Reduction of the computational cost of tuning methodology of a simulator of a physical system.

Mariano Trigila[1[0000-0001-7886-7163]], Adriana Gaudiani[2[0000-0003-1651-0403]], Alvaro Wong[3[0000-0002-8394-9478]], Dolores Rexachs[3[0000-0001-5500-850X]] and Emilio Luque[3[0000-0002-2884-3232]]

[1] Facultad de Ingeniería y Ciencias Agrarias, Pontificia Universidad Católica Argentina, Ciudad Autónoma de Buenos Aires, Argentina
mariano_trigila@uca.edu.ar

[2] Instituto de Ciencias, Universidad Nacional de General Sarmiento, Buenos Aires, Argentina
agaudi@ungs.edu.ar

[3] Depto. de Arquitectura de Computadores y Sistemas Operativos, Universidad Autònoma de Barcelona, 08193 Bellaterra (Barcelona) España
alvaro.wong@uab.cat; dolores.rexachs@uab.es; emilio.Luque@uab.es

Abstract. We propose a methodology for calibrating a physical system simulator and whose computational model represents its events in time series. The methodology reduces the search space of the fit parameters by exploring a database that contains stored historical events and their corresponding simulator fit parameters. We carry out the symbolic representation of the time series using ordinal patterns to classify the series, which allows us to search and compare by similarity on the stored data of the series represented. This classification strategy allows us to speed up the parameter search process, reduce the computational cost of the adjustment process and consequently improve energy cost savings. The experiences showed a reduction in the computational cost of 29% compared with our tuning methodology proposed in previous research.

Keywords: Parametric simulation, Tuning methodology, Ordinal pattern, Time series classification, Data driven.

1 Introduction

Computer simulators are software components developed from the implementation of a model that represents a real system whose behavior is interesting to study or predict future events. As the simulator evolves over time, it tends to lose its calibration, since the parameters that define the system depend on physical magnitudes that define the real system and these change over time. A simulator is out of calibration when it produces output data that differs from the observed data, which is the data that is| meas-

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

ured in the physical system, and the difference between both data exceeds a predetermined limit of error [10, 11, 12].
The calibration of a simulator is the process by which the search for the set of parameters close to the optimum is carried out through parametric simulations. The search for the values of the adjustment parameter is carried out over a very large search space given the large number of different values and the number of parameters that can be associated with the model that represents the physical system. Consequently, the simulator will be executed with each set of parameters (simulation scenario), generating the simulated data set for each of the scenarios [11, 12]. As can be seen, the calibration process takes a large amount of time and requires a high cost of computing resources, which consequently results in a process with a high energy cost. In our research, we propose a novel methodology to reduce the computational cost of the adjustment and calibration process of a physical system simulator.
In this article we propose a methodology that offers us a more efficient way to find the parameters' values that best fit the observed data. We improve the efficiency of the search algorithms by reducing the search space of the adjusted set of parameters. Consequently, we can achieve considerable savings in computing resources. We use the strategy of storing the data defining a particular event in each moment in the past, when we had detected a lack of precision in the simulation. We also store the set of adjusted parameters found for tuning the simulation at that moment. In this way, we can use the stored events in the future when a disruption event occurs, and we can find those similar events that can potentially tune the simulator to the current event using a low computational cost method.
Our methodology fits into the data assimilation paradigm, since it combines observations of reality data and predictions of the states of the model parameters [8]. It also fits into the data driven paradigm in that measurement data is used to improve model accuracy and runtime, while the computational output produced by the model helps drive the measurement process itself [9]. We implement an ordinal pattern [1, 2, 3, 4] approach to efficiently find similar events stored in the past events database. In this work, we use a riverbed computational model that represents its events of interest through time series.
In Fig. 1 you can see the entities that interact with the “Fit Model” component: “ Real System” represents the real physical system and it is this which provides the observed data. “Model Simulator” represents the simulator, which is fed with the simulation scenarios and produces the output data, and “Past Events Store” represents the data store where the events that have occurred are recorded.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

