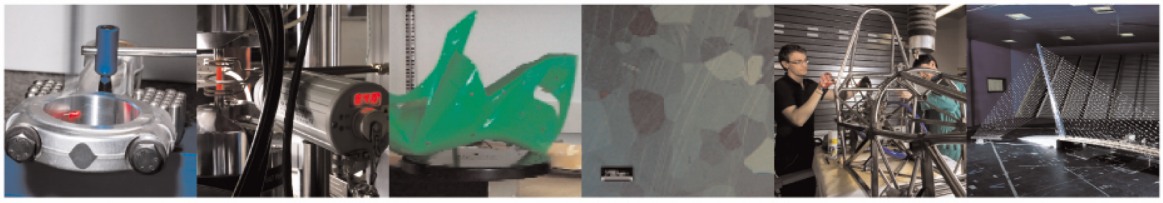
POLITECNICO
MILANO 1863

DIPARTIMENTO DI MECCANICA

mecc



Online Validation of Digital Twins for Manufacturing Systems

Giovanni Lugaresi, Sofia Gangemi, Giulia Gazzoni, Andrea Matta

This is a post-peer-review, pre-copyedit version of an article published in *Computers in Industry*. The final authenticated version is available online at:
http://dx.doi.org/10.1016/j.compind.2023.103942

DIPARTIMENTO DI MECCANICA ■ POLITECNICO DI MILANO
via G. La Masa, 1 ■ 20156 Milano ■ EMAIL (PEC): pecmecc@cert.polimi.it
http://www.mecc.polimi.it
Rev. 0

Published Journal Article available at: https://dx.doi.org/10.1016/j.compind.2023.103942

Online Validation of Digital Twins for Manufacturing Systems

Giovanni Lugaresi[1*], Sofia Gangemi[2], Giulia Gazzoni[2], Andrea Matta[2]

[1]Laboratoire de Génie Industriel, CentraleSupélec – 3 rue Joliot Curie, 91190 Gif-sur-Yvette, FRANCE

[2]Department of Mechanical Engineering, Politecnico di Milano – Via La Masa 1, 20156 Milano, ITALY

[*] Corresponding author: giovanni.lugaresi@centralesupelec.fr

Abstract. Digital twins are considered among the most important instruments to optimize production and support decision making. To benefit from their functionalities, it is essential to guarantee a correct alignment between the physical system and the associated digital model, as well as to assess the validity of the digital model online. This operation should be conducted rapidly and with a limited available data set, especially in highly dynamic contexts. Further, the whole time behaviour of a system may be of interest rather than the sole average performance. Traditional validation techniques are limited because of the restrictive assumptions and the need of large amounts of data. This work defines the problem of checking the validity of digital twins for production planning and control while the physical system is operating. A method describing the data and the types of validation is proposed including a set of algorithms to be used at different levels of detail. The congruence between the physical system and the corresponding digital model is measured by treating data as sequences, and measuring their similarity level with digitally-produced data exploiting a proper comparison technique. The numerical experiments show the potential of the proposed approach and its applicability in realistic settings.

Keywords: digital twin; online validation; smart manufacturing; discrete event simulation.

1 Introduction

Recently, manufacturing systems have been characterized by disruptive innovations driven by the digitization efforts of the Industry 4.0 revolution. The change is enabled by the development of new technologies such as Internet of Things, big data analytics, cloud computing and artificial intelligence. These technologies allow to interconnect elements within the production facility, to collect and analyse large amounts of data as well as to reproduce and predict the behaviour of a system. The Digital Twin (DT) is a growing technology capable to improve the performance of a manufacturing system and support decision making. With reference to manufacturing systems, the DT is defined as: *"a virtual representation of a production system that is able to run on different simulation disciplines that is characterized by the synchronization between the virtual and the real system"* (Negri et al., 2017).

Within the scope of production planning and control, the virtual entity can be represented by a Discrete Event Simulation (DES) model. The DT is a dynamic concept and it can be seen as the evolution

of simulation, integrating modelling and optimization capability. Simulation has been used since many years for the design, validation, and testing of manufacturing systems as well as for improving decision making. These applications traditionally take place in the early life cycle stages of a system. However, simulation can be applied also in the operational stages for offline decision-making, proactive reasoning, and reaction definition (Monostori et al., 2016). The DT is able to collect and use real-time data from the field. Differently from traditional simulation approaches, a bi-directional exchange of information and the synchronisation between the physical and the virtual components are essential features of a DT. In addition, the DT capability of providing feedback actions to the physical system allows to improve promptness and efficiency of decision making. Because of the physical-digital fusion, the DT is capable of optimizing a system during the entire lifecycle allowing short-term decision-making (Liu et al., 2021). Indeed, short-term decision making requires a rapid detection of disruptions, and any delay in the implementation of corrective actions may cause a negative impact on production, reflecting on economic losses for the company. To prevent this scenario, it is essential to have a model coherently representing a real system along its whole life-cycle, adapting to possible changes. As a consequence, to benefit from the DT functionalities, it is essential to assess online the validity of the virtual component.

Existing digital twin applications do not discuss how the validity of the digital model can be assessed online. Also, significant contributions on the digital model validation for short-term applications are scarse. Traditional validation techniques rely on statistical methods which require large data sets and multiple independent replications due to the stochastic nature of the physical system (Balci and Sargent, 1982). These requirements affect the promptness with which disruptive events are detected: the most recent behaviour is averaged with historical data and potential deviations might not be detected. Moreover, since only one set of data can be acquired from the field, batching procedures have to be applied to avoid auto-correlation effects and satisfy the assumption of independence. Batching leads to data averaging and the performance trend might be lost. As a consequence, changes that do not impact on the average behaviour of the system might not be identified and corrected. Given these characteristics, it is possible to conclude that traditional statistical approaches are not suitable for online applications, especially for manufacturing systems that do not produce large data sets.

This work represents the first study that defines the problem of online validation of digital twins by discussing the technical issues emerging when the sets of data generated by the physical and digital system are compared for validation purposes. This work also aims to overcome the aforementioned limitations by providing a new methodology to validate online DES models representing a discrete event system. A special focus is given to manufacturing systems, that embody one of the main DT application fields. The method deals with a limited data set from a single simulation experiment, and it is capable to analyse the evolution in time of the system behaviour. The digital model validity is assessed measuring the similarity between real system data and digital model data. As example, Figure 1 shows the online collection of inter-departure times from a manual assembly station. These data are typically saved in the

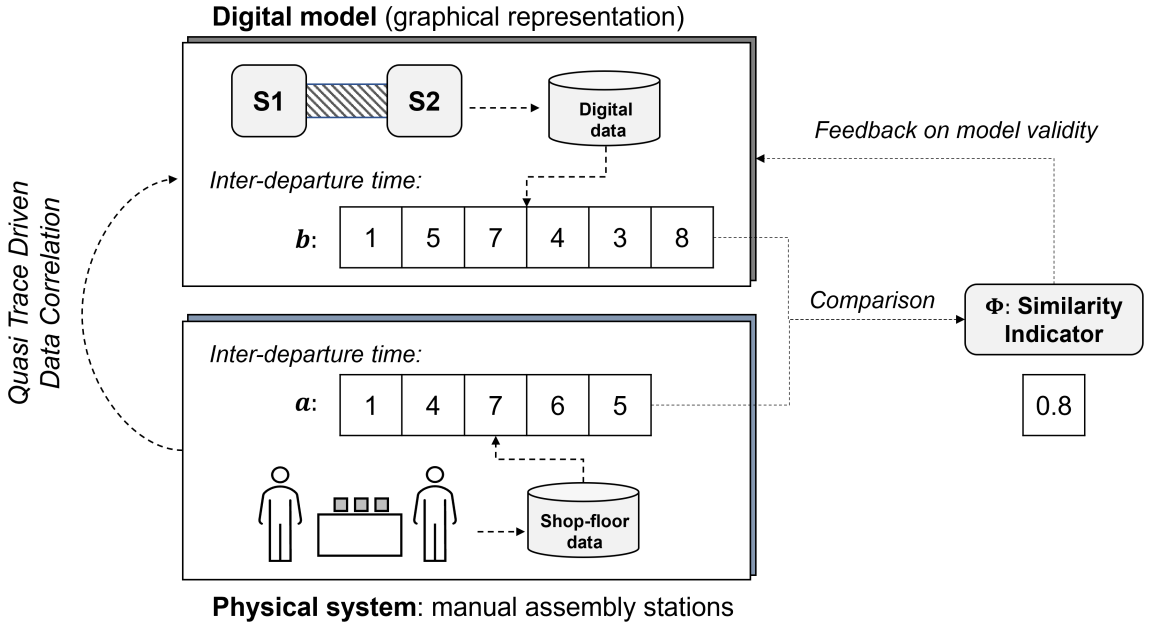


Fig. 1: Example - Comparison of data sequences generated by a real system composed by two manual assembly stations and its corresponding digital model. The inter-departure times are collected from the physical system (**a**) and measured also in the digital space (**b**). A similarity indicator is used to assess the validity of the digital model.

form of time-series and it is possible to treat them as sequences. Hence, the proposed method measures the similarity level with digitally produced data by means of a proper sequence comparison technique. The similarity level is then used to determine if the digital model is valid.

The rest of the paper is organized as follows. The related works and the research gap on online validation are discussed in section 2. The developed methodology to assess online the validity of a DES model is explained in section 3. Section 4 presents the numerical experiments to test the applicability of the proposed methodology. Finally, conclusions are drawn in section 6.

2 Related Literature

This section summarizes significant contributions in the literature that study the consistency between manufacturing systems and their digital models. Table 1 summarizes the main contributions and their characteristics. The focus is on methods that address the validation of the digital model, either offline or online. Finally, the research gap is highlighted.

2.1 Offline Validation of Digital Twins

Assessing the correctness of the digital model is a fundamental step of its development phase. Indeed, digital models are used to support decision making in the real world and it is important to guarantee the reliability of the obtained results. The process of valuation of the model's correctness and capability to reproduce a system's behaviour is typically called *verification and validation* (V&V). According to Sargent (Sargent, 2010), V&V of a simplified model development process includes four parts: (1) conceptual model validation, (2) computerized model verification, (3) operational validation, and (4) data validity. This work focuses on the operational validation procedure defined as *"determining that the model's output behaviour has a satisfactory range of accuracy for the model's intended purpose over the domain of the model's intended applicability"*, that in the remainder of this paper is simply referred to as validation. It is important to underline that the model validity is always checked under specific experimental conditions and for specific purposes. Moreover, results of the validation process depend on the level of accuracy defined by the user (Sargent, 2010).

