Explainable Metamodels for ATM Performance Assessment

Christoffer Riis[†], Francisco Antunes[†], Tatjana Bolić[*],
Gérald Gurtner[*], Francisco Câmara Pereira[†], Carlos Lima Azevedo[†]

[†]DTU Management
Machine Learning for Smart Mobility
{chrrii,franant,camara,climaz}@dtu.dk

[*]University of Westminster
School of Architecture and Cities
{t.bolic,g.gurtner}@westminster.ac.uk

***Abstract*—Fast-time simulation constitutes a well-known and long-established technique within the Air Traffic Management (ATM) community. However, it is often the case that simulation input and output spaces are underutilized, limiting the full understandability, transparency, and interpretability of the obtained results.**

In this paper, we propose a methodology that combines simulation metamodeling and SHapley Additive exPlanations (SHAP) values, aimed at uncovering the intricate hidden relationships among the input and output variables of a simulated ATM system in a rather practical way. Whereas metamodeling provides explicit functional approximations mimicking the behavior of the simulators, the SHAP-based analysis delivers a systematic framework for improving their explainability. We illustrate our approach using a state-of-the-art ATM simulator across two case studies in which two delay-centered performance metrics are analyzed. The results show that the proposed methodology can effectively make simulation and its results more explainable, facilitating the interpretation of the obtained emergent behavior, and additionally opening new opportunities towards novel performance assessment processes within the ATM research field.

***Keywords*—Air Traffic Management Simulation Modeling; Simulation Metamodeling; XGBoost; Model Explainability; SHAP values**

I. Introduction

Modern Air Traffic Management systems consist of a plethora of players, variables, uncertainty, and, ultimately, human behavior, interacting across both airspace and ground operation levels. By aiming at encompassing all these essential elements while ensuring the safety and efficiency of air traffic flows [1], ATM systems typically become inherently hard to model [2]. Due to this overwhelming complexity, such systems can reliably be modeled and studied through fast-time simulation approaches. Simulations allow researchers and practitioners to investigate, test, and propose a wide range of designs and alternative solutions within a virtual environment, i.e., a simulator[1], which would be otherwise practically infeasible to conduct in the real-world system.

Despite the well-known advantages in modeling complex and real-world stochastic systems and their evolution over time, simulation-based solutions suffer from a major drawback: they end up being developed and employed as opaque tools with often little room for understandability, transparency, and interpretability with regard to the details of the underlying simulation model and the obtained results. This is mostly an immediate consequence of their intrinsic modeling nature rather than an intentional design of the simulators' developers. In fact, whereas it is expected that simulators' internal individual components (equations, functions, formulations, theoretical foundations, etc.) and their logical interactions can be clearly identified and understood, their external emergent behavior may fall behind in terms of explainability. Furthermore, the understanding of this high-level behavior is potentially aggravated for end-user practitioners, policy-makers, and non-experts, who are typically oblivious to the intricate low-level implementation details.

The ATM R&D community is familiar with the trade-offs between benefits and disadvantages posed by fast-time simulation modeling due to its long experience with such techniques [1], [3]–[6]. While any simulation model is by and large a simplified representation of the real system [7], most may still evolve to be complex software programs with numerous input/output variables and parameters, requiring vast amounts of data and workload for calibration. Trivially, this complexity is an increasing function of the degree of detail, realism, and purpose of the simulation approach, as well as of the actual system to be modeled and the problem to be addressed. Particularly overwhelming dimensional sizes and value ranges of input spaces can constitute a serious burden in the exploration of the simulation behavior as a whole. Besides, the lack of an explicit and manageable closed-form function that transforms the former into the latter, as opposed to pure analytical approaches, can lead to difficulties in understanding the true impact of the input variables and parameters in the system's output metrics or Key Performance Indicators (KPIs), and their interrelationships. Simulation metamodeling [8], [9] provides a framework to approximate the implicit unknown relationship transforming the inputs into the outputs by an explicit known functional form, thereby opening the way to better understand the simulator's behavior. To the best of our knowledge, the use of these techniques in the scope of ATM

[1]Technically, a 'simulator' is the computer implementation of a 'simulation model.' We use both terms interchangeably in this text.

research is quite recent [10].

Many ATM simulation-based studies focus on performance impact assessment of new proposed solutions and concepts (e.g., single (SESAR) Solutions [11]–[13][2]) on a given actual system or planned one, often relying on a manual exploration of the underlying simulator's behavior. Domain knowledge and expert-driven scenario design, and what-if approaches are ultimately employed in an effort to reduce and discretize the input space into a limited set of possible system states to investigate. Despite the utility of these methodologies, especially when it comes to the communication of the results to stakeholders, there is a risk of failing to properly assess the simulated system's behavior in its entirety, leaving relevant simulation input regions unexplored. Hence, by narrowing down the simulation possibilities to a pre-defined finite set of input value combinations specified according to the hypothesized situations, the simulation analysis becomes restricted to particular small regions of the uncertainty space, thereby curbing the reach of its insights and conclusions.