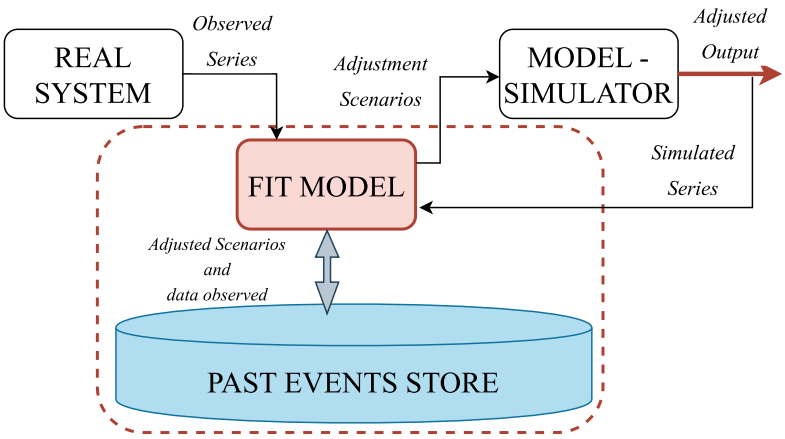

Fig. 1. Fit Model Contextual Diagram.

Carrying out the experiences and using our proposed methodology, we achieved savings in the use of computational resources of 29% compared to the successive steps methodology SSM [12]. Compared to the number of simulation runs we needed with the SSM methodology, we saved on average 1044 simulator runs, which leads directly to energy cost savings. It was also possible to speed up the search process for the adjustment parameters.
This article is organized as follows: In Section 2 we describe the simulation model and the simulation domain characterization, and in Section 3 we describe the proposed methodology, which includes, in subsections, the explanation of symbolic representation and the search temporal subsequences by similarity. In Section 4 we explain the experimental environment and the results of this experimentation. The conclusions are presented in Section 5.

2 The simulation model and the simulation domain characterization.

We use a simulation model developed by the National Water Institute of Argentina (INA), which calculates the translation of waves in rivers and canals [11]. The simulator output data to be considered in this work are the heights of the river at 15 measurement stations, that is to say, physical points of the riverbed where the height of the river is measured. The simulator fit parameters that have the greatest impact on the output data are the roughness coefficients, the “Manning coefficient” and the “Manning edge” [10]. The simulation model discretizes the riverbed in 76 sections, and for each section, supplying the Manning coefficient values will be required. For a complete run of the simulator, a set of adjustment parameters must be provided, which is made up of all the Manning values for both edge and plain for each of the 76 sections. For each section, riverbed and plain subsections are considered at both ends of the section, and it is necessary to have parameter values in each of these subsections.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

The simulator provides the simulated time series, which are the outputs it produces in each complete run, for each station, for the configured scenario. This series contains the daily height of the river for the simulated period.
There is also a time series set of observed data, which are the real records taken daily of the height of the river in 15 monitoring stations for a period of 11 years. With this data set, the output of the simulator will be validated at each station.

3 Proposed methodology.

We propose a calibration methodology that reduces the search space of the fit parameters by using the exploration of a database that contains stored historical events of the real system, the historical events of the simulated system and the history of the best fit parameters of the simulator.
The methodology is centered on a Fit Model, which for its analysis and description is divided into the following functional modules (see Fig. 2): 1) Check Simulation - Reality (CSR): This checks that the time series output from the simulator has an acceptable similarity with respect to the time series of the real data observed. 2) Simulations Shoot (SS): This triggers the simulations with new simulator input parameters to obtain the best-fit outputs. 3) Set / Get Past Events (SGPE): This performs the reading, writing of events in the database and the characterization of events.