Balci (Balci, 1994) proposed a taxonomy on more than seventy-five V&V testing techniques composed of four main categories, each comprising a wide set of techniques, characterized by an increasing level of formality and complexity: (1) *informal techniques* based on human reasoning and subjectivity; (2) *static techniques* established on the model design and source code (i.e., no model execution is needed); (3) *dynamic techniques* dependent on the model execution and comparison of the output behaviour; (4) *formal techniques* rely on a mathematical proof of correctness. As stated by Sargent (Sargent, 2013), model validation consists in comparing the output behaviour of the system with the one of the simulation model, this approach is in accordance with the techniques identified in the dynamic category of the taxonomy proposed by Balci (Balci, 1998). Results of the comparison determine the level of accuracy of the digital model.

Balci (Balci, 1998) proposed that V&V should be conducted continuously throughout the entire lifecycle of a validation study in order to identify and rectify deficiencies. The following errors must be prevented: (1) type I error, which is committed when the simulation results are rejected when in fact they are sufficiently credible; (2) type II, which represents a situation in which invalid simulation results are accepted as valid; (3) type III error is when the wrong problem has been solved hence when the problem formulation does not contain the actual issue. When simulation is used to support decision-making, fatal consequences may arise from type II and III errors. For this reason, early detection and correction of errors is extremely important.

A model is always an abstraction of a system and a perfect representation cannot be achieved, hence validity is checked with respect to the application objectives. The outcome of V&V process should be expressed as a percentage value of credibility instead of a binary variable assessing the correctness in absolute terms. Due to time and cost constraints, it is not possible to test a model under all input

conditions, hence the credibility required depends on the predefined objectives (Banks, 1998). In addition, Balci (Banks, 1998) distinguishes between two types of simulation to obtain data for validation purposes. In a Trace-Driven Simulation (TDS) the model is run using as input the same trace data collected from the system. In a self-driven simulation, the model is run by sampling input data (i.e., technically random variates) from probabilistic models. Usually, input data are modelled by fitting probability distributions to the observed data.

Similarities with model validation can be found in conformance checking approaches. Conformance checking is a technique belonging to the wider spectrum of process mining (van der Aalst, 2016). Process mining comprises a variety of techniques that, based on event logs, allow the analysis of business processes. By means of data coming from the event logs, process mining techniques enable to discover, monitor, and improve processes. Within process mining there are three classes of techniques: discovery, conformance checking and enhancement. Conformance checking deals with comparing an available event log of the process with the discovered process model to assess its correctness (Naderifar et al., 2019). Camargo et al. (2020) propose a technique for automated discovery of a business process simulation model using process mining. To measure the accuracy of the discovered model, a variant of the Damerau-Levenstein distance (Bard, 2006) is used to compare traces[1] in the event log. In addition, traces are paired using the Hungarian algorithm (Kuhn, 1955) to minimize their distance. The similarity between event logs is then assessed as the overall distance between paired traces. The concept behind this technique is similar to validation, nevertheless in the conformance checking field the model accuracy is assessed only once after model discovery, and not during operational phases.

According to Sargent (2013), the most common approaches used for model validation are graphical comparison, confidence intervals, and hypothesis tests. These methods encounter their limitations when put to work in a digital twin framework, where decisions must be taken within short time frames, hence the validity of a model has to be checked quickly and with limited need of human intervention. The limitations are evident. Graphical comparisons are characterized by subjective decisions which lead to difficulties in evaluating results. Also, a graphical comparison activity implies that the information flows cannot be fully automatized. On the other hand, confidence intervals and hypothesis tests are based on statistical assumptions that are not always satisfied, for example the availability of enough data points or specific requirements on the statistical distribution (e.g., normality of data). In addition, the comparisons are typically based on the average behaviour of the system. Hence, traditional methodologies are not suitable for the purpose of this work.

[1] A trace is defined as a sequence of activities performed by an element in the system (van der Aalst, 2016).

2.2 Online Validation of Digital Twins

Recently, new studies about online validation have been published: the most relevant are reported in the following paragraphs. Marquardt et al. (2021) introduced the importance of online validation when dealing with a DT. In order to provide both strategic and operational decision support, it is fundamental for the digital model to be always updated with respect to the real system. In particular, the paper highlights the effects caused by a model not coherently representing reality, considering both inconsistencies of input parameters and model assumptions. Although explaining the importance of having a model accurately representing the real world, the paper lacks a feasible online validation approach. An attempt to tackle this issue has been proposed by Overbeck et al. (2021), that uses statistical indicators to compare the performances of the physical and digital system. This paper shows the growing interest in the topic but still focuses on the average behaviour of the two systems.

Lugaresi et al. (2019) presented a new methodology to perform online validation based on harmonic analysis. Compared to statistical techniques, this method is capable to achieve reliable results even when applied to a relatively small data set. The main idea behind this procedure is to consider Key Performance Indicators (KPIs) trends as signals. Using the power spectral density (Howard, 2004), it is possible to detect potential deviations between the model and the real system. Namely, the difference between the power spectral densities of the two outputs is used to define a measure called spectral indicator, that can be used to assess the level of consistency of the model. A methodology to discriminate between two systems based on the Fourier transform of the output trajectory is proposed in (Morgan and Barton, 2022). The weighted average of the Fourier coefficient magnitudes is used to detect differences between the two systems under analysis. What differentiates the latter two approaches from other methodologies is the focus on the whole trend of the performance measure rather than on the average value. Nevertheless, both approaches analyse the performance measure in the frequency domain, making them less likely to be applied in the industrial practice.

Recent approaches considered the idea of comparing data traces (or sequences) for validating in real-time digital models. Leroy et al. (2018) introduced a collection of operators for comparing execution traces of state machines models. The operators consist of dynamic trace filtering, trace comparison with differential computation and visualization, and extraction of graph-based views to analyze cycles. The authors use the Levenshtein distance to measure the traces similarity. Gong et al. (2022) introduced a method for human-computer interaction that allows remote manipulation of life-size humanoid robots. The authors included a novel set of metrics for evaluating the similarity between the motion of a robotic arm and the one of a human in real-time. Muñoz et al. (2022) presented a method for aligning the traces of digital twins in the form of data sequences representing their progressive states. For this scope, the authors propose a modified version of the Needleman-Wunsch algorithm. Despite being promising approaches, these techniques do not allow for the distinguished validation of logic and inputs of the digital

models. Also, the comparison methods do not allow to include a random number generator in the digital model, hence they cannot be used while the simulation is running. This is probably the reason why none of the aforementioned works has been applied to manufacturing systems. An algorithm based on sequence comparison techniques for online input validation is proposed in (Lugaresi et al., 2022) and successfully applied on a simple single server system.

To the best of the authors knowledge, few solutions to the online validation problem for manufacturing systems are present in literature. This work focuses on data that represent material flow events within a production facilities, for instance, the entrance of a part in a machine, the inter-departure times from a station, the beginning of a production operation. Compared to the existing literature, the work that shares most of these objectives is (Muñoz et al., 2022), which also proposes sequence comparison techniques to assess the alignment between a system and its digital counterpart. However, the following main differences can be identified: (1) the proposed approach assumes to use two data sequences taken with the same time steps, and to be able to compare each sequence point-by-point. This is hard to guarantee in complex manufacturing systems, where even the settings of the sensors could determine the misalignment of data sequences (Schmetz et al., 2022); (2) the validation approach does not consider digital models with a random number generation; hence it can directly compare the data sequences; (3) given the aforementioned points, the proposed approach is applied to other domains such as robot trajectory planning, and does not consider manufacturing systems applications. This paper aims at overcoming the aforementioned limitations by developing a methodology that can be applied to validate a digital twin by exploiting real-time data streams from a manufacturing shop floor while operating. Specifically, the approach (1) can be applied to any sort of input data from the real system including a possible misalignment of the traces, (2) allows to still use a random number generator within the digital model while validating it Lugaresi et al. (2019); (3) can be applied within manufacturing system applications.

The main contributions of this paper are twofold. First, the paper describes the problem of online validation of digital twins within the scope of production planning and control. It is worth to notice that the same problem can be present in several other contexts in which the physical system can be modeled as a discrete event system, such as logistics and transportation, warehouse management, and supply chain management. The second contribution is the proposal of a general methodology for online validation of digital twins that can be deployed at different levels of detail and in an automated way while the physical system is operating. The proper technique to detect similarities and deviations of the digital model from the physical system may depend on the specific application. In this work, three alternative techniques have been identified from the literature on sequence comparison problems, and adapted for the online validation problem.

Table 1: Summary of main validation techniques in the literature and their limitations.

References	**Methods**	**Limitations**
Sargent (2013); Balci (1994); Kleijnen (1998)	Graphical Comparison	Subjective decisions; difficulties in evaluating results: information flows cannot be fully automatized
	Confidence Intervals	Based on strong assumptions; need for the availability of lots of data
	Hypothesis Tests	Typically based on the average behaviour of the system
Khan et al. (2018)	Model specification checking	Only for automata and petri net models, focus on model verification
Schruben and Cogliano (1981); Lugaresi et al. (2019)	Signal-processing techniques	Need to switch to the frequency domain
Camargo et al. (2020)	Conformance checking	No approaches applied during the operational phases
Leroy et al. (2018); Muñoz et al. (2022); Gong et al. (2022)	Trace Analysis	Appropriate for deterministic applications (e.g., trajectory planning); currently not applied in manufacturing systems

3 Methodology

This work proposes a methodology to perform online validation of digital twins based on DES models. The starting point of this study is to understand when a model can be defined a good representation of the physical system and what is the useful information to delineate the system behaviour. In general, the concept of model validity can be related with the similarity level between real and digital information. To be able to assess the correctness of the whole evolution in time of a specific information, both real and digital information can be considered as sequences, and their similarity level can be determined exploiting sequence comparison techniques. Hence, the goal of our methodology is to obtain a *similarity indicator* (e.g., a value ranging between 0 and 1). The model is defined valid if the indicator is greater than a predefined acceptance threshold set by the user.

In smart manufacturing systems, it is reasonable to assume that an adequate sensor network has been put in place, and data collection is possible within the same level of detail expected by the user. In this context, both real and digital data may be treated as sequences. A sequence is defined as a succession of values ordered in time. The starting point for both real and digital data is the event log, a registry of the events occurring in the system with the associated time-stamps. The events stored in the real event log depend on which type of information can be collected from the physical system. On the other hand, the digital event log can store all events occurring in the simulation model. In addition to events, the event log is useful to calculate KPIs. For instance, Figure 1 shows as example the extraction of the sequence of inter-departure times from an assembly station. Each time a part is done and it departs the station, the inter-departure time (IDT) is computed as the time elapsed since the previous part has left the station. This value is computed and recorded in the company information system. A derived KPI such as the system throughput can be calculated as $1/IDT$. The same information can be computed within a digital model.