The concern for explainability does not only involve traditional simulation techniques but has also recently reemerged along with the advent and proliferation of Artificial Intelligence (AI) and Machine Learning (ML) technologies, most of them likewise developed in a 'black-box' fashion [14]. Accounting for explainability has been increasingly regarded as the solution to improve model transparency, trust, and reliance, especially those integrated within decision-making frameworks [15]. In this context, SHapley Additive exPlanations (SHAP) values have been used to mitigate the lack of model explanation within AI/ML systems [16]–[18]. Originally proposed within the cooperative game theory field, the SHAP value method provides a systematic framework for quantifying the individual contribution or impact that each input variable has on the output(s), thereby enhancing the understanding of the associated interactions and, ultimately, the overall explainability of a given arbitrary model.

In this paper, we propose to combine simulation metamodeling and SHAP values analysis to address the issue frequently posed by black-box ATM simulators. Whereas metamodeling provides explicit functional approximations mimicking the behavior of the simulators, the SHAP-based analysis delivers a systematic framework for improving their explainability, thereby enhancing the overall understanding of the interactions among the variables of interest. We illustrate the methodology using a state-of-the-art ATM simulator across two case studies in which two delay-centered performance metrics are analyzed w.r.t. seven input variables. The results show that the proposed methodology can effectively make simulation and its results more explainable, facilitating the interpretation of obtained associated emergent behavior and additionally opening new opportunities for novel performance assessment processes in ATM.

[2]A SESAR Solution represents a change in the way air traffic management is performed. Solutions are new operational concepts, procedures, and relevant technologies.

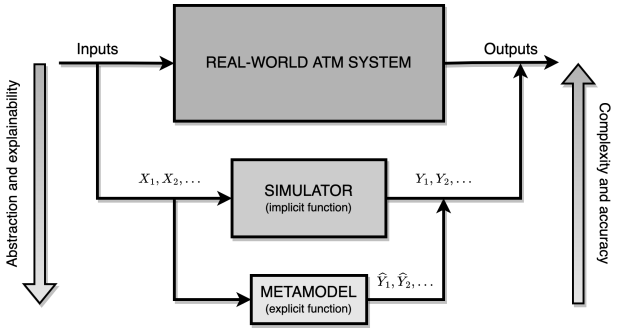

Figure 1. Relationship between the real-world system under study, the simulator, and the metamodel.

II. Background

A. Metamodeling

By definition, a metamodel is a model of a model. Although the term itself is relatively vague, having different meanings and interpretations across fields, in this paper, we focus on fast-time simulation metamodels [8], [9], that is to say, models specially designed to reproduce the behavior of simulation models. If a simulation model corresponds to an abstraction of a particular real-world system or phenomena, a metamodel[3] can be regarded as an abstraction of the simulation model itself, as depicted in Figure 1.

Simulation metamodels are any type of model that can be used to deduce the unknown input-output mapping inherently defined by the simulation model. Whereas the functional structures of the former are generally known and analytically defined, the same does not hold true for most simulators. Remember that although the average arbitrary simulator is oftentimes comprised of a plethora of internal analytic expressions and logic models, it can be externally treated as a single 'black-box' function with no concrete mathematical formula. Nevertheless, the 'emergent behavior,' resulting from its inner interactions and dynamics that evolve over time, can be directly observed. Metamodels aim at mimicking precisely this output behavior as an explicit function of the simulation inputs, therefore contributing to a better explainability of the underlying simulator and, consequently, the problem under study.

Within the related literature, it is common to use Linear Regression and Gaussian Processes as the underlying frameworks for metamodeling [10], [19], but we create our metamodel using a Gradient Boosting Machine (GBM) [20], as it is computationally more efficient and achieves high accuracy. A GBM is an ensemble of weak predictors, in our case, decision trees, used for regression and classifications in machine learning with great success [21]. The model is built in a stage-wise fashion similar to other ensemble methods, e.g, AdaBoost or Random Forest. Let $h_m(x; a_m)$ denote the small regression tree with hyperparameters a_m, then the GBM is given as

[3]For simplicity, we use the terms 'simulation metamodel' and 'metamodel' interchangeably.

$$F\left(x;\{\beta_m,a_m\}_{m=1}^{M}\right)=\sum_{m=1}^{M}\beta_m h_m\left(x;a_m\right),$$

where x is the input variable, β_m is the learning rate, and M is the number of boosting steps. The optimal solution is then given by the function minimizing the expected loss of all samples (x, y) as

$$F^*=\arg\min_F \mathbb{E}_{x,y}[L(y,F(x;\{\beta_m,a_m\}_{m=1}^{M})],$$

where $L(y, F(x; \{\beta_m, a_m\}_{m=1}^{M}))$ is the loss function, e.g., Root Mean Square Error (RMSE), and y is the target to x.

We use the framework XGBoost [22], which efficiently implements a GBM with decision trees. XGBoost uses a depth-wise tree growth strategy and builds trees until a certain depth. It leverages the speed of GBM by implementing the algorithm such that it can compute the splits of the regression trees in parallel. XGBoost also uses several tricks to avoid overfitting, e.g., with L2-regularization [23] or the learning rate as a shrinkage parameter.