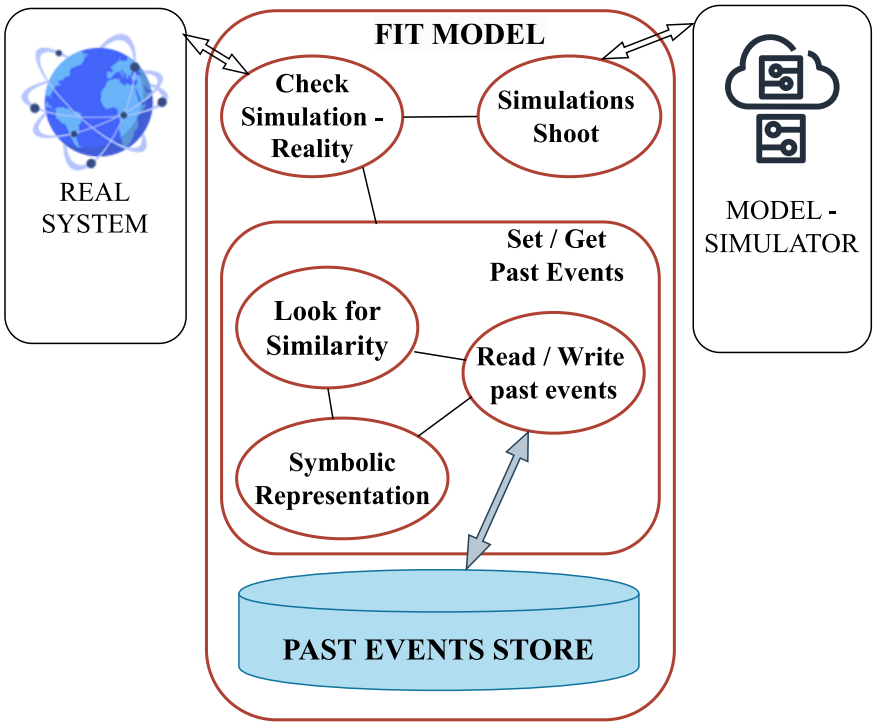

Fig. 2. Fit Model Detail - Functional Modules.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

In the following subsections, we carry out a detailed explanation of each of the functional modules including the innovative concepts that are used in the methodology.

3.1 Check Simulation - Reality

The CSR functional module has the purpose of measuring and controlling the difference between the output values produced by the simulator for a simulation scenario and the actual measured values observed at station k.
We define S as the set of the complete scenarios of the simulator, S_k as the set of scenarios for station k and $\widehat{s_K} \subseteq S_k \subseteq S$ as the best fit scenario for measuring station k.
Using a divergence index implemented with the mean square error estimator (RMSE), we determine the best scenario $\widehat{s_K}$ by comparing the simulated data set SD with the observed data set OD.

$$DI_{k,d}^{t} = RSME_{k,d}^{t} = \sqrt[2]{\frac{\sum_{i=0}^{i=d-1} \left(H_k^{OD,t} - H_k^{SD,t}\right)_i^2}{d}} \quad (1)$$

The $DI_{k,d}^{t}$ index is the RMSE error of the series of simulated river heights $H_k^{SD,t}$ in relation to the observed series of river heights $H_k^{OD,t}$, at a measuring station k, for a value in time t, for a subsequence of the series of size d.
We define acceptable difference AD as a reference value, provided by the engineers who modeled the river and use the simulator selected for this work, and its decimal value can be between $0 \leq AD \leq 1$. AD as will be the reference value to determine if $DI_{k,d}^{t}$ is acceptable or not. When AD is close to 0, it will indicate that the scenario is a good fit, otherwise it will indicate that it is a bad scenario.
If the $DI_{k,d}^{t}$ index is greater than the reference value AD, then a difference has been found between the observed and simulated series. This indicates that an adjustment of the simulator must be carried out by requesting Simulations Shoot to execute the simulator for a new set of parameters.
If the $DI_{k,d}^{t}$ index is less than or equal to the reference value AD, then the difference between the observed and simulated series is acceptable, therefore a scenario $\widehat{s_K}$ has been found. In this case, Set / Get Past Events is requested to store $\widehat{s_K}$, the observed height subsequence $H_{k,d}^{OD,t}$ of size d, the simulated subsequence $H_{k,d}^{SD,t}$ of size d and the time t.