The following sections provide a deeper understanding on the validation methodology. In section 3.1, the two validation types addressed in this work are defined, namely input and logic validation. Then, in section 3.2 input data and their level of detail are discussed. In section 3.3 the proposed procedure to perform both input and logic validation is introduced. Finally, in section 3.4 the specific techniques that are used for comparing data sequences are presented.

3.1 Types of Validation

As introduced by Nelson et al. (2013), a system is described by its logic and input data. Hence, the logic validation of the model concerns the building of the model itself and its assumptions. The input validation ensures the correctness of the input data fed to the model. The uncoupling of logic and input validation is useful to understand where a specific deviation has generated, and allows for a faster resolution. Traditional validation instead is usually performed in one single phase. As a consequence, when a model

is declared as not valid it is difficult to understand if the reason is either a wrong logical assumption or the wrong modeling of input data. Following the aforementioned insights, the proposed methodology takes into account both logic and input validation: each time a validation is performed, both aspects are checked.

3.2 Levels of Validation

The validation procedure can be adapted to the type of system is being analysed, and consequently to the level of detail of the respective data. This concept is illustrated in Figure 2. A complete set of events must be considered to focus on the whole system behaviour, while a partial set of events may represent only a portion of the first set, depending on a specific user interest (for instance, considering only the start and finish events of a process). For example, referring to Figure 1, the time-stamp associated with the exit of a finished part from station 2 represents an event belonging to the larger set of material flow events. Meanwhile, the inter-departure time represents a local performance indicator that is derived from the aforementioned set of events. Therefore, the corresponding data set has a higher level of granularity.

Accordingly, two levels of validation procedures can be distinguished, respectively (1) event-level validation and (2) performance-level validation. Event-level validation investigates if the sequences of events produced by the physical and digital system are comparable. It is applied when a detailed view on the system behaviour is required. Specific sections of the manufacturing system could be investigated: for example, the events produced by the process sequence in a manufacturing cell might be of interest within a validation procedure. An event is uniquely defined by the event identifier string. For example, the start of the process on a generic resource 1 can be indicated by the string *"S1"*. Together with the sequence containing information about the event IDs, also the sequence with the associated time-stamps ($\mathbf{t}$) can be used to measure the similarity. Performance-level validation compares the performance of the system with the ones estimated using digital experiments. In this case, only the sequence of KPI values is analysed to assess the similarity. It is possible to evaluate global KPIs, representing the general system performances (e.g., system time, inter-departure time, etc), or local KPIs, representing the performances of specific portions of the system (e.g., waiting time, buffer level, etc). This choice is done considering that performance–level validation is characterized by a lower level of detail. In addition, KPIs usually already contain information about time. For instance, the sojourn time in an assembly station is a performance indicator that is dependent on the time dimension. KPIs level validation focuses on the general performance of the system by comparing sequences of KPIs and it is characterized by a lower level of detail.

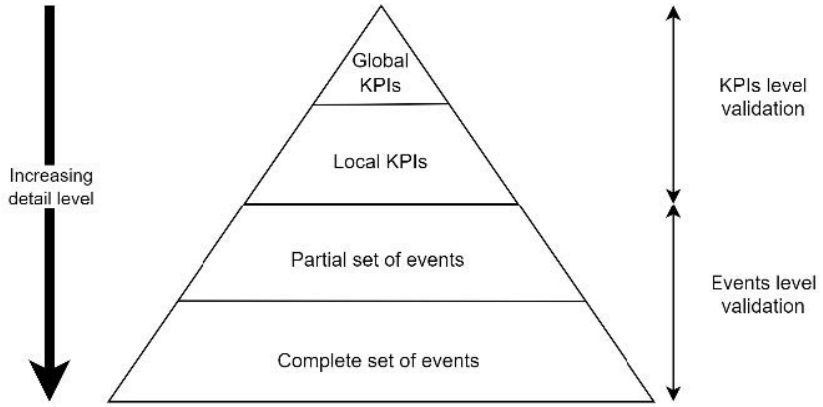


Fig. 2: Representation of the levels of detail achievable with the proposed validation methodology.

3.3 Online Validation Procedure

A model is defined valid if both its logic and inputs properly represent the ones of the real system. Based on this consideration, the validation procedure is divided in two parts, one for each aspect; Figure 3 shows an overview of the proposed validation procedure. First, logic validation checks the correctness of the model logical layout and assumptions. In this phase, digital data are obtained following the Trace Driven Simulation (TDS) (Law, 2007). Namely, each parameter of interest is monitored on the real system (e.g., the processing time of a welding station). The sensors in the real manufacturing system collect these data, and each data point is made available to be reproduced in the digital model. Input validation, shown at the bottom of Figure 3, assesses the correctness of the input data distributions. In this case, random variates must be generated in a way that can be comparable with the data collected from the physical system. This is done using a particular procedure named *quasi Trace-Driven Simulation* (Lugaresi et al., 2019). The validation procedure is conducted comparing a set of real and digital data to obtain a similarity indicator ranging between 0 and 1. Acceptance thresholds are defined in accordance to the user requirements. If data are characterized by an appropriate similarity level both in terms of logic and input, the model is considered a valid representation of reality and it can be used for decision making purposes. On the other hand, if the digital model is not capable to represent coherently the real system behaviour, an update is needed. The logic update implies a modification of the simulation logic model or even the generation of a new model, while an input update requires a modification of the input simulation models.

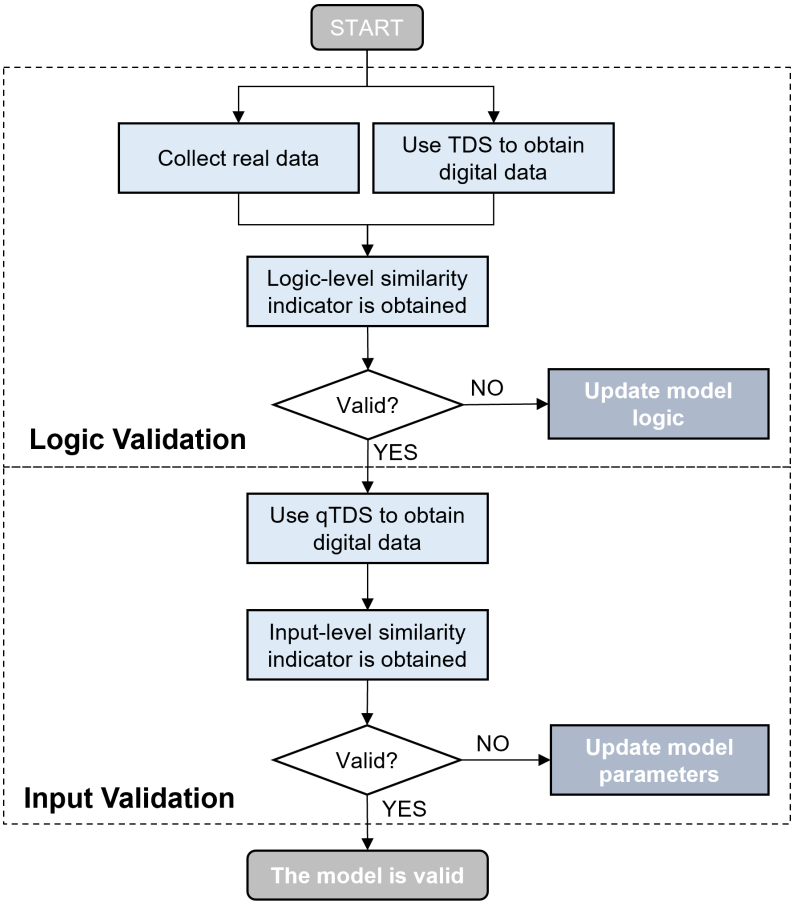


Fig. 3: Proposed online validation procedure (TDS: Trace Driven Simulation; qTDS: quasi Trace Driven Simulation).

3.4 Sequence Comparison Techniques

This section presents the techniques that have been identified and adapted to calculate the similarity indicators in the logic and input validations. The choice of sequence comparison techniques has been driven by the need to use a limited amount of data for the validation. Indeed, the selected techniques have been historically developed to compare limited data arrays, hence they fit the requirements for short-term validation. Starting from the techniques available in literature, proper normalization procedures have been done and additional parameters have been introduced to achieve an indicator capable to measure the similarity level of two sequences.

For coping with event-level validation, the sequence comparison techniques have been selected to be able to deal with sequence of strings. Moreover, in order to not declare equal two events significantly shifted over time, the associate sequence of time values has been added to the comparison. Regarding performance-level validation, sequence comparison techniques able to deal with sequences of real numbers

are investigated. Methods applicable to both sequences of strings and numbers are included in the research, in order to compare both sequences of events and KPIs. Moreover, the techniques have been selected taking into account that data sequences may be of different length and that there must not be limitations regarding the minimum data length. As a result, three techniques for the online validation of digital twins are proposed and listed in the following sections.

The following notation is used:

- **a** and **b** represent two sequences of data. For simplicity, the same notation is used in case the two sequences are made by numbers or strings (i.e., event identifiers). m and n represent the length of **a** and **b**, respectively.
- $a_{[i]}$ represents the i-th point of the sequence **a**.
- $a_{[i:j]}$ represents the portion of **a** sequence from the i-th to the j-th point.
- $t_{a,[i]}$ represents the time-stamp associated with the i-th data point in the sequence **a**.
- $d(x, y)$ is the Euclidean distance between two values x and y.

Table 2 collects the notation used along the whole paper.

3.4.1 Longest Common Sub-sequence (LCSS). This technique is introduced to deal with string comparison. It is mostly applied to bio-informatics and computational linguistics fields (Bergroth et al., 2000). This technique aims at identifying the maximum number of matching elements in the same sequential order. The original version of this technique does not take into account a possible shift over time of the two sequences, a clear limitation in case the digital model is simply reproducing events with a constant delay respect to the physical system. In this section a modified version of the LCSS technique is exploited to compute the similarity measure. This modification allows to compare sequences of event identifiers together with their respective time-stamp sequences t. Namely, two events with identical identifiers which are produced within a time interval longer than a threshold value δ are not declared similar. The parameter δ is defined as the *indifference threshold*, since a deviation smaller than δ is not sufficient to say the digital model is not valid. The choice of the indifference threshold is problem-dependent according to the validation-on-purpose concept (Sargent, 2013). In case of string identifiers, the distance d is equal to 0 if the strings are identical, 1 otherwise; for example, $d(\text{"}s1\text{"}, \text{"}s1\text{"}) = 0$, while $d(\text{"}s1\text{"}, \text{"}f2\text{"}) = 1$. The similarity measure is a matrix, where each element $L_{[i,j]}(\mathbf{a},\mathbf{b})$ corresponds to the comparison of the sequences $a_{[0:i]}$ and $b_{[0:j]}$, as reported in Equation (1).

$$L_{[i,j]}(\mathbf{a},\mathbf{b}) = \begin{cases} 0 & \text{if} \quad i = 0 \quad \vee \quad j = 0 \\ L(a_{i-1}, b_{j-1}) + 1 & \text{if} \quad d(a_i, b_j) = 0 \quad \wedge \quad d(t_{a,[i]}, t_{b,[j]}) < \delta \\ \max\{L(a_{i-1}, b_j), L(a_i, b_{j-1})\} & \text{else.} \end{cases} \tag{1}$$

Table 2: Notation used in this paper.

