence is known as the phase space. A time series is a finite sequence of states in a dynamical system measured directly or indirectly. A relevant approach to perform time series analysis is Taken's embedding theorem [1], which states that, from a sequence of states $S = \{y_{t_1}, y_{t_2}, \ldots, y_{t_n}\}$ (i.e., time series) in a dynamical system, it is possible to generate all the system's phase space U. More specifically, for a sequence of observations x of dimension m (embedding dimension) and a constant τ (time delay), there exists a function f such as:

$$y(t) = f(x) = f[y(t-\tau), y(t-2\tau), \ldots, y(t-(m-1)\tau)] \quad (1)$$

From Equation (1) it can be inferred that, given a time series S, it is possible to predict the state at time t (hereafter, y_t) by using m previous observations sampled at frequency τ. The two problems, and their solutions, are the following: (1) the function f is often too complex to be found analytically, which is when machine learning algorithms comes into play with the objective of using a supervised learning algorithm to learn f; (2) it is necessary to find the correct modeling for the time series, i.e., the optimal values for m and τ, for which Random search, Grid search, and Mutual information + Dimension Congruence can be used.

2. Theoretical Background

Given Equation (1), the first task is to find the optimal value for the time delay τ and embedding dimension m.

2.1. Mutual Information

Regarding the τ, in [2] Cao, L. proposes using mutual information. The process relies on making $y(t)$ and $y(t-\tau)$ as independent as possible to maximize the information obtained from each variable in the reconstruction of the phase space. To achieve this, the *mutual information function* (2) can be applied:

$$I_\tau = \sum_{\Omega} P(N_{i+\tau}|N_i) \ln\left(\frac{P(N_{i+\tau}|N_i)}{P(N_{i+\tau})P(N_i)}\right) \tag{2}$$

Note the similarity of this with entropy, i.e., this function measures how surprising it is that $N_{i+\tau}$ results, given that N_i resulted, i.e., when $N_{i+\tau}$ and N_i are very independent, then $I_\tau \approx 0$. To find the τ it is, therefore, enough to minimize the function (2).

Once the τ is fixed, it is necessary to find the embedding dimension m. The latter is achieved by using the false neighbors [2] to determine the dimension congruence.

2.2. Dimension Congruence

The aim of this procedure is for the distances between neighbors (data close to each another) on dimension m of Equation (1) to be constant. To this end, firstly, the distance $E(i,j,m)$ between $y(t)$ and $y(t')$ in the dimension m is defined as the maximum between the differences of their components, as in Equation (3):

$$E(t,t',m) = \max_{(k,l)\in[0,m-1]} |y(t-k\tau) - y(t'-l\tau)| \tag{3}$$

Now, we can say that the nearest neighbor of $y(t)$ is $y(t')$ if t' is satisfied such that:

$$E(t,t',m) = \min_{t''\in[0,n-m\tau],t''\neq t} E(t,t'',m) \tag{4}$$

where n is the sample size, it is worth mentioning that t' depends on t so we call it $t'(t)$ and then we define the "nearest neighbor" congruence of $y(t)$ in m as:

$$F(t,m) = \frac{E(t,t'(t),m)}{E(t,t'(t),m+1)} \tag{5}$$

Note that in $F(t,m) \approx 1$ if $y(t'(t))$ is sufficiently congruent being the nearest neighbor of $y(t)$ in m, there is the possibility to define the "dimension congruence" of m as follows:

$$G(m) = \frac{1}{n-m\tau} \sum_{t\in[0,n-m\tau]} F(t,m) \tag{6}$$

In summary, the dimension congruence measures how true it is that the nearest neighbors continue to be nearest neighbors as the dimension increases, which is useful, given the assumption that there is an attractor in the system under study [2].

In this work, m was selected as lower m satisfying $G(m) > 0.95$. As alternative strategies to find m and τ, Evolutionary computation algorithms, Random Search(RS) and Grid Search (GS) can be used. In this paper we focus on comparing Mutual Information + Dimension Congruence with RS and GS, given [3], which states that random search is good enough for parameter optimization.