3.2 Simulations Shoot

The Simulations Shoot functional module has the purpose of requesting its execution to the simulator by supplying the values of the scenario set S that is desired to be tested in the simulation process. Scenario set S is composed of the adjustment parameters $Sc_m \subseteq S$, one for each section. m is the section number and its value is between $1 \leq m \leq 76$. The scenarios Sc_m that are supplied to the simulator contain the mp_m that is Manning value of plain and mc_m that represents the Manning value of the

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

channel. We define a scenario Sc_m for a section m, subdivided into three subsections, like a 3-tupla (2).

$$Sc_m = (mp_m, mc_m, mp_m) \quad (2)$$

Simulator execution is triggered with the set of adjustment parameters S provided by the CSR module after requesting its execution.
The simulation response, which is the simulated data series, is sent as a response to the CSR module for evaluation.

3.3 Set / Get Past Events

Functional module Set / Get Past Events performs the management of writing, reading the database and it handles characterization and symbolic representation of the subsequences of time series that are stored. Receives from RSC requests to retrieve past events from the database based on a current event or to update the database with new events that are not yet registered.
For better analysis and description, we divide it into the following functional submodules: Symbolic representation (SR), Look for similarity (LFS) y Read / Write past events (RWPE). In the following subsections we explain this in detail.

3.3.1 Symbolic representation

The SR module has the purpose of mapping the subsequences of time series of the observed heights $H_{k,d}^{OD,t}$ or simulated heights $H_{k,d}^{OS,t}$ to symbolic representations. It also performs the unmapping of symbolic representations to subsequences of time series Fig. 3.
The methods of symbolic representations of time series discretize the series and transform it into a sequence of symbols making, in our case, the search for series of similar events from the past, stored in the database, more efficient, thus accelerating the process of search by comparison. We will use the symbolic representation methodology of Bandt and Pompe (2002) [4], hereinafter BP, to represent a subsequence of values of a time series into an array of values belonging to an alphabet Θ which will represent the characterization of the subsequence, where φ is a symbol, $\varphi \in \Theta$, and $0 \leq \varphi \leq d$, d $(d \in \mathbb{N})$, d is the number of values in the subsequence is the number of values that make up each subsequence and this is called the embedding dimension, which is usually set between $3 \leq d \leq 7$.
In the BP approach, the concept of ordinal pattern (OP) (also called permutation pattern) is defined [1, 2, 3, 4]. Given a time series X_t, It is divided into subsequences of consecutive values $(x_t, x_{t+\tau}, \ldots, x_{t+\tau d})$; d $(d \in \mathbb{N})$; y τ $(\tau \in \mathbb{N})$ the time between consecutive points and this is called the embedding delay, usually set to 1 [1]. An ordinal pattern π_t is a sequence of symbols φ of dimension d which is associated with each of the time series subsequences. The OP is a sequence of symbols $\varphi \in \mathbb{N}$, where each symbol inside the pattern indicates the permutations that must be applied to the element of the temporal subsequence in order obtain a set of d dimension, with the increasing order of the values of the temporal subsequence. To

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

broaden any knowledge, we recommend the bibliography where the subject is explained in detail [1, 2].