Variables	Explanation
Data sequences:	
$\mathbf{a}$	Vector of values representing a data sequence.
$\mathbf{t}_a$	Vector of acquisition time-stamps of each data point in the sequence $\mathbf{a}$.
a_i	i-th element of data sequence $\mathbf{a}$.
$a_{[i:j]}$	Portion of the data sequence $\mathbf{a}$, from the i-th to the j-th value.
Indicators:	
$L_{[i,j]}(a,b)$	LCSS-based similarity measure comparing the two data sequences $\mathbf{a}_{[0:i]}$ and $\mathbf{b}_{[0:j]}$.
$M_{[i,j]}(a,b)$	mLCSS-based similarity measure comparing the two data sequences $\mathbf{a}_{[0:i]}$ and $\mathbf{b}_{[0:j]}$.
$D_{[i,j]}(a,b)$	DTW-based similarity measure comparing the two data sequences $\mathbf{a}_{[0:i]}$ and $\mathbf{b}_{[0:j]}$.
$\Phi_m(a,b)$	Model validity indicator evaluated according to the method m comparing the two data sequences $\mathbf{a}$ and $\mathbf{b}$. $\Phi \in [0,1]$.
Parameters:	
δ	Indifference threshold, meaning a deviation smaller than δ is not significant enough to say the digital model is not valid.
ϵ	Time closeness threshold, two elements are declared similar if their time difference is smaller than ϵ.
w	Number of warning signals that do not trigger a model update action.
t_{query}	Length of the last interval of time used to collect data.
t_{update}	Time interval determining the frequency with which the similarity indicator is computed.
θ_m	Threshold used to compare the indicator based on the method m.
τ_1	Time to detect a change in the system.
τ_2	Time to detect that the system is correct after the resolution of the disruption.

Then, the LCSS-based similarity indicator, ranging between 0 and 1, is obtained dividing $L_{[m,n]}(\mathbf{a},\mathbf{b})$ by the length of the shortest sequence as reported in Equation (2).

$$\Phi_{LCSS}(\mathbf{a},\mathbf{b}) = \frac{L_{[m,n]}(\mathbf{a},\mathbf{b})}{\min(m,n)}. \tag{2}$$

3.4.2 Modified Longest Common Sub-sequence (mLCSS). A variation to the LCSS technique is applied to deal with sequences of real numbers. Differently from LCSS, a matching threshold distance ϵ is introduced to compare real numbers instead of strings. In this case, the information related to the sequence of time values is not considered in the comparison. The modification allows to declare similar two elements not equal but characterized by a Euclidean distance smaller than the ϵ parameter. The corresponding mLCSS-based similarity measure is a matrix, where each element $M_{[i,j]}(\mathbf{a},\mathbf{b})$ corresponds to the comparison of the sequences $a_{[0:i]}$ and $b_{[0:j]}$, as reported in Equation (3).

$$M_{[i,j]}(\mathbf{a},\mathbf{b}) = \begin{cases} 0 & \text{if} \quad i = 0 \quad \vee \quad j = 0 \\ M(a_{i-1}, b_{j-1}) + 1 & \text{if} \quad d(a_i, b_j) < \epsilon \\ \max\{M(a_{i-1}, b_j), M(a_i, b_{j-1})\} & \text{else.} \end{cases} \tag{3}$$

Then, the mLCSS-based similarity indicator is ranging between 0 and 1. This is achieved dividing $M_{[m,n]}(\mathbf{a},\mathbf{b})$ by the length of the shortest sequence as reported in Equation (4).

$$\Phi_{mLCSS}(\mathbf{a},\mathbf{b}) = \frac{\varphi_{mLCSS[m,n]}(\mathbf{a},\mathbf{b})}{\min(m,n)}. \tag{4}$$

3.4.3 Dynamic Time Warping (DTW). One of the most common techniques to measure the distance between two temporal sequences in the time domain analysis is Dynamic Time Warping (Müller, 2007). It is based on an algorithm able to identify the best alignment between two sequences and measure their distance. The procedure computes all possible alignments between each point of both sequences and then identifies, as the best alignment, the one that minimizes the distance. The achieved alignment is defined as warp path. The distance measure is a cumulated distance since it represents the sum of the distances between all matching points in the path. The DTW distance measure is a matrix, where each element $D_{[i,j]}(\mathbf{a},\mathbf{b})$ corresponds to the comparison of the sequences $a_{[0:i]}$ and $b_{[0:j]}$, as reported in Equation (5).

$$D_{[i,j]}(\mathbf{a},\mathbf{b}) = \begin{cases} 0 & \text{if} \quad i = 0 \quad \wedge \quad j = 0 \\ \infty & \text{if} \quad i = 0 \quad \vee \quad j = 0 \\ d(a_i, b_j) + \min\{D(a_i, b_{j-1}), D(a_{i-1}, b_j), D(a_{i-1}, b_{j-1})\} & \text{else.} \end{cases} \tag{5}$$

An additional normalization procedure allows to achieve the DTW similarity indicator ranging between 0 and 1. The two sequences **a** and **b** are normalized with respect to their maximum element, as in equation (6).

$$\bar{a} = \frac{\mathbf{a}}{\max(\max(\mathbf{a}), \max(\mathbf{b}))} \quad and \quad \bar{b} = \frac{\mathbf{b}}{\max(\max(\mathbf{a}), \max(\mathbf{b}))} \tag{6}$$

where m and n are the length of the sequences **a** and **b**, respectively.

Equation (5) is used to compute the cumulative distance $D_{[m,n]}(\mathbf{a}, \mathbf{b})$, which is normalized by dividing it by the maximum length of the two sequences. Finally, the DTW-based indicator is obtained as indicated in equation (7):

$$\Phi_{DTW}(\mathbf{a},\mathbf{b}) = 1 - \frac{D_{[m,n]}(\mathbf{a},\mathbf{b})}{\max(m, n)}. \tag{7}$$

4 Numerical Experiments

This section illustrates the experiments carried out to assess the applicability of the proposed methodology. The behaviour of each technique is tested with respect to the associated parameter settings. Two experimental campaigns are performed, respectively dealing with input validation and logic validation. In both cases, performance-level validation is carried out with two different reference KPIs: *system time* and *inter-departure time*. The first measure is frequently used to monitor and control the waiting time in queues. The second measure is the inverse of the throughput rate. Event-level validation considers start and finish events of the processing of each station and it is used when a detailed validation of the digital model is needed. The system and the digital model used to conduct the experiments are DES models realised on Rockwell Arena Simulation Software. The online validation methodology described in section 3 has been implemented in python[2].

4.1 System Description

The experiments regard the five-station closed system illustrated in Figure 4. The system is characterized by four stations in a closed loop with an inner loop composed of an additional station. Stations are assumed to be perfectly reliable and the first-come-first-served dispatching policy is applied. a station is indicated with the number s, and p_s represents the processing time on the s-th station measured in seconds per part. Each station has a stochastic processing time that is normally distributed with a mean μ and standard deviation σ. The distribution parameters and the utilisation of each station are reported in Table 3. The conveyors determine the length of the queue of pallets that wait in front of a station. The maximum number of pallets in the queue is fixed and results in buffers with a capacity equal to 4 pallets

[2] Python codes repository: `https://github.com/giovannilugaresi/digital_twin_validator`

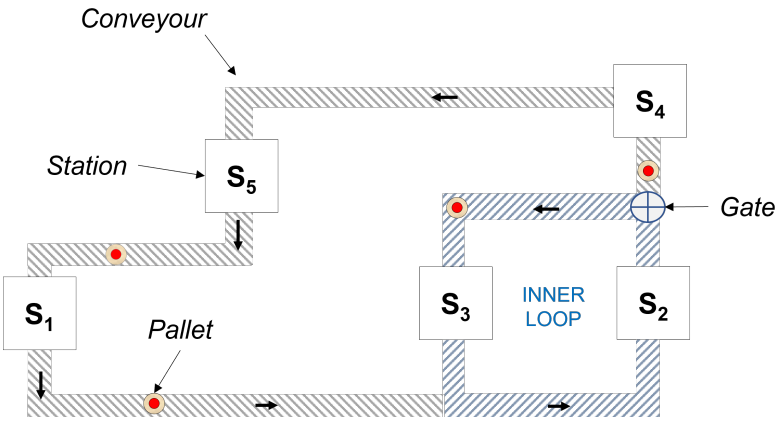


Fig. 4: Experiments – Manufacturing system used for the experimental campaign.

Table 3: Experiments – Processing time distributions and utilisation of each station.

Station s	**Processing Time** p_s [s]	**Utilization**
1	Norm(12, 0.07)	0.797
2	Norm(5, 0.07)	0.996
3	Norm(7, 0.07)	0.929
4	Norm(12, 0.07)	0.797
5	Norm(12, 0.07)	0.796

for each station. The Blocking After Service (BAS) discipline is applied. The number of circulating pallets is equal to 15. To complete the production cycle, pallets must go through the inner loop two times. The maximum number of pallets circulating in the inner loop is equal to 5. This limit is controlled by allowing the exit of parts from station $s = 1$.

4.2 Design of Experiments

Two separate experimental designs have been executed to test the applicability of the proposed techniques for coping with both input validation and logic validation. In the experiments, the digital model is modified by introducing deviations with respect to the parameters of the physical model. The consequence of this modification is a bias in the estimated performance. The proposed methodology is applied to detect this artificially introduced bias. The following sections explain in detail how the experiments have been designed.

4.2.1 Input Validation. Input validation is performed by applying the techniques proposed in section 3.4, where the sequence of data points coming from the real system is used to generate a sequence of

Table 4: Experiments – Input Validation: factors and respective levels of each method.