B. SHAP Values

SHapley Additive exPlanations (SHAP) is a unified framework for interpreting model predictions [16]. To explain a complex predictive model, SHAP constructs a simpler explanation model with N simplified input variables. SHAP builds upon additive variables attribution methods, which have an explanation model g that is a linear combination of binary variables, i.e.,

$$g(x)=\phi_0+\sum_{i=1}^{N}\phi_i x_i,$$

where $x_i \in \{0,1\}^N$ and $\phi_i \in \mathcal{R}$. The method gives a unique solution and has the two following properties:

- Consistency and local accuracy: The output of the explanation model is equal to the output of the predictive model, and if a variable has a positive impact on the prediction in the explainable model, it will not have a smaller impact in the predictive model.
- Additivity of explanations: The sum of all the variable attributions gives the output of the predictive model. This enables aggregating variable contributions, e.g., if a categorical variable is one-hot encoded[4], each class' contribution can afterward be summed together while still being consistent with the predictive model.

C. Simulation setup

Mercury [6] is a simulator developed over several years, able to produce detailed network-wide performance assessments, in particular regarding passenger mobility in Europe. Mercury is implemented as an event-driven simulator. The underlying model can be seen as a Monte Carlo simulation, sampling distributions (delays, missed connections, etc.) based on causal rules representing actual processes of the air transportation system (e.g. if passenger delay is bigger than a given threshold, an airline incurs costs for compensation and assistance to the passenger). It is also an agent-based model, modeling different stakeholders and components of the system as agents. The agents' memories are private, i.e., they have attributes that cannot be accessed by other agents.

The simulator models the movements of individual aircraft throughout one day of operation, including turnaround processes, tracking the passengers on board, as well as passenger connections. The passengers in Mercury are modeled through passenger processes, simulating, for instance, connecting times for individual passengers, connecting options, etc. The flight is described in terms of the time it takes to complete different processes (turn-around, taxi, take-off, cruise, etc.). The simulation of the en-route phase is approximated using sets of historical flight plans and delay distributions. As already mentioned, different types of agents are present in the system, sometimes instantiated multiple times (e.g. airline operating center or flight), sometimes once (e.g. Network Manager). The most important agents are: Airline Operating Centers (AOC), Flights, Airports, DMANs and AMANs, and Network Manager. Passengers are not modeled explicitly (i.e. as agents), but are bundled in groups that share common itineraries, and these groups are handled by the other agents, specifically AOC and the airport. These groups are dynamic, i.e., in case of missed connections they can be split if their respective passengers need to board different planes. Mercury does not model the airspace explicitly and as such does not track sectors a flight is traversing. Furthermore, no tactical air traffic control takes place. Instead, the distribution of "delays" is used to modify the flight times between navigation points, extracted from historical data, as an approximation of ATM actions.

1) Assessing SESAR Solutions using Mercury: Here, we explore the impact of two particular SESAR Solutions on the small-scale ATM system. These Solutions are used as case studies in the paper: 1) User Driven Prioritization Process (UDPP), and 2) Extended - Arrival Manager (E-AMAN).

UDPP The areas of the European air traffic network that are (likely to be) stressed due to high levels of traffic can receive an ATFM regulation, i.e., a restriction on the number of flights that can enter a given volume of airspace, or airport in a given time. The flights that were supposed to cross this zone during the regulation duration are then explicitly assigned ATFM slots, which imply departure delay. The way flights are currently assigned to ATFM slots is using the so-called "First-Planned First served" CASA algorithm (FPFS) [24]. The UDPP lets the airlines swap their own slots. Whilst swapping is efficient in some cases, it lacks the capacity to find good solutions when airlines have a small number of flights in the regulation (the issue of so-called low-volume users). The natural extension would be to allow for inter-airline slot swapping, which is the UDPP extension implemented in Mercury, the goal being to test the reduction of delay and offer airlines flexibility in managing flight and passenger delays and related costs. **E-AMAN** The aim of AMAN is to sequence

[4]One-hot encoding is a process by which categorical variables are converted to binary ones

arriving flights, making sure that safety is respected but also that the runway throughput is as high as possible, given the capacity. Since runway capacity cannot be infringed, traffic overload translates into delay, usually in a holding stack, where flights wait for a (runway) slot to land. The E-AMAN agent in Mercury manages an explicit queue of slots for the airport. The algorithm optimizes the arrival sequence between the planning and the tactical horizon, considering a particular objective function. When a flight enters the planning horizon, all flights present between the planning and the execution horizon are re-optimized, i.e., assigned to the slots which are either planned or available, considering a given optimization function. The optimization of all the flights within the E-AMAN every time a flight enters or exits the system ensures that the best sequence is maintained within the arrival manager with respect to the optimization function and that a flight may slow down to absorb part of the delay. When a flight enters the planning horizon, it receives the amount of delay that it is expected to experience and tries to absorb as much delay as possible by slowing down, saving some fuel (we term it planned absorbed delay). At the tactical horizon, a flight will be issued with a slot (assigned as the output of another re-optimization), and the required delay (if any) will be performed as holding.

III. Methodology

As mentioned previously, we propose to integrate metamodeling and SHAP-based analysis for enhanced explainability of ATM simulators for performance assessment. Figure 2 provides an overview of the proposed approach.