2.3. Random Search

Let f be a model that depends on a parameter λ. The random search method [3] involves defining a range (a_0, a_1) for λ, a probability distribution $g : (0,1) \longrightarrow (a_0, a_1)$, and the number of values to be tested. Then, n parameters $\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n$ are drawn from the distribution g, and the behavior of each of the corresponding models $f_{\lambda_1}, f_{\lambda_2}, f_{\lambda_3}, \ldots, f_{\lambda_n}$

is evaluated by computing a fitness function. The best-performing model, f_{λ_i}, is selected based on the fitness value.

2.4. Grid Search

In contrast with random search, the grid search method [4] involves sampling λ values equally spaced in the range (a_0, a_1), specifically, $\lambda_1 = a_0 + \frac{1}{n}, \lambda_2 = a_0 + \frac{2}{n}$, $\lambda_3 = a_0 + \frac{3}{n}, \ldots, \lambda_n = a_0 + \frac{n}{n} = a_1$.

From Figure 1 we can appreciate the differences between random search and grid search.

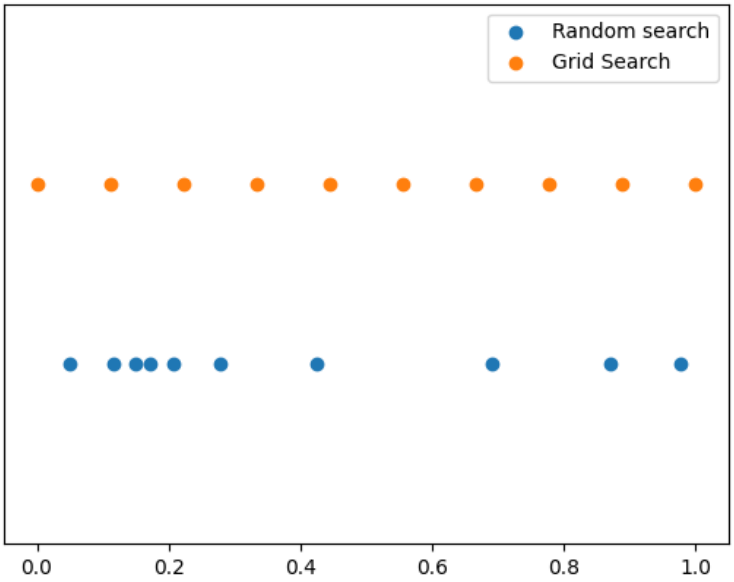


Figure 1. Differences between Random Search and Grid Search.

Having described the procedures for finding m and τ, it is time to describe the fitness measures used.

2.5. Fitness Function

The Mean Absolute Percentage Error (MAPE) is the fitness function determining the optimal value for parameters m and τ. Additionally, the Mean Squared Error (MSE) and the Coefficient of Determination (R^2) were used to compare the three optimization procedures with MAPE. For completeness, a brief description of each one is provided.

2.5.1. MAPE

MAPE [5] is a widely used measure in time series forecasting and seems to yield good results. Note that in Equation (7) MAPE takes the average of the absolute value of the errors expressed as a percentage of the actual value, and if it approaches 0 the better the fit, while if it approaches ∞ the worse the fit.

$$MAPE = \frac{1}{n-e} \sum_{i=e+1}^{n} \left| \frac{N_i - \hat{N}_i}{N_i} \right| \tag{7}$$

2.5.2. MSE

Ref. [6] is the average of the squared errors. If the model fits perfectly then $MSE = 0$. The closer it is to ∞ the worse it is. It is computed by using Equation (8):

$$MSE = \frac{1}{n-e} \sum_{i=e+1}^{n} \left(N_i - \hat{N}_i\right)^2 \tag{8}$$

2.5.3. R^2

Ref. [7] calculates the ratio between the model's variance and the actual data's variance. In other words, it ascertains how similar the predicted and actual data variances are. If they are equal, R^2 is equal to 1, which means the model fits perfectly. The worse value for R^2 is $-\infty$. To find R^2 Equation (9) is used:

$$R^2 = 1 - \frac{\sum_{i=e+1}^{n} \left(N_i - \hat{N}_i\right)^2}{\sum_{i=e+1}^{n} \left(N_i - \bar{N}\right)^2}, \quad \bar{N} = \frac{1}{n-e} \sum_{i=e+1}^{n} N_i \tag{9}$$

Finally, in the next section we describe the models we used for the experiments.