			Indicators		
Factors	**Levels**	**Values in the Digital Model**	Φ_{LCSS}	Φ_{mLCSS}	Φ_{DTW}
Mean μ	3	μ; $1.05\,\mu$; $1.2\,\mu$	•	•	•
Standard deviation σ	3	σ; $1.2\,\sigma$; $2\,\sigma$	•	•	•
Station s	2	bottleneck ($s = 2$); another station ($s = 4$)	•	•	•
Parameter δ	3	0.1; 0.3; 0.5	•		
Parameter ϵ	3	0.1; 0.3; 0.5		•	

digital points using the quasi-Trace Driven Simulation technique Lugaresi et al. (2019). The experimental design on input validation has been carried out in order to understand:

- How performance-level validation based on different KPIs reacts to deviations of the input distribution parameters of the digital model. Specifically, the mean and standard deviation of the processing time distribution of one specific station are modified on purpose.
- If the validation methodology is able to detect deviations also when the variations occur on a part of the system that does not strongly determine its performance (i.e., a non bottleneck station). A categorical factor called *station* is used to identify which station is subjected to the parameter deviation. This factor is characterized by two levels, that represent if the corresponding station is the bottleneck of the system or not.
- How the ϵ and δ parameter values affect the system response. The indifference threshold ϵ is chosen as a percentage of the reference KPI used for validating the digital model. Experiments investigate the behaviour of the methods using for both ϵ and δ values 10%, 30% and 50% of the real mean inter-departure time.

The system response is measured as the value of the similarity indicator for which convergence[3] is reached. The time can be measured as the number of parts or events necessary to reach a stable indicator. Table 4 summarizes factors, levels, and values of each method. Each experimental point has been replicated five times and the replications are independent. As a consequence, for the test of both LCSS- and mLCSS-based indicators, the experiments are characterized by 54 experimental points, and a total of 270 runs have been done. For the test of the DTW-based indicator, 18 experimental points are determined by the design, and a total of 90 runs have been done.

[3] Herewith convergence is defined as reached if the indicator does not vary more than the 1% of the value obtained with the whole set (i.e., 10,000 parts).

Table 5: Experiments – Logic Validation: factors and respective levels of each method.

			Indicators		
Factors	**Levels**	**Values in the Digital Model**	Φ_{LCSS}	Φ_{mLCSS}	Φ_{DTW}
N° pallets	3	15; 12; 7	•	•	•
Policy	2	BAS; BBS	•	•	•
Parameter δ	3	0.1; 0.3; 0.5	•		
Parameter ϵ	3	0.1; 0.3; 0.5		•	

4.2.2 Logic Validation. Logic validation is performed by applying the techniques listed in section 3.4, where the data points coming from the real system are directly used in the digital model respecting the Trace Driven Simulation principle. The logic validation experimental campaign has been done in order to understand:

- If logic validation can identify a change in the number of pallets. In the digital model, the number of pallets circulating in the system is modified on purpose (three levels).
- If logic validation can correctly identify a change in a production policy of the system. The station blocking policy is modified on purpose. Two levels are identified: blocking after service (BAS) and blocking before service (BBS) .
- If ϵ parameter value depends on the reference KPI used for performance-level validation. For the methods including ϵ and δ parameter, an additional factor is present with three levels.

Similarly to the experiments for input validation, the system response is measured as the indicator value for which convergence is reached.

Factors, levels, and values of each method are reported in Table 5. Each configuration of factors has been replicated five times and the replications are independent. As a consequence, experiments to test LCSS- and mLCSS-based indicators are characterized by 18 experimental points, and a total of 90 runs have been done. The experiments to test DTW-based indicator are characterized by 6 configurations, hence 30 experiments have been done.

4.3 Numerical Results

In this section the numerical results obtained from the experiments are described.

4.3.1 Input Validation Results. The detailed results of Kruskal-Wallis test are reported in the Appendix, in Table 6 for the main effects and in Table 7 for the interactions. The main effects plot and interaction plots are illustrated in Figure 5. Independently from the sequence comparison method, a

change of the processing time mean is significantly impacting the validity indicator, while the effect of the variation of the standard deviation is not significant. This means that it is possible to identify a variation of the former parameter, while the latter could remain unnoticed. The parameters δ and ϵ are significantly impacting the behaviour of the indicator, meaning their choice within a validation procedure is important. Also, the indicator is significantly influenced by the location of the parameter variation. Hence, if the alteration occurs in a station that is not significantly impacting the performance of the system (i.e., not the bottleneck), the validation procedure could not identify this change. The interactions among the parameters for the input validation are not significant in the reported experiments. In the following, the most significant results for each developed indicator are reported.

4.3.1.1 LCSS-based Indicator. Figure 6a shows the LCSS-based indicator for different configurations of input distribution parameters as a function of the number of events considered. Results are associated to δ parameter equal to 50% of the mean inter-departure time. When the model is characterised by correct input distribution parameters, the similarity indicator shows high values (blue curve, i.e. $\mu = 5$, $\sigma = 0.7$). As the input parameters deviate with respect to the ones characterizing the system, the indicator drops. This behaviour highlights the capability of LCSS to detect deviations between the system and the model in terms of occurring events due to a wrong estimation of the input distribution parameters of the model. In these experiments, the logic of the digital model is not changed. Figure 6b instead shows the indicator evaluated for a model with not correct input distribution parameters for different values of δ, the indifference zone parameter (i.e. δ=10%, δ=30%, δ=50%). As expected, it is observed that the greater δ, the higher the indicator.

4.3.1.2 mLCSS-based Indicator. Figure 7a shows the mLCSS-based indicator using as reference KPI the inter-departure time. The indicator is reported as a function of the number of produced parts and the ϵ parameter is selected as 30% of the mean inter-departure time. The highest curve (blue curve, i.e. $\mu = 5$, $\sigma = 0.7$) represents the similarity indicator associated to correct input distribution parameters of the model. In this case, the indicator is characterized by large values meaning the digital model is adequately representing the system. As the input parameters deviate from the ones characterising the system, the similarity indicator drops as represented by curve in the middle (orange curve, i.e. $\mu = 5.25$, $\sigma = 0.84$) and by the one on the bottom part of the graph (green curve, i.e. $\mu = 6.25$, $\sigma = 1.4$). Similarly to the LCSS-based method, results prove the capability of the mLCSS-based indicator to detect deviations in terms of KPIs. Figure 7b reports the indicator associated to a model with a large deviation in the input distribution parameters for different values of ϵ parameter (i.e. ϵ=10%, ϵ=30%, ϵ=50%). The larger the parameter value, the more the indicator rises, meaning significant changes between the system and the digital model are more likely to be accepted.

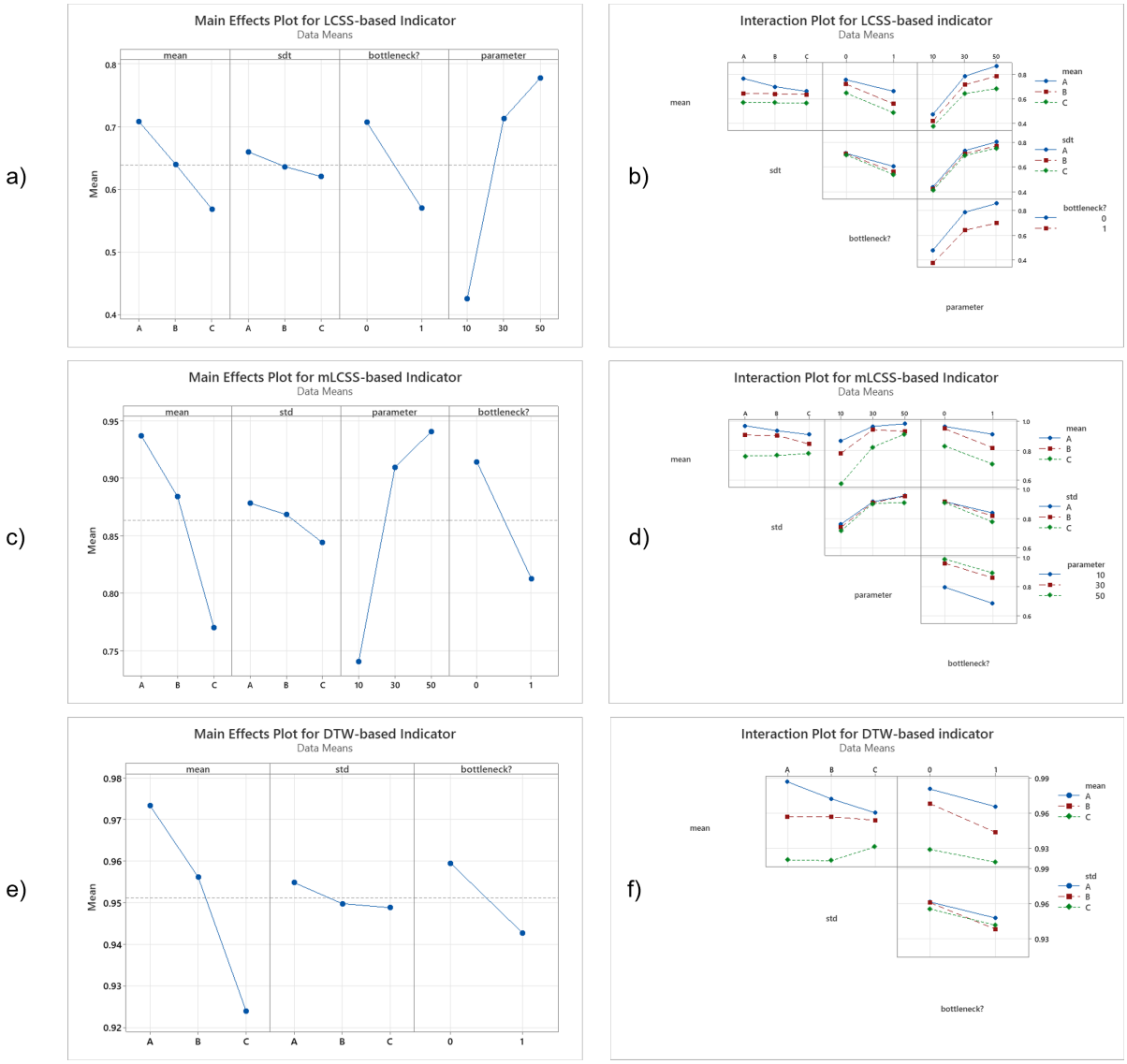


Fig. 5: Experiments – Input validation (KPI: *inter-departure time*): (a) main effects plot and (b) interaction plot for LCSS-based indicator, (c) main effects plot and (d) interaction plot for mLCSS-based indicator, (e) main effects plot and (f) interaction plot for DTW-based indicator.