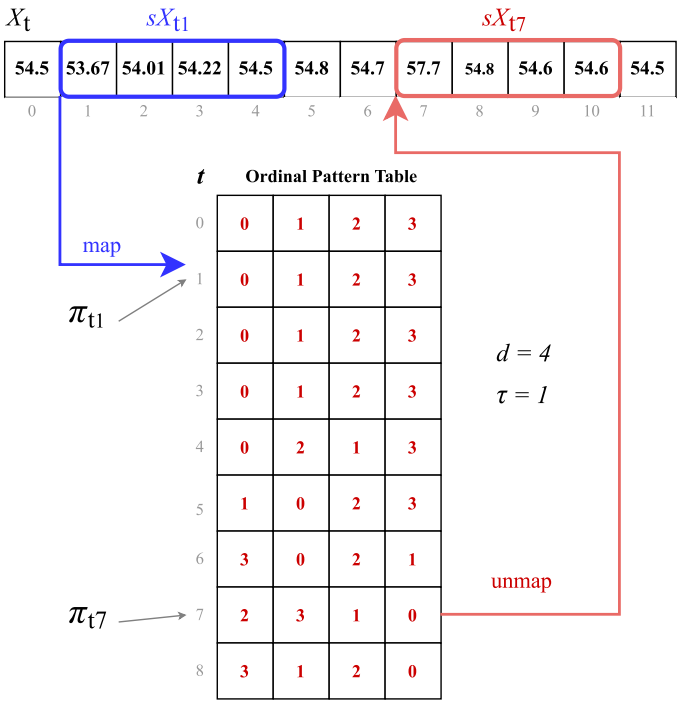

Fig. 3. Map / Unmap Ordinal Pattern.

We then use, in our methodology, the BP approach through a mapping function M, which transforms a time series subsequence $sX_{t\alpha}$ into an ordinal pattern $\pi_{t\alpha}$, for time series X_t with ($sX_{t\alpha} \subseteq X_t$), for a moment in time $t\alpha$, for an interval of time between consecutive points τ ($\tau \in \mathbb{N}$) , for a number of values d ($d \in \mathbb{N}$) of values that make up the subsequence [5, 6]. All the above can be summarized in (3).

$$M\,(sX_{t\alpha}, t\alpha, \tau\ , d) \mapsto \pi_{t\alpha} \tag{3}$$

To solve the mapping M, we developed a software program that performs the transformation using "ordpy" [3], which is a software package developed in python that implements permutation entropy and several of the main methods related to the Bandt and Pompe framework.

To get a subsequence or a list of subsequences that have a certain ordinal pattern $\pi_{t\alpha}$, we use the unmapping function U. To solve the function unmapping U, we develop a software program that searches the database of stored past events and returns a list L of the subsequences which have the same ordinal pattern $\pi_{t\alpha}$. That is, given a$\pi_{t\alpha}$ we get a list L where $sX_{t\alpha} \subseteq L$ (4).

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

$$U\left(\pi_{t\alpha}\right) \mapsto L \tag{4}$$

3.3.2 Look for similarity.

The LFS functional module has the purpose of searching for subsequences of similar time series.
From the mapping of each subsequence, we can identify its associated ordinal pattern and classify the temporal sequences [6, 7]. Therefore, we can relate a set of temporary subsequences to an OP that will be the class to which the subsequence belongs. Thus, we can indirectly compare two time subsequences by comparing their OP. Two temporal subsequences with the same OP belong to the same class, that is, it indicates a certain similarity between the subsequences, but it does not necessarily indicate that they are the same. With a similarity index implemented with the root mean square error estimator (RMSE), the similarity between two temporal subsequences that have the same OP is quantified.
Let us see how a similarity search develops:
Given the subsequence $sX_{t\alpha}$, from eq. (3), we obtain the ordinal pattern OP $\pi_{t\alpha}$. Then we access the database of past events and look for the temporary subsequences that have the same OP. If we find any $\pi_{t\beta}$ equal to $\pi_{t\alpha}$, then we generate a list of temporal subsequences that are related to the pattern $\pi_{t\beta}$.
Then, with each of the subsequences found in list L we apply the similarity index $SI_{\alpha,\beta}$ to determine what the subsequence $sX_{t\beta}$ most similar to subsequence $sX_{t\alpha}$ is. That is to say, if the index $SI_{\alpha,\beta}$ is closest to the value 1 (5).

$$SI_{\alpha,\beta} = 1 - \sqrt[2]{\frac{\sum_{i=0}^{i=d} (x_\alpha - x_\beta)_i^2}{d}} \tag{5}$$

The similarity index can vary between $0 \leq SI_{\alpha,\beta} \leq 1$, the closer it is to 1, the more similar the subsequences will be.
As previously explained, the search method generates list L, which is obtained from the data previously stored in PES (Past Events Store), so that $L \subseteq PES$. The PES module is described below.