2.6. Support Vector Regression Algorithm (SVR)

The SVR algorithm is based on Support Vector Machine (SVM) Algorithm [8]. SVM is an algorithm for separating samples depending on the class they belong to. The algorithm works by increasing the size of the sample space via a kernel, and in that larger size, three parallel hyperplanes are constructed, separated by an ϵ distance each. The main idea is to optimize the kernel and the hyperplanes so that only a small number of samples, controlled by the parameter ξ, are outside the region to which they belong, i.e., the samples of one class belong to one side of the hypertube and those of the other class belong to the other side of the hypertube.

On the other hand, SVR aims for all the samples to be inside the hypertube and only a small amount of samples to be outside the hypertube, controlled by the parameter ξ, and, thus, uses the image of the central hyperplane projected in the original space to predict future values of the time series.

- $\mathbb{S}$, $\mathbb{H}_\epsilon$ and $\mathbb{H}_\epsilon$ are the parallel hyperplanes.
- $\mathbb{H}_\epsilon$ is located at a distance ϵ above $\mathbb{S}$.
- $\mathbb{H}_{-\epsilon}$ is located at a distance ϵ below $\mathbb{S}$.
- Then $\mathbb{H}_\epsilon$ and $\mathbb{H}_{-\epsilon}$ form the hypertube
- The quantity $\frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n}\|\vec{\xi}_i\|$ is the minimum possible, subject to $|N_i' - \vec{w}\cdot\vec{x}_i| < \|\vec{\epsilon}\| + \|\vec{\xi}_i\|$

Where $\vec{w}$ is the SVR kernel weight and $\vec{\xi}_i$ is the distance that the i-th data moves away from the hypertube, while C is a regularization parameter.

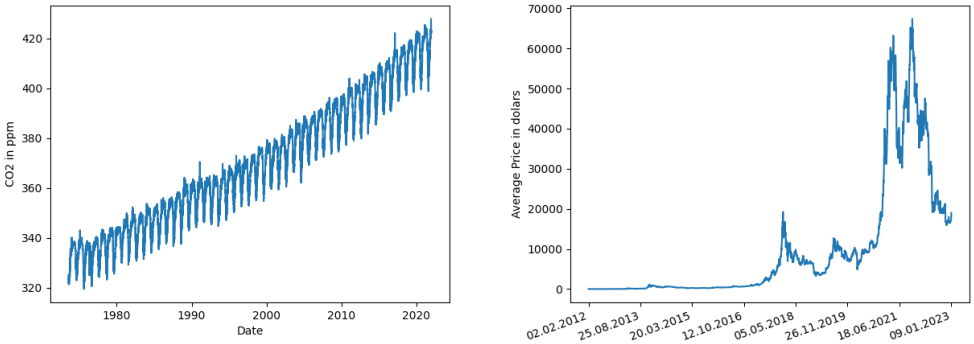
Note how the larger c is, the less freedom the data have to move out of the hypertube. The idea is to find a hypertube that approximates the data.

3. Experiments and Results

The study focused on six real-world time series, each representing a measurement of a real-world phenomenon. The aim was to examine more complex time series than artificially generated ones. The selected time series displayed a wide range of characteristics, including exponential and moderate growth patterns, general trends, and horizontal patterns. The goal was to evaluate the generalization ability of the proposed methodologies by considering time series with diverse characteristics. Below is a brief description of each of the time series used.

3.1. SARS-CoV-2 in Mexico (COV)

The time series data for this study was obtained from the General Direction of Epidemiology (https://www.gob.mx/salud/documentos/datos-abiertos-152127 (accessed on 19 December 2022)). It consists of the confirmed and suspected COVID-19 cases in Mexico. The data spans 1025 days, and the number of confirmed cases per laboratory ranges from 0 to 9800. The data were normalized such that the number of cases fell from 0 to 1. For the purposes of this study, this time series is referred to as *COV*. Figure 2a depicts the evolution of the COV time series.