The first step is trivially to define the object of study, i.e., a particular ATM system, along with a problem statement and set of features (e.g., SESAR Solutions) to be evaluated, eventually leading to the specification of our two case studies, which are then run in the simulator. The Mercury design allows for an easy switch between the two case studies, which in turn are encoded as input parameters. The metamodel is then used to provide an approximate explicit functional relationship between the input variables and the KPIs of interest for the case study. After the metamodel is fitted to a data set comprised of pre-computed simulation results, the SHAP method is applied to improve its explainability and, by proxy, the transparency of the underlying simulator. We deem this metamodel as an 'explainable metamodel' as it aims to better explain the relationships between the variables and KPIs, which ultimately provides an additional level of understanding regarding the problem being addressed.

A. Experimental Setup

The simulation used in the paper runs the day of operations at Charles De Gaulle airport, based on the historical data from 12 September 2014: flights, origin-destination, routes, aircraft types, estimated cruise wind, distributions on climb and descent profiles, requested nominal cruise speeds and flight levels, companies, airspace structure, ATFM regulations, minimum connecting times, minimum turnaround times (sourced from EUROCONTROL's Demand Data Repository),

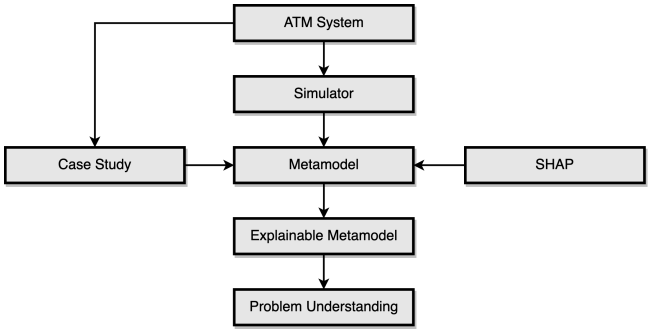

Figure 2. Methodology Overview.

TABLE I. Variables used in the case studies.

Variable	Description	Theoretical range	Practical range	Default
Fuel price	Price of one kg of fuel.	$[0, \infty)$	[0, 5]	1
Hotspot solver	Type of solver in the hotspot.	[GlobalOpt, NNBound, UDPP, ISTOP]	NA	ISTOP
Planning horizon	Distance horizon where the EAMA tries to optimize the arrival, in NM.	$(100, \infty)$	[100, 1000]	300
Cruise uncertainty	Deviation in the aircraft speed during cruise.	$[0, \infty)$	[0, 10]	1
Turn-around time scale	Scaler of mean of the distribution of turn-around times.	$[0, \infty)$	[0, 10]	1
Minimum connecting time scale	Scaler of mean of the distribution of passenger minimum connecting times.	$[0, \infty)$	[0, 10]	1
Claim rate	Proportion of passengers claiming compensation.	[0, 1]	[0, 1]	0.14

cost of delay values [25], taxi times (from International Air Transport Association summer 2010 report), non-ATFM delays (from CODA databse), passenger itineraries, fares and alliances (from Paxis), schedule departure/arrival times (from Innovata). In this simulation, all the flights to and from Charles De Gaulle are simulated with all corresponding information on passenger connections and turnaround. Furthermore, the ATFM regulations are inflated to be able to test UDPP mechanisms (i.e. case study 1).

In order to find interesting relationships between variables, those that are harder to model, and thus more interesting from the metamodelling point of view, we selected the input variables shown in Table I, with a short description, theoretical and practical range, and a default value. The practical ranges are defined to limit the scope of the case studies but are wide enough to capture the effects of high values, e.g., in theory, there is no maximum value for the fuel price, but in practice, we believe there is some upper limit.

As can be noted, a large number of input variables and output KPIs are available from the simulator, and not all the relationships and emergent behaviors can be shown here due to space restrictions. However, to showcase our methodology, for each case study, we chose to explore and present relationships between one output KPI and all input variables.

TABLE II. HYPERPARAMETERS OPTIMIZED WITH A GRID SEARCH.

Hyperparameter	Space
Max depth	[3, 4, 5, 6, 7, 8, 9]
L2-regularization	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
Learning rate	[0.01, 0.05, 0.1, 0.2, 0.3, 0.4]

B. UDPP Case Study

As mentioned, the goal of UDPP Solution is to reduce delays and offer airlines flexibility in managing flight and passenger delays and related costs. Flight and passenger delays are not the same when there are connecting passengers. Thus, we want to explore the effect of input variables on "Passenger arrival delay" which represents passenger arrival delay to the final destination. The UDPP mechanisms allow airlines to minimize passenger delay, as they take into account the costs that would be due for compensation and duty of care for large delays.

C. E-AMAN Case Study

E-AMAN strives to sequence flights, further away from the airport (we test up to 1000 NM), with the goal to transfer the planned delay from holding to the en-route phase of the flight. Here, we explore the KPI "Planned absorbed delay" which denotes how much of the delay flights manage to absorb during the en-route portion of the flight. Absorbing delay in the cruise phase consumes less fuel but is limited to the aircraft performance and the flight cost strategies used by airlines.