3.3.3 Read / Write past events.

The functional module Read / Write past events is for the purpose of storing and obtaining the data in the database PES (Past Events Store). Past events are recorded in the PES, which can then be used in similar current situations. They are stored in a table whose record has the following structure:
t: The time the event occurs.
sX_t: The subsequence of the time series X_t.
π_t : The ordinal pattern related to the subsequence sX_t.
Sc: The full set of simulator best fit parameters associated with the moment t .

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

4 Experience Environment and Experimental Results

The experiments that we present in this article were carried out focusing on a single measurement station. The station on which the experiments were carried out has the real name of "Itabaite" and its mnemonic in the simulator is "ITAE" and represents a city located in the domain of the river OD_K. So, for the station $k = "ITAE"$ we have a series of observed data SD_K with a quantity of 5533 records of daily measurements of water height H_k^{OD} between the dates $09/04/2010 \leq t \leq 09/04/2010$. The OD_K series has a maximum height of 58.3, a minimum height of 52.3 and a standard deviation of 0.93.

To carry out the experiments, the observed data was partitioned OD_K at 70% - 30% (see Fig. 4). 70% corresponds to the reference series, which are the first records of the observed data series that goes from $t_1 = 01/08/1994$ to $t_{70} = 31/12/2005$.

These observed data $OD_k^{t_1, t_{70}}$ are stored in the PES database, together with the corresponding simulated data $SD_k^{t_1, t_{70}}$, with their associated ordinal patterns $\pi_k^{t_1, t_{70}}$ (see Fig. 5, the histogram of π) and their corresponding best-fit parameters $Sc_m^{t_1, t_{70}}$. Therefore, with these data we built the database, PES, in a startup/initial state, to search for similarities.

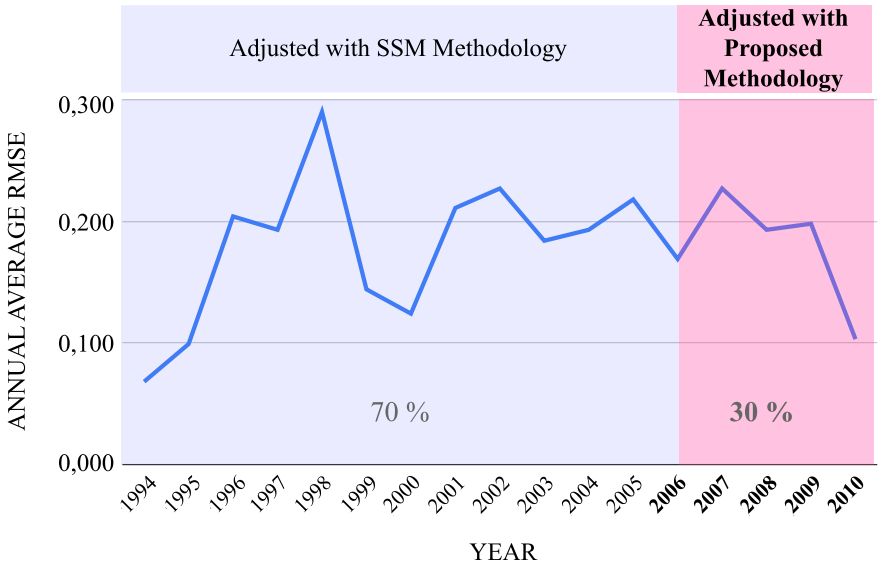

Fig. 4. Annualized mean square error between the simulated series with adjusted parameters and the observed series.