(**a**) Evolution of SARS-CoV-2 in Mexico (**b**) Evolution of the Bitcoin Price

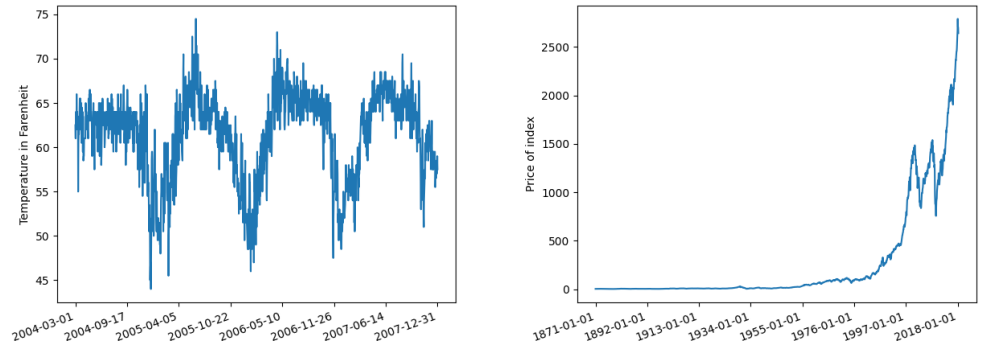
Figure 2. Covid and Bitcoin.

3.1.1. Bitcoin Price on Bitfinex (BIT)

This time series comprises daily variations in the price of Bitcoin in dollars, recorded on the Bitfinex platform between February 2012 and January 2023 (The data is available at https://www.investing.com/crypto/bitcoin/btc-usd-historical-data (accessed on 10 February 2023)). The dataset includes the daily high and low prices. The average price is computed as $(minimumPrice + maximumPrice)/2$. The time series were normalized between 0 and 1 for consistent analysis with the other time series used in this study. Figure 2b displays the evolution of the BIT time series.

3.1.2. Air Temperature in Acuitzio del Canje (TEM)

This time series consists of temperature data recorded by the MXN00016001 weather station located in Acuitzio del Canje between 2004 and 2007 (The data was obtained from https://www.ncei.noaa.gov/ (accessed on 15 January 2023)). The dataset comprises 1401 data points of daily minimum and maximum temperatures. The average temperature is calculated as $(T_{max} + T_{min})/2$. The data is recorded in degrees Fahrenheit and was subsequently normalized between 0 and 1 for comparison with other time series in this study. The evolution of the TEM time series is depicted in Figure 3a.



(**a**) Evolution of Air Temperature (**b**) S&P 500 Index

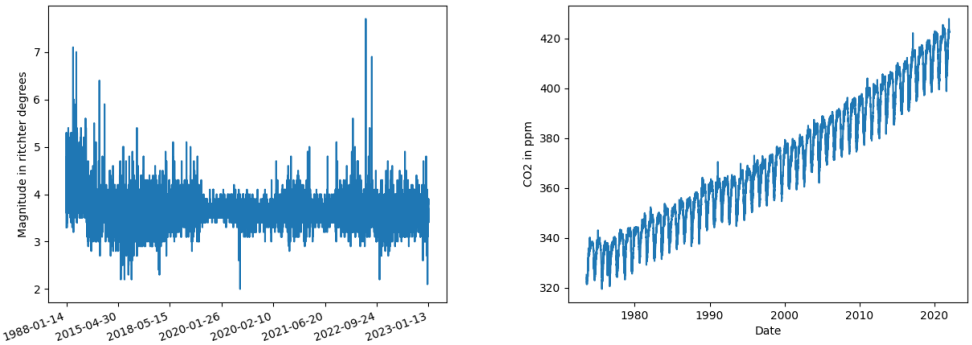
Figure 3. Temperature and S&P.

3.1.3. S&P500 Index

The S&P500 index series (Data was sourced from: https://datahub.io/core/s-and-p-500 (accessed on 11 February 2023)) is a monthly measurement of the value of the S&P500 stock index, which represents the 500 most valuable companies in the United States. It consists of 1768 monthly value data points calculated from 1871 to 2018. The data was normalized for analysis between the range of $(0, 1)$. Figure 3b shows the evolution of the S&P500 index graphically.