D. Optimizing the metamodel

Following the general approach in the machine learning field, we optimize the hyperparameters of each metamodel using the classical train-validation-test split to get a good generalization performance for each of the KPIs. We generate a training data set with 50k simulations and a test data set with 10k simulations, both created with Latin hypercube sampling to cover the full variable space. We randomly split the train data set into a smaller training data set with 40k simulations and a validation data set with 10k simulations, such that we can tune the hyperparameters of XGBoost with a grid search. We optimize the maximum depth of each decision tree, the L2 regularization, and the learning rate, using the grid search and 10-fold cross-validation splits. The hyperparameters and their search space are listed in Table II. Additionally, we use the exact greedy algorithm to choose the splits in the decision trees. For all the models, we set the maximum number of boosting steps to 1000 and apply early stopping if the validation accuracy does not improve for five steps. We evaluate the accuracy of XGBoost by comparing the Root Mean Square Error (RMSE) and the Root Relative Square Error (RRSE) with a baseline model, which predicts the average value of the training set.

IV. RESULTS

We first open the simulation box by using the explainable metamodel to investigate the KPIs. Then, we report the model performance together with the used hyperparameters, along with several computational considerations.

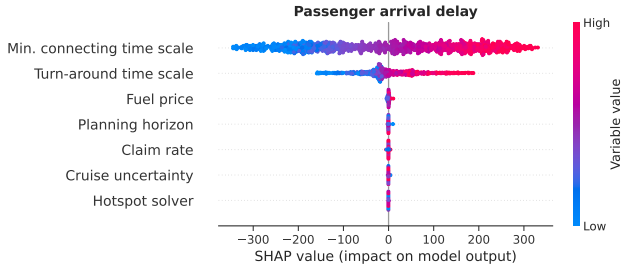

Figure 3. Variable impact on passenger arrival delay.

A. UDPP Case Study

In this section, the KPI *passenger arrival delay* is explored, using the SHAP values to explain the metamodel's - and thus the simulator's - predictions, and afterward use the metamodel to investigate different scenarios, shedding light on interactions between variables relevant for passenger arrival delay.

Figure 3 shows the importance of variables (given by the SHAP values) in predicting the passenger arrival delay.

On the y-axis, the variables are sorted in descending order, such that the minimum connecting time scale[5] is the variable with the largest impact on the metamodel's output, and the choice of hotspot solver has the smallest impact on the metamodel's output. On the x-axis, the SHAP value (the impact on the model output) is shown. Lastly, the color bar denotes the value of the variables. Thus, the graph shows that the two most important variables are the minimum connecting time scale (MCT) and turn-around time scale (TAT), where the higher MCT and TAT (red points), the higher delay (positive SHAP values). Moreover, the SHAP values for the MCT seem to be equally spread and turning from blue to red, going from left to right. This suggests a linear connection between the MCT and the passenger arrival delay. For the TAT, there are more points centered around smaller SHAP values, suggesting a non-linear relationship between TAT and the passenger arrival delay. Both of these patterns can be further explored with the SHAP values and with predictions from the metamodel.

In Figure 4, there are two interaction plots based on the SHAP values. In the left plot, we have the minimum connecting time scale (MCT) on the x-axis and the SHAP values for MCT on the y-axis. The colors denote the value of the feature turn-around time scale, such that a blue point and red point correspond to a low and high turn-around time scale, respectively. Irrelevant of the colors, we see that the higher MCT, the higher the SHAP value, meaning that the higher MCT, the more it contributes to a higher passenger arrival delay. The colors indicate the value of the turn-around time scale (TAT), showing that if TAT is low, there is a more

[5]Mercury samples both minimum connecting times and turn-around times from theoretical distributions. The scale effectively changes the shape of the distribution from which the sampling is performed by multiplying the mean of the distribution by this parameter.

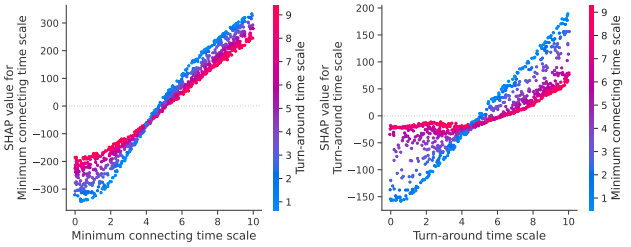

Figure 4. SHAP interaction plots for minimum connecting and turn-around time scales.

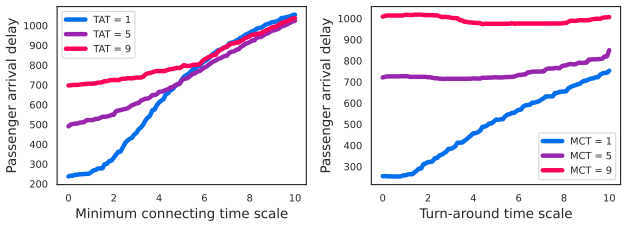

Figure 5. Metamodel predictions of passenger arrival delay for different minimum connecting and turn-around time scales.

considerable change in the passenger arrival delay when MCT is increased compared to the case where TAT is high. In other words, if TAT is high, the value of MCT has a smaller impact on the passenger arrival delay. We investigate the same interaction in the right plot, now having TAT on the x-axis. When MCT is low, the increase in the passenger arrival delay is almost linear with respect to TAT. On the other hand, when MCT is high, the value of TAT affects the passenger arrival delay less.