The process of carrying out the experiment consisted of adjusting the simulator every 5 days, the number of days is 1500 for the period of t_{71} to t_{100}, therefore 300 adjustments should be made. If we consider that the SSM methodology has an av-

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

erage execution of 12 calls to the simulator for each adjustment, then SSM on average would call the simulator 3600 times to perform the 1500-day adjustment.

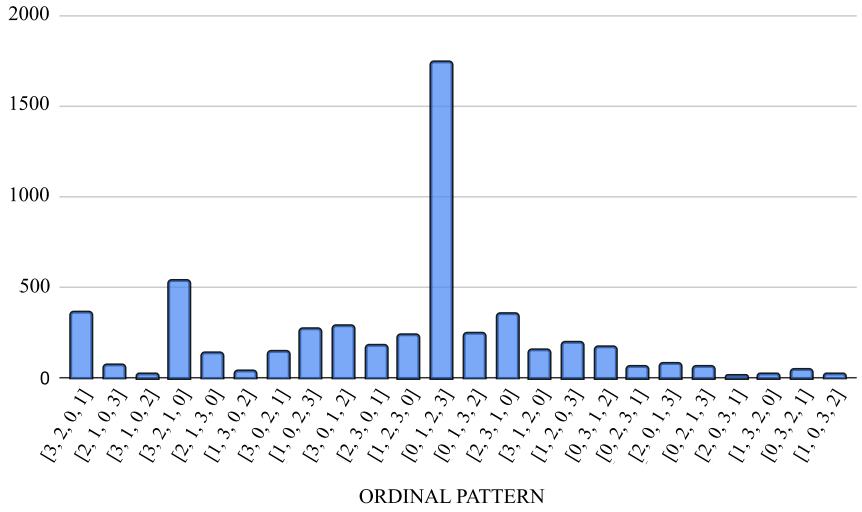

Fig. 5. Histogram of ordinal patterns.

To determine the efficiency of our methodology, we measured the number of times the correct fit parameters were obtained for a current event by accessing the PES database and carrying out a similarity search. If the adjustment parameters were not found in PES, then we proceeded to obtain it using the SSM methodology, making parametric simulations. In Fig. 6 the goodness of the proposed methodology can be observed.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

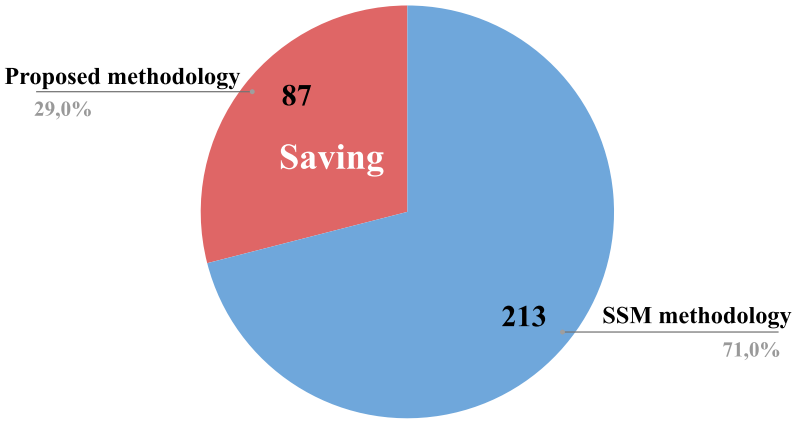

Fig. 6. Savings obtained with the proposed methodology.

With the methodology that we propose, we found the adjustment parameters in PES in 87 (29%) adjustment processes, while in the remaining 213 (71%) adjustment processes, we obtained the adjustment parameters with the SSM methodology. This indicates a saving of computing resources of 29%.

5 Conclusions and future works

In this article, we have proposed a new simulator tuning methodology that saves the use of computational resources by reducing the search space of the adjustment parameters set.

We understand that the strategy of using ordinal patterns to classify the time series in a database greatly accelerated the search process for the adjustment parameter registered in the database through similarity comparison. The experimental results of this work are promising and validate our proposal. We obtained a 29% improvement compared to the methodology used in our previous work.