3.1.4. Seismic Activity in Michoacán

This series comprises seismic activity recorded by the National Seismological System in Michoacán (The time series is available at the following URL: http://www2.ssn.unam.mx:8080/catalogo/ (accessed on 22 January 2023)). The values cover the period from 1988 to 2023. This time series was of interest as the data was not evenly spaced. One possibility was to summarize the data to create an indicator to identify "how active each month was". However, for our study, the original sampling frequency was maintained. Each event is a numerical value representing its magnitude in Richter scale degrees and consists of 17,500 data points, which were normalized between $(0, 1)$. Figure 4a graphically depicts these data.



(**a**) Seismic Activity in Michoacán (**b**) Carbon Dioxide Concentration.

Figure 4. Seismicity and CO_2.

3.1.5. Atmospheric Carbon Dioxide Concentration

This is a series of daily atmospheric carbon dioxide (CO_2) concentration measurements taken at the Barrow Atmospheric Baseline Observatory (Data obtained from https://www.co2.earth/daily-co2 (accessed on 9 February 2023)) in the United States. The CO_2 concentrations are reported in parts per million (ppm) and cover the period from 1973 to 2021. To facilitate the analysis and interpretation of the data, all the values were normalized to the range of $(0, 1)$.

Figure 4b shows this time series.

3.2. Experimental Setup

For each of the analyzed time series, the parameters m,τ, and the parameters C and ϵ for the SVR were optimized. Three strategies were used: mutual information + congruence, grid search, and random search. The flow diagram in Figure 5 summarizes the process.

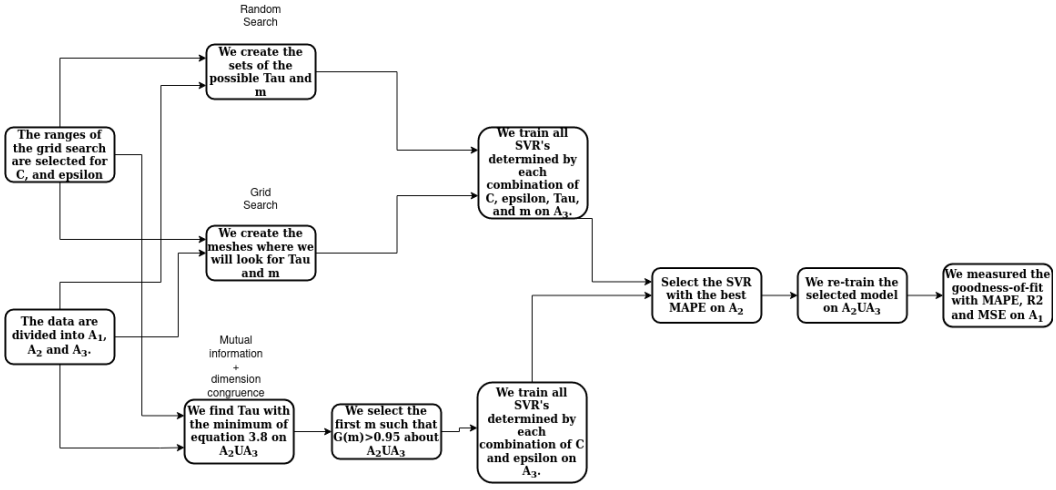


Figure 5. Flow diagram of each procedure applied to each time series.

The diagram in Figure 5 illustrates the general overview of the three procedures applied to each time series. It is noteworthy that the final outcome for each time series were nine goodness of fit measures. These measures were then used to compare the procedures. Before diving into the specifics of each process, it is essential to consider a few key points.

The data were divided into three sets:

- Set $\mathbb{A}_1$ contained the last 5% of the data to be used for testing and calculating the model's fitness.
- Set $\mathbb{A}_2$ contained the last 5% of the data once set $\mathbb{A}_1$ had been removed, to be used for hyper-parameter tuning.
- Set $\mathbb{A}_3$ consisted of the remaining data to be used for training the parameters.

All of the models used were Support Vector Regression (SVR), and for the SVR C and ϵ hyper-parameters, the following applied:

- Grid search was used to find the C and ϵ for all SVR models.
- The grid for C values was in $\mathbb{C} = [0.1, 1, 10, 100]$.
- The grid for ϵ values was $\mathbb{E} = [0.001, 0.01, 0.1, 1]$.