The interaction plots based on the SHAP values give a good interpretation of the metamodel and, consequently, of the simulator, but they do not say anything about the actual output of the simulator. Instead, we can use the metamodel directly to predict the simulator's output in different settings. We investigate how MCT and TAT affect the passenger arrival delay when all the other variables are set to their default values, cf. Table I. Since we already know there is an interaction between MCT and TAT, we explore both variables for a low, medium, and high value for TAT and MCT, respectively. In Figure 5, the left plot shows that the passenger arrival delay increases when MCT increases.

When TAT is low (blue line), the passenger arrival delay increases from 200 to 1000 as MCT is increased from 0 to 10. If TAT is high (red line), the passenger arrival delay is already high for low values of MCT, and for MCT higher than six, the passenger arrival delay is only slightly affected by TAT. In Figure 5, the right plot shows that for a high MCT, the passenger arrival delay is around 1000 and is not affected by TAT. Conversely, when MCT is low, TAT has a high effect on the passenger arrival delay. Overall, these relationships are to be expected. It is interesting that MCT and TAT play the most important roles in passenger delay, even in UDPP processes that allow airlines to reduce this KPI, albeit based on the cost

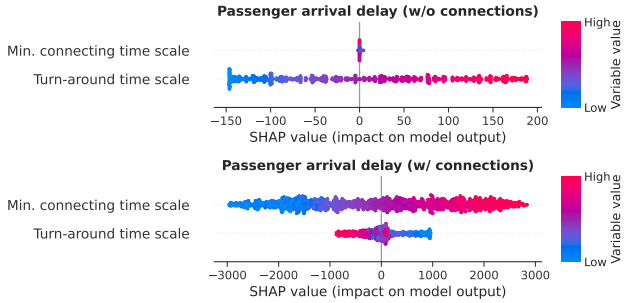

Figure 6. The impact of minimum connecting and turn-around time scales on arrival delay for passengers with and without connections.

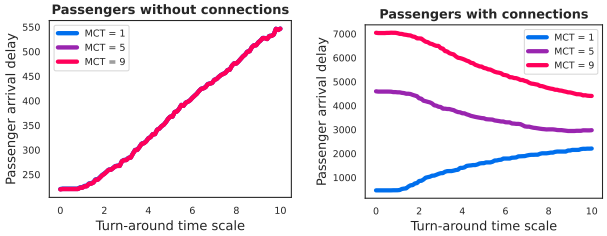

Figure 7. Metamodel predictions of arrival delay for passengers with and without connection for different minimum connecting and turn-around time scales.

of delay.

1) Passenger arrival delay with and without connections: The passenger arrival delay consists of the average passenger arrival delay for all types of passengers. However, we can analyze the KPI for passengers with and without connections. We focus on the impact of MCT and TAT since other variables are less important, cf. Figure 3. In Figure 6, we see two things: 1) the minimum connecting time scale (MCT) has almost no impact on the delay for passengers without connections, whereas it is essential for passengers with connections, and 2) a higher turn-around time scale increases the delay for passengers without connections but seems to have the opposite effect on passengers with connections.

In Figure 7, we investigate the different effects of the turn-around time scale (TAT) further by using the metamodel to predict the arrival delay for passengers with and without connections.

We predict the delay given TAT for a low, medium, and high value of MCT, and with the rest of the variables fixed to default values. The effect of TAT is the same across the values of MCT and has a positive linear correlation, except for values below 1, with the arrival delay for passengers without connections. For connecting passengers, the TAT interacts with the MCT: if the MCT is low, the delay is smallest for low TAT, but if it is high, the delay is smallest for high values of TAT.

B. E-AMAN Case Study

In this section, we follow the same procedure as for the first KPI and give a second example of how SHAP values, together with a metamodel can be used to uncover the simulator's behavior.

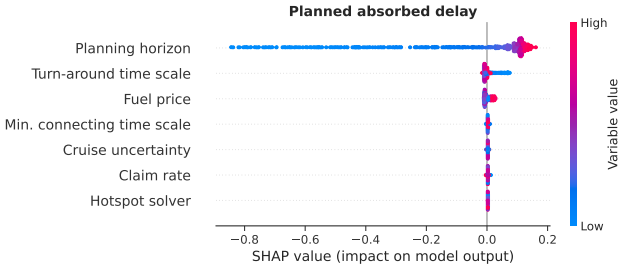

Figure 8. Variable impact on passenger arrival delay.

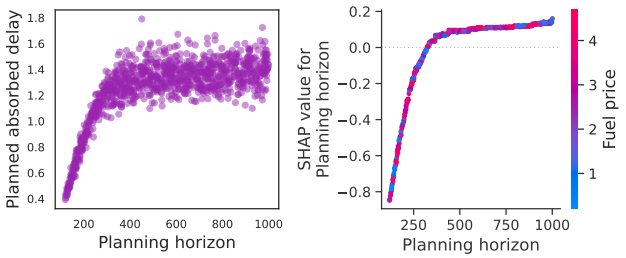

Figure 9. Investigating the effect of the planning horizon on the planned absorbed delay. Left: The planned absorbed delay given the planning horizon. Right: The SHAP values for the planning horizon given the planning horizon.

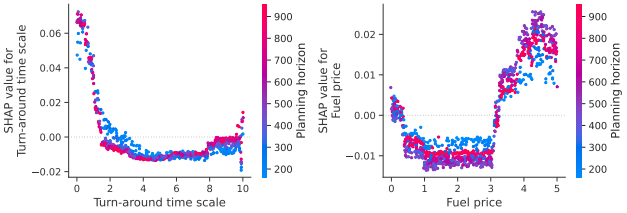

Figure 10. SHAP interaction plots for turn-around time scale and fuel price.