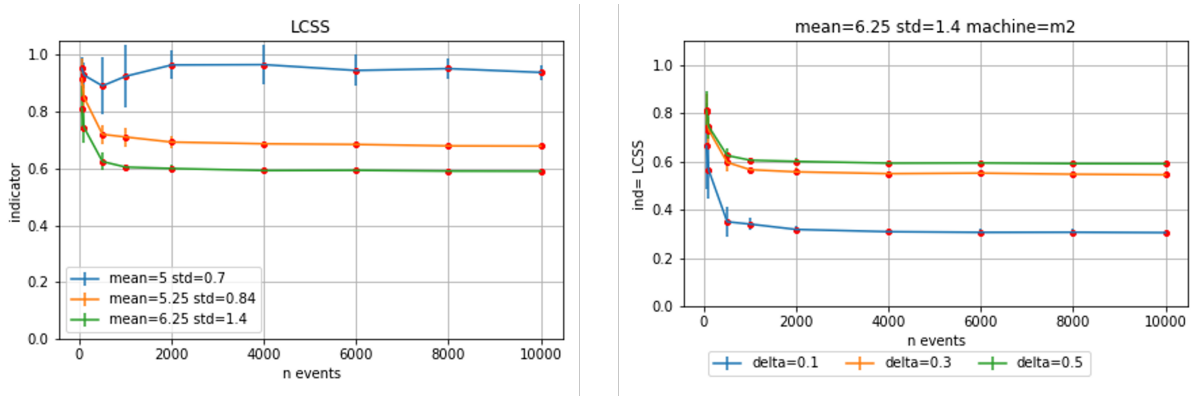


Fig. 6: Experiments – Input validation at event-level: 95% confidence intervals of (a) LCSS-based indicator with $\delta = 0.5$ (b) LCSS-based indicator obtained with wrong input distribution parameters for $\delta = 0.1$, $\delta = 0.3$, and $\delta = 0.5$.

4.3.1.3 DTW-based Indicator. Figure 8 shows the DTW-based indicator using as reference KPI the inter-departure time, the indicator is reported as a function of the number of produced parts. The highest curve (blue curve, i.e. $\mu = 5$, $\sigma = 0.7$) represents the similarity indicator associated to correct input distribution parameters of the model. As expected, the indicator is characterized by high values meaning the digital model is a proper representation of reality. As the input parameters deviate from the ones characterising the system, the similarity indicator drops as represented by the curve in the middle (orange curve, i.e. $\mu = 5.25$, $\sigma = 0.84$) and by the one in the bottom part of the graph (green curve, i.e. $\mu = 6.25$, $\sigma = 1.4$). Results prove the capability of DTW to detect deviations between the system and the digital model in terms of KPIs due to a wrong estimation of the input distribution parameters. It is worth to notice that this is true also with 50 and 100 data points, which demonstrates that the indicator is sensitive also with relatively small data sets.

4.3.2 Logic Validation Results. The results of Kruskal-Wallis test are reported in the Appendix, in Table 8 for the main effects and in Table 9 for the interactions. The main effects plot and interaction plots are illustrated in Figure 9. Independently from the sequence comparison method, the number of pallets is significantly impacting the validity indicator. The change of the blocking policy is correctly detected by the LCSS- and mLCSS-based indicators, while it is not discerned by the DTW-based approach. The parameters δ and ϵ are significantly altering the behaviour of the indicators, confirming that their choice is important for a validation procedure. The interactions among the parameters δ and ϵ are not significant for all methods, while the interactions among the number of pallets and the blocking policy are significant. Hence, if the starting point is a situation in which the model is correct (for instance, in Figure 9b, the BAS policy) and another significant parameter is changed, the validation procedure is correctly identifying the

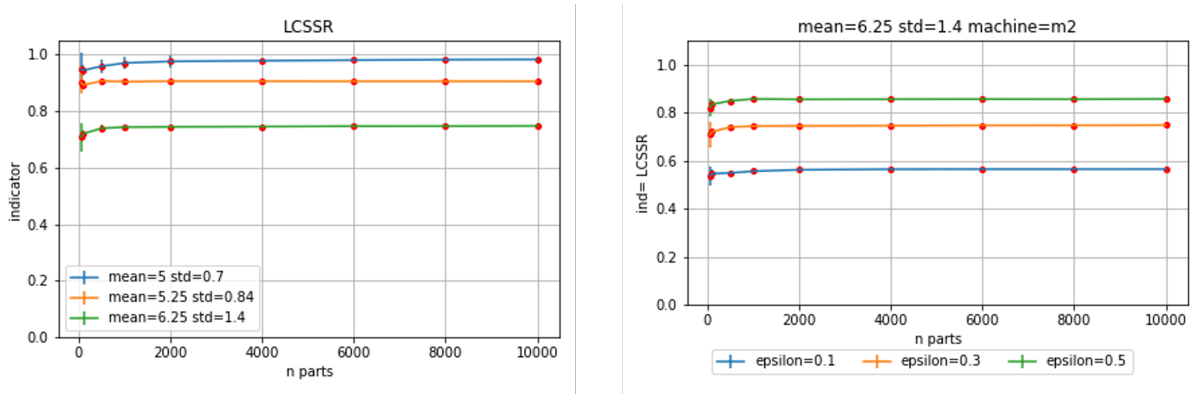


Fig. 7: Experiments – Input validation at performance-level: 95% confidence intervals of (a) mLCSS-based indicator with $\epsilon = 0.3$ and inter-departure time as reference KPI (b) mLCSS-based indicator with wrong input distribution parameters for $\epsilon = 0.1$, $\epsilon = 0.3$, and $\epsilon = 0.5$.

change. However, the change is not identified when the initial point is an already altered model. This result feeds the intuition that another way to perform validation is by altering a significant parameter of the production system in the digital model. If the validity indicator does not change substantially, it probably implies that the digital model was already incorrect.

The most significant results are shown in figures 10, 11, and 12. Each figure shows the indicator for three levels of circulating pallets. All the figures show results based on the BAS policy.

4.3.2.1 LCSS-based Indicator. Figure 10 illustrates the results of event level validation with the LCSS indicator with δ parameter equal to 30% of the mean inter-departure time. The highest curve (blue curve, i.e. N° pallet=15) shows that when the number of circulating pallets and blocking policy are correct, the indicator value is stable on 1. When the number of circulating pallets in the digital model is reduced, the indicator drops as represented by the curve in the middle (orange curve, i.e. N° pallet=12) and by the one on the bottom part of the plot (green curve, i.e. N° pallet=7).

4.3.2.2 mLCSS-based Indicator. Figure 11 illustrates the mLCSS-based indicator for inter-departure time associated to BAS policy with ϵ parameter equal to 30% of the mean intedeparture time. The highest curve (blue curve, i.e. N° pallet=15) is characterized by an indicator value equal to 1. The curve in the middle (orange curve, i.e. N° pallet=12) and the one on the bottom (green curve, i.e. N° pallet=7) represent the indicator evaluated for a different number of circulating pallets. The deviation in the number of circulating pallets does not affect significantly the indicator based on the inter-departure time.

4.3.2.3 DTW-based Indicator. Figure 12 shows the results of performance-level validation using as reference KPI the system time and a DTW-based similarity measure. The highest curve (blue curve, i.e. N° pallet=15) corresponds to the model with reference blocking policy and reference number of circulating

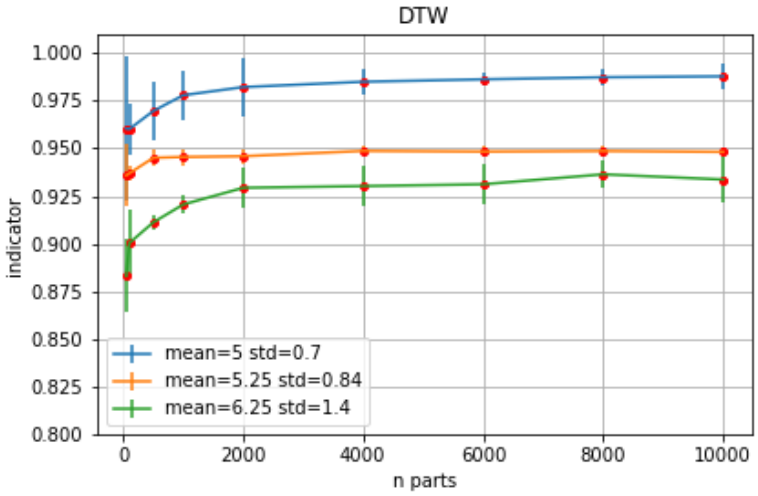


Fig. 8: Experiments – Input validation at performance-level: 95% confidence intervals of DTW-based validity indicator (KPI: *inter-departure time*).

pallets, the indicator value is equal to 1. The system time is significantly affected by the deviation in the number of pallets as represented by the middle curve (orange curve, i.e. N° pallet=12) and by the bottom curve (green curve, i.e. N° pallet=7).

5 Remarks on Implementation

The experimental campaign presented in section 4 provides useful insights on the practical application of the methodology from Figure 3. The following list aims at exploiting the lessons learned toward the implementation of online validation in a manufacturing system.

5.1 Logic Validation Setting

All three tested techniques can be applied for logic validation, if the interest is to detect changes that strictly influence the performance of the system (e.g., number of circulating pallets). For less sensitive changes such as the change of a production policy, based on the results of the experiments it is encouraged the use of LCSS- or mLCSS-based indicators. For LCSS and mLCSS, the parameters ϵ and δ should be chosen depending on the indifference zone. Indeed, the experiments show that the parameters values influence the indicator: a higher parameter value will tend to have a higher validity indicator, hence a larger indifference zone results in a higher probability that the model will be declared as valid. Also, it is worth to notice that logic validation is not characterized by the intrinsic error of the random number generation. Consequently, for correct logical assumptions and a digital-digital comparison the indicator