For RS and GS of τ and m, the following conditions were met:

- The sets $\mathbb{T}_i$, $\mathbb{M}_i$ contained all possible values of τ and m for each time series, and each procedure (Random Search and Grid Search) had 20 elements (for computational capacity reasons).
- The infimum of these sets was always 2.
- The supremum was always $int(\sqrt{|\Omega|} - 10)$ (so that $m * \tau < |\Omega|$).
- The distribution used for the random search was always uniform.

With this in mind, the procedures used to search m (dimension of the reconstructed phase space) and τ (delay) were:

- Mutual information + dimension congruence:
  1. Find τ using the mutual information function from Equation (2) on $\mathbb{A}_2 \cup \mathbb{A}_3$, and take the minimum.
  2. Find the embedding dimension by selecting the first m that satisfies $G(m) > 0.95$ in Equation (6) with the obtained τ on $\mathbb{A}_2 \cup \mathbb{A}_3$.
  3. Train all possible SVRs determined by the elements of $\mathbb{C} \times \mathbb{E}$ on $\mathbb{A}_3$.
  4. Select the model having MAPE on $\mathbb{A}_2$ which is the minimum.
  5. Measure the goodness of the selected model using MAPE on $\mathbb{A}_1$.
- Random search and grid search:
  1. Use each element of $\mathbb{C} \times \mathbb{E} \times \mathbb{T}_i \times \mathbb{M}_i$ to train $|\mathbb{C}||\mathbb{E}||\mathbb{T}_i||\mathbb{M}_i| = 4 * 4 * 20 * 20 = 6400$ models on $\mathbb{A}_3$.
  2. Select the model having MAPE on $\mathbb{A}_2$ which is the minimum.
  3. Measure the goodness of the selected model using MAPE on $\mathbb{A}_1$.

It is worth mentioning that the parameter space in both the grid search and the random search was not very large due to the lack of hardware. It is to be expected that enlarging the size of these spaces would improve the results.

Upon completion of the procedures, a comparison was made by evaluating the distributions generated by each of the fitness measures obtained by each proposed method.

3.3. Results

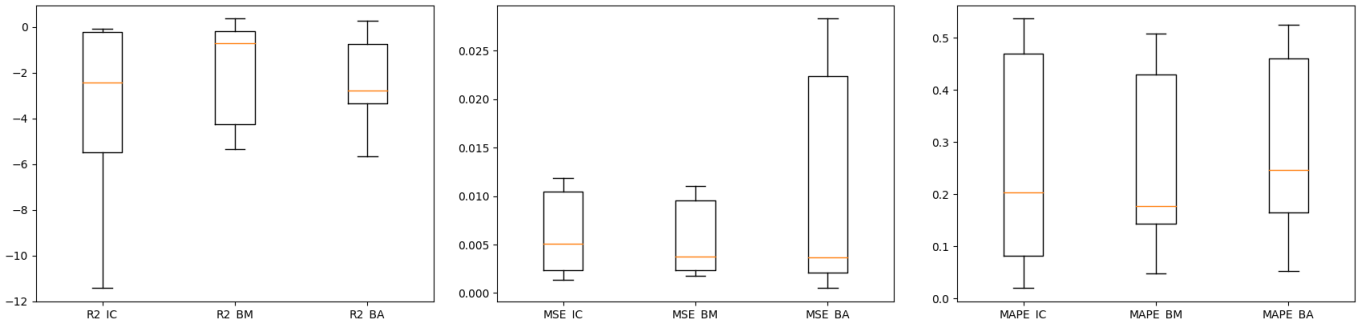
Table 1 shows the results for the three metrics. The MAPE, R^2 evaluation, and MSE, best results are boldfaced. For instance, in the series "BIT" GS was the best procedure with respect to MAPE, but also with respect to R^2 and with respect to MSE. As you can see, there was no procedure that was always better than another. There were some series where RS and IC were better than RS. However, it is essential to clarify that even though GS had better results, it is a brute force algorithm, in that, although it has better optimizations, the computational cost is too high (RS and GS take in the order of hours, while IC takes in

the order of minutes, on a i9 7th generation). From the results, it is recommended to work with IC to optimize the τ and m parameters, and, for the regression system parameters, to use RS. It is relevant to point out that IC is the fastest, while it provides a competitive prediction performance.