It is good to indicate that the process of loading the events in the database is simple and of low computational cost, since the information is stored and expanded as the simulator evolves over time in a dynamic way.

It is noteworthy that we observe that the proposed methodology is scalable in terms of the potential ability to process a larger number of events. We attribute this quality to the classification approach of the time sub-series using the symbolic representation.

There are several directions for future research, some of which could be: experimenting with much more stored event information, investigating the relationships between ordinal patterns and the probability of occurrence of sequence patterns of ordinal patterns over time, and we are working to tune more domain points in the same system calibration process. Also, a future direction in which we are focused is to extrapolate our methodology to other simulation models of physical systems that need to find

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49

behavior patterns and that also need to tune their simulator automatically. The level of abstraction achieved with our methodology allows us to infer that extrapolations to other physical simulators are highly feasible, thus expanding its field of application. The motivation is broad and calls us to continue expanding our knowledge about this methodology.

Acknowledgments

This research has been supported by the Agencia Estatal de Investigacion (AEI), Spain and the Fondo Europeo de Desarrollo Regional (FEDER) UE, under contract PID2020-112496GB-I00 and partially funded by the Fundacion Escuelas Universitarias Gimbernat (EUG).

References

1. Zanin, M., Olivares, F.: Ordinal patterns-based methodologies for distinguishing chaos from noise in discrete time series. Communications Physics, 4(1), 190 (2021).
2. Bariviera, A. F., Guercio, M. B., Martinez, L. B., Rosso, O. A.: Libor at crossroads: Stochastic switching detection using information theory quantifiers. Chaos, Solitons & Fractals, 88, 172-182 (2016).
3. Pessa, A. A., Ribeiro, H. V.: ordpy: A Python package for data analysis with permutation entropy and ordinal network methods. Chaos: An Interdisciplinary Journal of Nonlinear Science, 31(6), 063110 (2021).
4. Bandt, C., and Pompe, B.: Permutation entropy: a natural complexity measure for time series. Physical review letters, 88(17), 174102 (2002).
5. Mohr, M., Wilhelm, F., Hartwig, M., Möller, R., Keller, K.: New approaches in ordinal pattern representations for multivariate time series. In The Thirty-Third International Flairs Conference (pp. 124-129) (2020).
6. Sinn, M., Keller, K., Chen, B.: Segmentation and classification of time series using ordinal pattern distributions. Eur. Phys. J. Spec. Top. 222, 587–598 (2013).
7. Cuesta-Frau, D.: Using the information provided by forbidden ordinal patterns in permutation entropy to reinforce time series discrimination capabilities. Entropy, 22(5), 494 (2020)
8. Reich, S.: Data assimilation: the Schrödinger perspective. Acta Numerica, 28, 635-711 (2019).
9. Fujimoto, R., Barjis, J., Blasch, E., Cai, W., Jin, D., Lee, S., Son, Y. J.: Dynamic data driven application systems: research challenges and opportunities. Winter Simulation Conference (WSC) (pp. 664-678). IEEE (2018).
10. Berends, K. D., Warmink, J. J., Hulscher, S. J. M. H.: Efficient uncertainty quantification for impact analysis of human interventions in rivers. Environmental modelling & software, 107, 50-58 (2018).
11. Gaudiani, A., Wong, A., Luque, E., Rexachs, D. A computational methodology applied to optimize the performance of a river model under uncertainty conditions. J Supercomputing 79, 4737–4759, https://doi.org/10.1007/s11227-022-04816-6 (2023).
12. Trigila, M., Gaudiani, A., Luque, E.: Agile Tuning Method in Successive Steps for a River Flow Simulator. In:, et al. Computational Science – ICCS 2018. ICCS 2018. Lecture Notes in Computer Science(), vol 10862. Springer, Cham, https://doi.org/10.1007/978-3-319-93713-7_60 (2018).

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_49