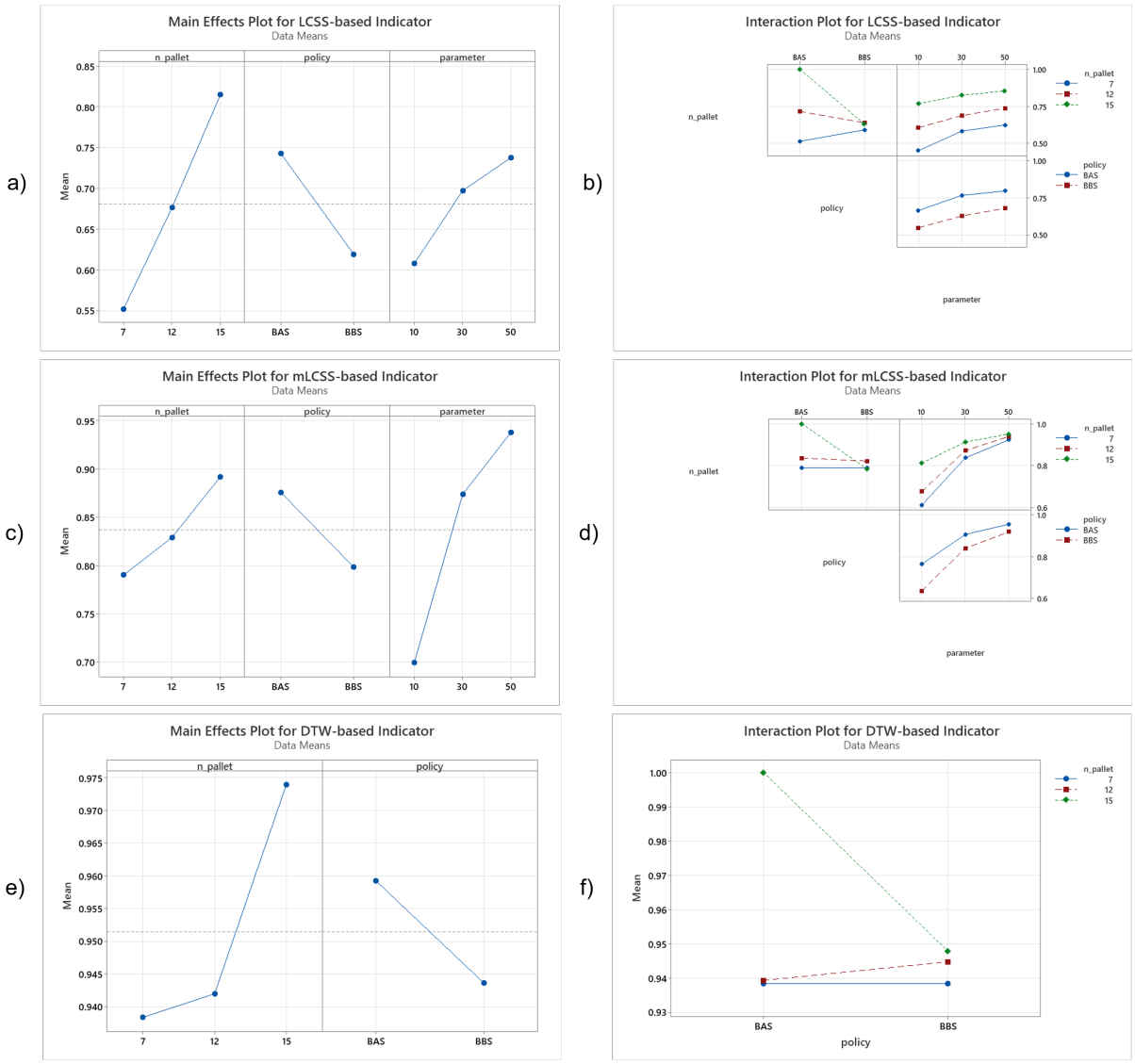


Fig. 9: Experiments – Logic validation (KPI: *inter-departure time*): (a) main effects plot and (b) interaction plot for LCSS-based indicator, (c) main effects plot and (d) interaction plot for mLCSS-based indicator, (e) main effects plot and (f) interaction plot for DTW-based indicator.

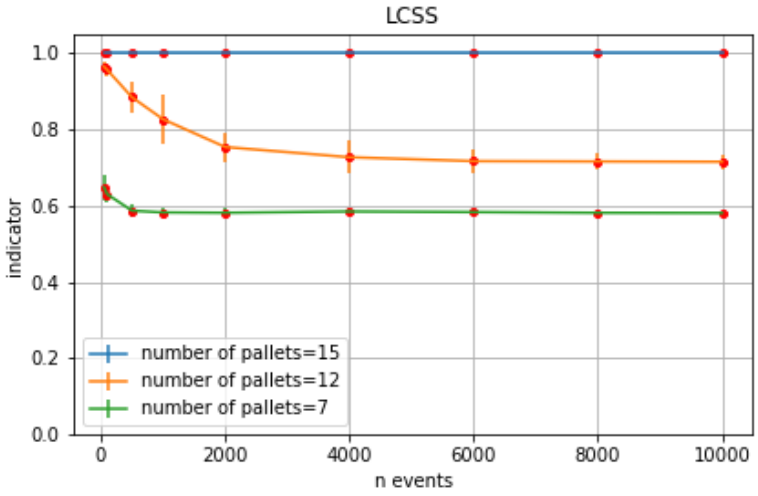


Fig. 10: Experiments – Logic validation at event-level: 95% confidence intervals of LCSS-based indicator with $\delta = 0.3$.

is always equal to 1. More generally, if the logical assumptions are accurate, a real-digital comparison for logic validation is characterized by larger indicator values with respect to input validation, especially for a small data set. This consideration allows to select lower values for ϵ and δ parameters with respect to the ones associated to input validation.

5.2 Input Validation Setting

Input validation should be set together with the quasi-trace driven simulation (Lugaresi et al., 2019) on the input data from the real system. All three tested methods proved to correctly identify the changes on parameters that directly influence the performance of the system (e.g., change of mean processing time on bottleneck station). LCSS- and mLCSS-based indicators should be set with the proper values of ϵ and δ parameter values. Different KPIs react differently to the same variation of the model. Hence, the proper ϵ value for mLCSS-based indicators depends specifically on the KPI under investigation. An ad-hoc calibration is needed for each validation campaign. For the δ parameter, the most appropriate value increases with the increase of the system complexity.

5.3 Threshold Values Definition

In order to identify the validity of the digital model online, validity thresholds should be defined for the specific application. Given the different input data, logic and input validation should be assigned different respective thresholds. The definition of thresholds is a non trivial task, and it remains strictly application dependent. As a general guideline, initial experiments should be performed using historical

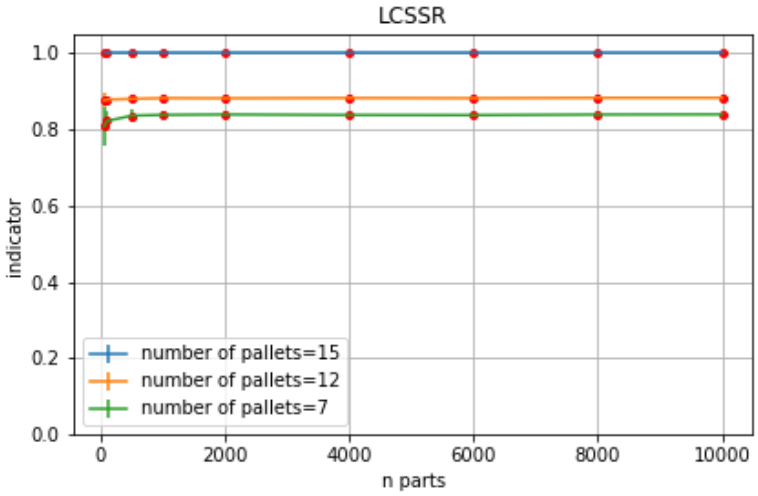


Fig. 11: Experiments – Logic validation at performance-level: 95% confidence intervals of mLCSS-based indicator (KPI: *inter-departure time*).

data and valid models, in order to identify the upper bounds of the similarity indicators. Then, a maximum allowed deviation can be defined (e.g., a 5 % drop of the indicator with respect to the upper bound). However, the threshold should always be revisited once new data are available and the noise of the real system can be apprehended, while being coherent with the allowed error probability.

5.4 Online Application Setting

Once the logic and input validation techniques have been set, the procedure of Figure 3 can be implemented in a real system. For such implementation, two main decisions have to be made concerning the quantity of data to be collected at each step, and the query update frequency. Despite the choice of these parameters is not the focus of this work, their influence and behaviour may be qualitatively addressed in the following:

- *Data set size.* In general, it is reasonable to assume that the faster the production pace, the smaller the parameters can be to obtain an adequate size data set. Besides, the size of the datasets will influence the computation time of the queries, which should be safely smaller than the update time interval.
- *Update frequency.* In general, the update frequency should be chosen in relation with the time to signal of interest, namely the time needed to identify a disruption that is desired to guarantee. The production pace will also directly influence this choice, as systems with a low cadency will tend to have a lower update frequency than faster ones.

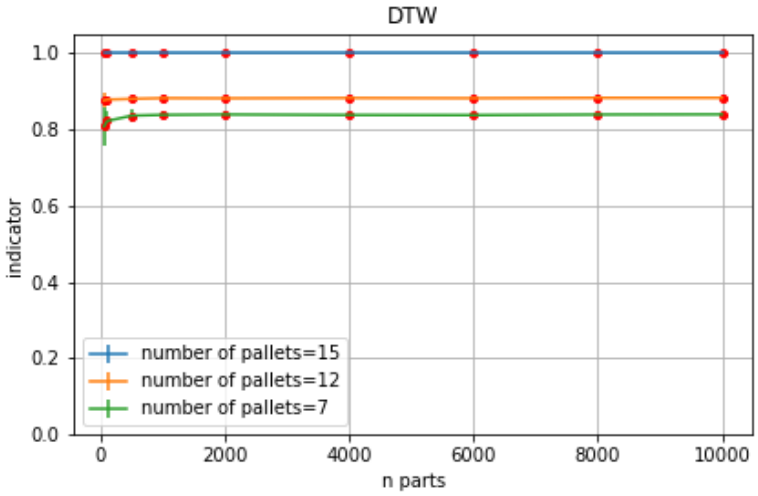


Fig. 12: Experiments – Logic validation at performance-level: 95% confidence intervals of the DTW-based indicator (KPI: *system time*).

In (Lugaresi et al., 2022), an initial attempt has been made by applying DTW-based validation in a controlled laboratory environment, using a query time of 10 minutes and an update frequency of 1 minute. The choice of the parameters has been based on the production pace of the system under analysis and on how fast deviations need to be identified. Hence, they should be identified case by case based on the specific application. Nevertheless, as the data set used to compute the indicator contains data acquired during the correct functioning of the system, some time is always required to detect the occurrence of a disruption. Therefore, the amount of data and the frequency with which the indicator is computed must be carefully chosen. Future research should be dedicated on designing specific experiments to further define the implementation of the online validation methodology.

6 Conclusions

This work proposes an innovative methodology to deal with online validation of simulation-based digital twins of manufacturing systems. Thanks to the proposed techniques, it is possible to guarantee that a digital model is always updated both in terms of inputs and logic, enabling short-term decision making. The main advantages of this research are the possibility to achieve reliable results even with limited data sets and the capability to monitor the whole trend of information instead of the sole average behaviour. The validation procedure allows to detect deviations rapidly and avoid possible erroneous alerts due to wrong data acquisition.