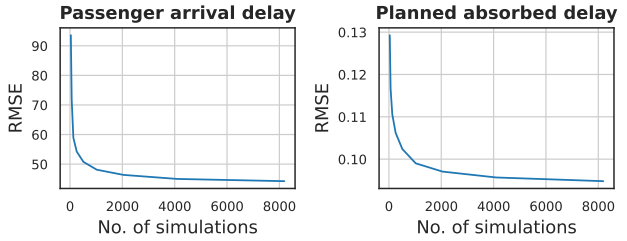

Figure 11. Predictive accuracy averaged across 10 runs for the metamodel trained on different numbers of simulations.

In Figure 8, it is seen that the planning horizon is by far the variable with the biggest impact on the planned absorbed delay, which is to be expected. The larger the planning horizon, the flight has more time and space to absorb the larger amounts of delay. The turn-around time scale (TAT) and fuel price have smaller but notable effects. Simply using the training data and plotting the planned absorbed delay as a function of the planning horizon - see the left plot in Figure 9 - we see that the planned absorbed delay increases when the planning horizon is increased from 100 to 400, but increasing it further has no effect on the planned absorbed delay. For a planning horizon higher than 400, there also seems to be a constant trend. Though this plot shows the general trend, it can not uncover any potential interactions. In the right plot in Figure 9, we see almost no variation in the SHAP value for the planning horizon for different planning horizon values; thus, there are no interactions with other variables. This means that the effect of changing the planning horizon is unaffected by changes in other variables.

The second and third most important variables are the turn-around time scale and fuel price. In the left plot in Figure 10, we see that a turn-around time scale (TAT) below two contributes to a higher planned absorbed delay compared to a TAT above two.

We also see a minor interaction with the planning horizon, where, e.g., a TAT between 2-4 gives a higher planned absorbed delay if the planning horizon is low compared to if it was high. In the right plot in Figure 10, we see that changing the fuel price in the range 1-3 has no contribution to the predicted planned absorbed delay. However, out of this range, the fuel price notably affects the model output. Most prominent is the jump in the SHAP value for a fuel price above 3. Lastly, we see interaction with the planning horizon, showing that the fuel price has, in general, the most significant effect on the prediction of the planned absorbed delay when the planning horizon is around 500-600, whereas a planning horizon around 100-200 has the smallest effect.

C. *Performance of metamodel*

The performance of the metamodels is given in Table III. For both KPIs, the metamodel achieves lower RMSE and RRSE, and is significantly better than the baseline models. On a CPU Threadripper 3960X, the grid search for XGBoost is parallelized and takes less than 10 minutes. With the tuned parameters, the training time is less than five seconds. Both times can be considered insignificant compared to the computational burden of acquiring the 50k simulations for training and validation.

D. *Computational considerations*

In practice, it is infeasible to create a data set with 50k simulations because of the computational burden of the simulator. One solution would be to employ an active learning [26] metamodel, which can balance the trade-off between model accuracy and the number of simulations needed, as seen in [10]. In short, the active learning metamodel iteratively augments a small initial data set by running new simulations that are selected to increase the performance of the metamodel. We leave the integration of explainable and active learning metamodels for future work. Here, we show that with smaller training data sets, the metamodel still reasonably performs well. In Figure 11, it is seen how the performance gain decreases as more data is added, and with only 2000 simulations, the metamodels perform almost equally well as those trained on 40k simulations in Table III. We believe that with active learning metamodels, the number of simulations needed to achieve a good performance can be reduced even further.

TABLE III. PERFORMANCE OF THE METAMODELS.

	RMSE	RRSE	Hyperparameters
Passenger arrival delay			
Baseline	202.3	1.00	-
XGBoost	43.0	0.21	Max depth = 5, learning rate = 0.05, L2-reg. = 0.4
Passenger arrival delay w/o connections			
Baseline	115.5	1.00	-
XGBoost	43.4	0.38	Max depth = 3, learning rate = 0.05, L2-reg. = 0.1
Passenger arrival delay w/ connections			
Baseline	1690.3	1.00	-
XGBoost	112.8	0.07	Max depth = 5, learning rate = 0.01, L2-reg. = 0.5
Planned absorbed delay			
Baseline	0.242	1.00	-
XGBoost	0.094	0.39	Max depth = 3, learning rate = 0.1, L2-reg. = 0.5

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a unified modeling framework that integrates SHAP values with simulation metamodels to create explainable metamodels, uncovering the intricate hidden relationships among the input and output variables of a simulated ATM system in a rather practical way.