Table 1. Quality measurements for each time series made with each one of the proposed optimization strategies.

Series	MAPE-RS	MAPE-GS	MAPE-IC	R^2-RS	R^2-GS	R^2-IC	MSE-RS	MSE-GS	MSE-IC
COV	**3.6714**	8.9901	5.6877	**−3.4295**	−41.4618	−11.4179	**0.0005**	0.00511	0.0014
BIT	0.228	**0.1566**	0.2656	−3.016	**−1.026**	−4.383	0.0046	**0.0023**	0.0061
TEM	0.265	0.1973	**0.02007**	−2.537	**−0.3787**	−0.4936	0.0283	**0.011**	0.0119
S&P	0.525	**0.508**	0.5368	−5.642	**−5.336**	−5.851	0.162	**0.152**	0.165
TEL	0.1447	**0.1395**	0.14087	−0.1595	**−0.1061**	−0.1011	0.001895	0.001808	**0.001800**
CO2	0.0526	**0.0491**	0.0619	0.2866	**0.3706**	−0.0578	0.0027	0.0024	**0.004**

Figure 6a–c suggests that GS had better results, both in its mean and dispersion. However, if we look only at IC and RS we observe that when one of the two had a better mean, it would also have worse dispersion, which indicates that some series work very well with IC and others with RS, but, in general, it is a good idea to try both methods.



(**a**) Comparing with R^2. (**b**) Comparing with MSE. (**c**) Comparing with MAPE.

Figure 6. Boxplots comparing each procedure with different goodness-of-fit measures.

3.4. Future Work

It remains for future work to evaluate the procedures with additional quality measurements. Selecting the model with Mean Squared Logarithmic Error (MSLE) could improve the predictions. Including the regression system in the optimization could improve the prediction performance, by, for instance, using Naïve Bayes and K-Nearest Neighbor systems.

Author Contributions: R.H.-M.: Conceptualization, formal analysis, investigation writing original draft preparation; J.O.-B. (Jose Ortiz-Bejar): Conceptualization, supervision, writing and proofreading; J.O.-B. (Jesus Ortiz-Bejar): Review, formal analysis and proofreading. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found at the URLs mentioned in Section 3 of the manuscript.

Conflicts of Interest: The authors declare no conflict of 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.

References

1. Takens, F. Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence, Warwick*; Springer: Berlin/Heidelberg, Germany, 1980; pp. 366–381.
2. Cao, L. Practical method for determining the minimum embedding dimension of a scalar time series. *Phys. D Nonlinear Phenom.* **1997**, *110*, 43–50. [CrossRef]
3. Bergstra, J.; Bengio, Y. *Random Search for Hyper-Parameter Optimization*; Universite de Montreal: Montreal, QC, Canada, 2012. Available online: https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf (accessed on 10 May 2023).
4. Yu, S.; Pritchard, M.; Ma, P.L.; Singh, B.; Silva, S. Two-step hyperparameter optimization method: Accelerating hyperparameter search by using a fraction of a training dataset. *arXiv* **2023**, arXiv:2302.03845.
5. De Myttenaere, A.; Golden, B.; Le Gr, B.; Rossi, F. Mean Absolute Percentage Error for regression models. *Neurocomputing* **2016**, *192*, 38–48. [CrossRef]
6. Lehmann, E.L.; Casella, G. *Theory of Point Estimation*, 2nd ed.; Springer: New York, NY, USA, 1998; ISBN 978-0-387-98502-2.
7. Yin, P.; Fan, X. Estimating R^2 Shrinkage in Multiple Regression: A Comparison of Different Analytical Methods. *J. Exp. Educ.* **2001**, *69*, 203–224. [CrossRef]
8. Rivas-Perea, P.; Cota-Ruiz, J.; Chaparro, D.G.; Venzor, J.A.P.; Carreón, A.Q.; Rosiles, J.G. Support Vector Machines for Regression: A Succinct Review of Large-Scale and Linear Programming Formulations. *Int. J. Intell. Sci.* **2013**, *3*, 5–14. [CrossRef]