The proposed methodology is subject to several limitations that highlight the need of further research. In a real-digital comparison, the logic validation indicator associated to a correct logical layout does not

reach a value equal to 1, due to the fact that it is not possible to achieve a perfect modelling of the system. Hence, a prior analysis on the system is still required to properly identify the acceptance thresholds. The definition of threshold and parameter values should be based on the system topology and characteristics, as well as its real-time state. Future research should be dedicated to introduce a learning procedure to automatically identify the acceptance thresholds based on the system dynamics and available historical data. To this end, the integration with relevant works in the same direction (Cruz et al., 2022; Castano et al., 2023) could be explored. The accuracy level requested by the user should also be taken into account. Last but not least, this methodology is not capable to identify the specific cause for the anomaly behaviour of the model. A significant development would be to study methods and tools to automatically detect the reason for which the model is not valid.

In future work, it is necessary to study the validity of the proposed methodology within real manufacturing environments. A step in this direction can be represented by tests using model factories. For instance, in (Lugaresi et al., 2022), a lab-scale model is used to implement a validation technique within a digital twin architecture. Experiments in a physical laboratory can bring the proposed approaches closer to real environments. Further, in future research the application of the proposed methodology to other types of manufacturing system data should be explored. Indeed, most systems record physical quantities along the sequential processing of parts, such as temperature or pressure. It is therefore of interest to validate digital models of specific physical processes within the production cycle, such as a heating curve for a treatment oven. Theoretically it is possible to extend the proposed approach toward these data types, provided the setting of proper sampling and pre-processing techniques. Last but not least, the present work has been driven by the need of validating digital models with few data. However, some manufacturing systems do produce sufficiently large datasets even in the short term, typically when they present a high production cadency. Hence, the proposed methodology could be compared with a set of traditional techniques to identify specific situations where both could be applied.

Bibliography

O. Balci. Validation, verification, and testing techniques throughout the life cycle of a simulation study. Annals of operations research, 53(1):121–173, 1994.

O. Balci. Verification, validation, and testing. Handbook of simulation, 10(8):335–393, 1998.

O. Balci and R. G. Sargent. Some examples of simulation model validation using hypothesis testing. Technical report, Institute of Electrical and Electronics Engineers (IEEE), 1982.

J. Banks. Handbook of simulation: principles, methodology, advances, applications, and practice. John Wiley & Sons, 1998.

G. V. Bard. Spelling-error tolerant, order-independent pass-phrases via the damerau-levenshtein string-edit distance metric. Cryptology ePrint Archive, 2006.

L. Bergroth, H. Hakonen, and T. Raita. A survey of longest common subsequence algorithms. In Proceedings Seventh International Symposium on String Processing and Information Retrieval. SPIRE 2000, pages 39–48. IEEE, 2000.

M. Camargo, M. Dumas, and O. González-Rojas. Automated discovery of business process simulation models from event logs. Decision Support Systems, 134:113284, 2020.

F. Castano, Y. J. Cruz, A. Villalonga, and R. E. Haber. Data-driven insights on time-to-failure of electromechanical manufacturing devices: A procedure and case study. IEEE Transactions on Industrial Informatics, 19(5):7190–7200, 2023.

Y. J. Cruz, M. Rivas, R. Quiza, R. E. Haber, F. Castaño, and A. Villalonga. A two-step machine learning approach for dynamic model selection: A case study on a micro milling process. Computers in Industry, 143:103764, 2022.

L. Gong, B. Chen, W. Xu, C. Liu, X. Li, Z. Zhao, and L. Zhao. Motion similarity evaluation between human and a tri-co robot during real-time imitation with a trajectory dynamic time warping model. Sensors, 22(5):1968, 2022.

R. M. Howard. Principles of random signal analysis and low noise design: The power spectral density and its applications. John Wiley & Sons, 2004.

A. Khan, M. Dahl, P. Falkman, and M. Fabian. Digital twin for legacy systems: Simulation model testing and validation. In 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE), pages 421–426. IEEE, 2018.

J. P. Kleijnen. Experimental design for sensitivity analysis, optimization, and validation of simulation models. Handbook of simulation, 173:223, 1998.

H. W. Kuhn. The hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2):83–97, 1955.

A. M. Law. Simulation Modeling & Analysis. McGraw-Hill, New York, NY, USA, 4 edition, 2007.

D. Leroy, E. Bousse, A. Megna, B. Combemale, and M. Wimmer. Trace comprehension operators for executable dsls. In Modelling Foundations and Applications: 14th European Conference, ECMFA 2018, Held as Part of STAF 2018, Toulouse, France, June 26-28, 2018, Proceedings 14, pages 293–310. Springer, 2018.

M. Liu, S. Fang, H. Dong, and C. Xu. Review of digital twin about concepts, technologies, and industrial applications. Journal of Manufacturing Systems, 58:346–361, 2021.

G. Lugaresi, G. Aglio, F. Folgheraiter, and A. Matta. Real-time validation of digital models for manufacturing systems: A novel signal-processing-based approach. In 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), pages 450–455. IEEE, 2019.

G. Lugaresi, S. Gangemi, G. Gazzoni, and A. Matta. Online validation of simulation-based digital twins exploiting time series analysis. In 2022 Winter Simulation Conference (WSC). IEEE, 2022.

T. Marquardt, C. Cleophas, and L. Morgan. Indolence is fatal: research opportunities in designing digital shadows and twins for decision support. In 2021 Winter Simulation Conference (WSC), pages 1–11. IEEE, 2021.

L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda. Cyber-physical systems in manufacturing. Cirp Annals, 65(2):621–641, 2016.

L. E. Morgan and R. R. Barton. Fourier trajectory analysis for system discrimination. European Journal of Operational Research, 296(1):203–217, 2022.

M. Müller. Dynamic time warping. Information retrieval for music and motion, pages 69–84, 2007.

P. Muñoz, M. Wimmer, J. Troya, and A. Vallecillo. Using trace alignments for measuring the similarity between a physical and its digital twin. In Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, pages 503–510, 2022.

V. Naderifar, S. Sahran, and Z. Shukur. A review on conformance checking technique for the evaluation of process mining algorithms. TEM Journal, 8(4):1232, 2019.

E. Negri, L. Fumagalli, and M. Macchi. A review of the roles of digital twin in cps-based production systems. Procedia Manufacturing, 11:939–948, 2017.

B. Nelson et al. Foundations and methods of stochastic simulation. A first course. International series in operations research & management science, 187, 2013.

L. Overbeck, A. Le Louarn, O. Brützel, N. Stricker, G. Lanza, J. Franke, and P. Schuderer. Continuous validation and updating for high accuracy of digital twins of production systems. Simulation in Produktion und Logistik, pages 609–617, 2021.

R. Sargent. Verification and validation of simulation models. Journal of Simulation, 7:12–24, 2013.

R. G. Sargent. Verification and validation of simulation models. In Proceedings of the 2010 winter simulation conference, pages 166–183. IEEE, 2010.

A. Schmetz, T. H. Lee, D. Zontar, and C. Brecher. The time synchronization problem in data-intense manufacturing. Procedia CIRP, 107:827–832, 2022.

L. W. Schruben and V. J. Cogliano. Simulation sensitivity analysis: a frequency domain approach. Technical report, Institute of Electrical and Electronics Engineers (IEEE), 1981.

W. M. P. van der Aalst. Process Mining: Data Science in Action. Springer, Heidelberg, 2 edition, 2016. ISBN 978-3-662-49850-7. https://doi.org/10.1007/978-3-662-49851-4.

Appendix

This section reports the detailed results of the experimental campaign described in section 4 by using the Kruskal-Wallis test.

Table 6: Experiments – Input Validation: results of the Kruskal-Wallis test for main effects.

Method	**KPI**	**Factor**	**DF**	**H-Value**	**P-Value**	**N Samples**
LCSS	IDT	Mean	2	29.08	0	90
		Std	2	2.82	0.244	90
		Parameter δ	2	142.6	0	90
		Station s	1	44.59	0	135
mLCSS	IDT	Mean	2	55.96	0	90
		Std	2	4.86	0.88	90
		Parameter ϵ	2	112.53	0	90
		Station s	1	55.17	0	135
	ST	Mean	2	122.9	0	90
		Std	2	1.33	0.513	90
		Parameter ϵ	2	41.01	0	90
		Station s	1	72.96	0	135
DTW	IDT	Mean	2	65.52	0	30
		Std	2	1.13	0.568	30
		Station s	1	9.6	0.002	45
	ST	Mean	2	43.27	0	30
		Std	2	0.3	0.863	30
		Station s	1	24.55	0	45

Table 7: Experiments – Input Validation: results of the Kruskal-Wallis test for interactions.

Method	**KPI**	**Factor**	**DF**	**H-Value**	**P-Value**	**N Samples**
LCSS	IDT	Mean*Std	1	1.28	0.258	40
		Mean*Parameter δ	1	0.21	0.648	40
		Mean*Station s	1	0.89	0.346	40
		Std*Parameter δ	1	0.04	0.832	40
		Std*Station s	1	0.79	0.373	40
		Parameter δ*Station s	1	0.07	0.799	40
mLCSS	IDT	Mean*Std	1	2.61	0.106	40
		Mean*Parameter ϵ	1	0.49	0.485	40
		Mean*Station s	1	1.49	0.222	40
		Std*Parameter ϵ	1	0.03	0.866	40
		Std*Station s	1	0.59	0.441	40
		Parameter ϵ*Station s	1	1.17	0.279	40
DTW	IDT	Mean*Std	1	5.16	0.023	20
		Mean*Station s	1	0.15	0.695	20
		Std*Station s	1	0.01	0.903	20

Table 8: Experiments – Logic Validation: results of Kruskal-Wallis test for the main effects.

Method	**KPI**	**Factor**	**DF**	**H-Value**	**P-Value**	**N Samples**
LCSS	IDT	N pallets	2	33.3	0	30
		Policy	1	13.4	0	45
		Parameter δ	2	19.91	0	30
mLCSS	IDT	N pallets	2	11.83	0.003	30
		Policy	1	11.69	0.001	45
		Parameter ϵ	2	38.36	0	30
	ST	N pallets	2	80.42	0	30
		Policy	1	0.22	0.637	45
		Parameter ϵ	2	1.89	0.388	30
DTW	IDT	N pallets	2	18.23	0	10
		Policy	1	0.08	0.771	15
	ST	N pallets	2	25.95	0	10
		Policy	1	0.04	0.835	15

Table 9: Experiments – Logic Validation: results of Kruskal-Wallis test for the interactions.

Method	**KPI**	**Factor**	**DF**	**H-Value**	**P-Value**	**N Samples**
LCSS	IDT	N Pallet*policy	1	9.34	0.002	20
		N pallet*parameter δ	1	0.11	0.735	20
		policy*parameter δ	1	0.48	0.49	20
mLCSS	IDT	N Pallet*policy	1	7.61	0.006	20
		N pallet*parameter ϵ	1	5.1	0.024	20
		policy*parameter ϵ	1	0.15	0.695	20
DTW	IDT	N Pallet*policy	1	0.89	0.345	10