Using the state-of-the-art ATM simulator Mercury, we conducted two case studies exploring two delay-centered performance metrics w.r.t seven variables, covering exogenous and endogenous variables. We created an explainable metamodel using XGBoost with SHAP values and showed how such a metamodel could be used as a systematic framework to analyze the simulator, thereby enhancing the overall understanding of the interactions among the variables. The case studies show that the proposed methodology can effectively make simulation and its results more explainable, facilitating the interpretation of obtained emergent behavior and opening new opportunities for novel performance assessment processes within the ATM research field. It is worth noticing that this approach is not meant, nor is it able, to completely discard traditional simulation-based analyses but rather to complement them in a constructive and meaningful way. We believe that the presented methodology can effectively enhance scenario-based and what-if analyses, greatly contributing to a more comprehensive and in-depth ATM performance assessment framework. In future work, we plan to extend the current methodology to integrate active learning as a solution to address the potential computational drawbacks associated with the generation of the data sets used for fitting the metamodels. In this way, we will be able to simultaneously tackle the lack of explainability of simulation models and the burden of running computationally expensive computer experiments.

ACKNOWLEDGEMENTS

This work was supported by the NOSTROMO (Next-generation Open-Source Tools for ATM PeRfOrmance Modelling and Optimisation) project, framed in the scope of the SESAR 2020 Exploratory Research topic SESAR-ER4-26-2019, 'ATM Validation for a Digitalised ATM,' with focus on the 'Macro-modelling applied to Air Traffic Management' area and funded by SESAR Joint Undertaking through the European Union's Horizon 2020 research and innovation programme under grant agreement No 892517.

AUTHORS CONTRIBUTION

The authors confirm their contribution to the paper as follows: study conception and design: C. Riis, F. Antunes, and T. Bolić; data collection: C. Riis and G. Gurtner; analysis and interpretation of results: all; draft manuscript preparation: C. Riis, F. Antunes, and T. Bolic. All authors reviewed the results and approved the final version of the manuscript.

REFERENCES

[1] A. Cook, *European air traffic management: principles, practice, and research*. Ashgate Publishing, Ltd., 2007.
[2] A. Cook and D. Riva, *Complexity Science in Air Traffic Management*. Routledge London, 2016.
[3] M. Phillips and D. T. Marsh, "The validation of fast-time air traffic simulations in practice," *Journal of the Operational Research Society*, vol. 51, no. 4, pp. 457–464, 2000.
[4] EUROCONTROL, "European operational concept validation methodology, e-ocvm v3," 2010.
[5] G. Gurtner, C. Bongiorno, M. Ducci, and S. Miccichè, "An empirically grounded agent based simulator for the air traffic management in the sesar scenario," *Journal of Air Transport Management*, vol. 59, pp. 26–43, 2017.
[6] L. Delgado, G. Gurtner, P. Mazzarisi, S. Zaoli, D. Valput, A. Cook, and F. Lillo, "Network-wide assessment of atm mechanisms using an agent-based model," *Journal of Air Transport Management*, vol. 95, p. 102108, 2021.
[7] A. M. Law, *Simulation Modeling and Analysis*, 5th ed. McGraw-Hill Higher Education, 2015.
[8] J. P. Kleijnen and R. G. Sargent, "A methodology for fitting and validating metamodels in simulation," *European Journal of Operational Research*, vol. 120, no. 1, pp. 14–29, 2000.
[9] R. B. Gramacy, *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. CRC Press, 2020.
[10] C. Riis, F. Antunes, G. Gurtner, F. C. Pereira, L. Delgado, and C. M. L. Azevedo, "Active learning metamodels for atm simulation modeling," *Proceedings of the 11th SESAR Innovation Days*, vol. 2021, 2021.
[11] SESAR Joint Undertaking, "Vision of the future performance research in sesar," Project PJ19 CI, Tech. Rep., 2018.
[12] ——, "European atm master plan: digitalising europe's aviation infrastructure, executive view," 2020.
[13] T. Bolić and P. Ravenhill, "Sesar: The past, present, and future of european air traffic management research," *Engineering*, vol. 7, no. 4, pp. 448–451, 2021.

[14] C. Zednik, "Solving the black box problem: a normative framework for explainable artificial intelligence," *Philosophy & technology*, vol. 34, no. 2, pp. 265–288, 2021.
[15] L. Chazette, W. Brunotte, and T. Speith, "Exploring explainability: a definition, a model, and a knowledge catalogue," in *2021 IEEE 29th international requirements engineering conference (RE)*. IEEE, 2021, pp. 197–208.
[16] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.
[17] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable ai for trees," *Nature machine intelligence*, vol. 2, no. 1, pp. 56–67, 2020.
[18] M. Sundararajan and A. Najmi, "The many shapley values for model explanation," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 9269–9278.
[19] C. Zhang, C. Osorio, and G. Flötteröd, "Efficient calibration techniques for large-scale traffic simulators," *Transportation Research Part B: Methodological*, vol. 97, pp. 214–239, 2017.
[20] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.
[21] D. Nielsen, "Tree boosting with xgboost - why does xgboost win "every" machine learning competition?" Master's thesis, NTNU, 2016.
[22] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.
[23] C. Cortes, M. Mohri, and A. Rostamizadeh, "L2 regularization for learning kernels," *arXiv preprint arXiv:1205.2653*, 2012.
[24] EUROCONTROL, "Atfm modelling capability - amoc," 1997.
[25] A. J. Cook and G. Tanner, "European airline delay cost reference values - updated and extended values (version 4.1)," Tech. Rep., 2015. [Online]. Available: https://www.eurocontrol.int/sites/default/files/publication/files/european-airline-delay-cost-reference-values-final-report-4-1.pdf
[26] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2010.