Automatic Evaluation of LS-DYNA® Simulation Results Using Statistical Database and Python

Daxin Wu[1], Olaf Hartmann[2]
[1]ARRK Engineering
Frankfurter Ring 160, 80807 Munich, Germany
[2]ARRK Engineering

Abstract

Thanks to the significant increase of computational power, full-vehicle crash simulation has become a standard procedure in vehicle crashworthiness design. It has enabled engineers to accommodate constantly shortened development period in automotive industry. On the other hand, with eased access to simulation tools, simulation results flood during the project development. Most of the time, only a small specific portion of the results is analyzed by engineers. Based on internal statistics gathered from various projects, the number of curves in history outputs from a full-vehicle crash simulation varies from 14,000 to 500,000, depending on the model complexity and conventions of the project. Nevertheless, only 0.1 to 1 percent of the entire outputs are used. The rest of the curves are not analyzed mainly due to two reasons, namely little relevance to the focus of the analysis and absence of post-processing method for systematic analysis of a large amount of outputs. However, considering the size of a full vehicle crash model and the complexity of the crash event, a lot of information can be overlooked.

Using data mining, history outputs from past simulations can be systematically gathered, processed in a statistical manner and then stored in a database to serve as a reference for future simulations and even as the basis for more advanced evaluations methods. In the scope of this work, we developed a method for automatic evaluation of history outputs of LS-DYNA simulations by comparing them with a reference database created from previous simulation results, which are predecessors or comparable with the new simulation. The comparison identifies the history outputs with the most significant deviations and records the time points, when such discrepancies occur. Furthermore, the spatial information of these history outputs are extracted, namely the positions on the vehicle. The comparison result therefore shows both the time when deviations occur and the structural regions which are most likely responsible for the deviations. This helps determine the sequence of different structural behaviors and their interdependencies. Significant deviations can come from initial differences in the model, which may indicate modelling errors, or arise over time. This tool was adopted in the crash development of a sports car in order to ensure model quality and identify the sequence of different structural behaviors and their causes.

In this paper, we present the possibility of automatic evaluation of all LS-DYNA history outputs using python. This serves as a foundation for further evaluation techniques based on big data analysis.

Introduction

As computational power keeps increasing significantly, Computer Aided Engineering (CAE) plays a more important role in product development across various industries. In the automotive industry, nowadays crashworthiness design of vehicles relies heavily on the utilization of crash simulation with virtual development models. Compared with conventional prototype based development, CAE based development has enabled the engineers to explore more possibilities with much less time and costs. Commercial simulation software such as LS-DYNA provide the possibilities to output different kinds of measurements during simulation. These measurements are output and stored in binary databases. In LS-DYNA, history outputs are stored in "binout" files. According to internal statistics, "binout" of a full-vehicle simulation consists of more than 10 different types of history outputs, which contain in total between 14,000 and 500,000 curves, depending on the model size and conventions of the project as shown in Fig. 1. Most of the time, only less than one percent of the entire history outputs are considered in analysis. This is mainly due to two reasons. Firstly, the rest of the curves are not directly related to the focus of the analysis. Secondly, processing of all curves is difficult in terms of current post-processing possibilities.

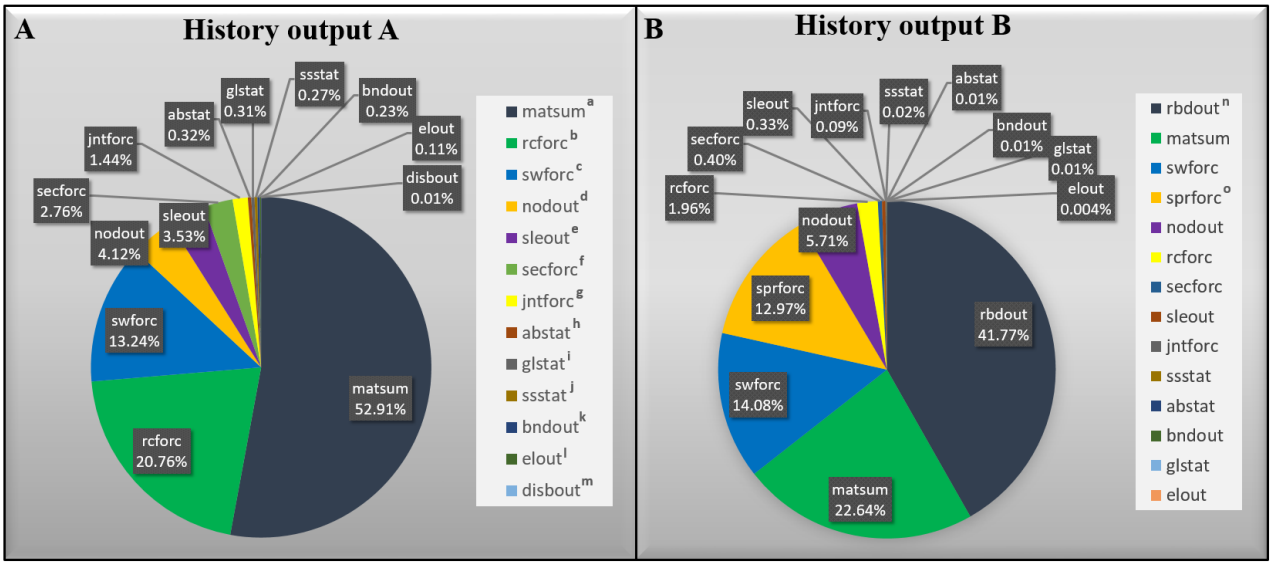

Fig. 1: Distribution of history outputs of full-vehicle crash simulations from internal statistics. **a** History output distribution of FMVSS208 full frontal crash of a two-door sports car with 14,210 outputs in total. **b** History output distribution of IIHS small overlap crash of a limousine with 482,080 outputs in total.

[a] material energies [b] contact forces [c] nodal constraint reaction forces [d] nodal point data
[e] sliding interface energy [f] cross section forces [g] joint force [h] airbag statistics
[i] global data [j] subsystem data [k] boundary condition force and energy [l] element data
[m] discrete beam element [n] rigid body data [o] SPR force

However, the current approach has two potential disadvantages. Firstly, considering the complexity of crash events, relevant information can be easily overlooked. Secondly, history outputs, which have no direct values for a specific analysis, may also possess useful information when considered in a larger scale, i.e., in a statistical manner. Currently, no post-processor provides ready-for-use functionalities for systematic processing of such a large amount of data. As pointed out by the developers of the qd-cae-python library [1], post-processors make it easier for engineers to access data, but at the meantime, also impose limitations on the freedom in terms of data manipulation. With this limitation becoming more obvious, the open-source qd-cae-python library was developed.

This library enabled direct access to LS-DYNA results with python. After this was realized, possibility of machine learning techniques with crash simulation analysis has been explored [2]. In this work, we propose a method for automatic evaluation of the all history outputs by comparing with a statistical database using qd-cae-python,

Method

The method developed in this paper enabled automatic evaluation of a new simulation result by comparing it with a statistical database generated using previous simulation results, which were considered similar with the current simulation. The aim of the comparison was to identify the significant differences between current simulation and its predecessors. In general, differences in the history output correspond to differences of structural behaviors and is fundamentally caused by model changes. In principle, the logic of this causality is rather straight forward, namely, model changes lead to different structural behaviors, which then result in different history outputs. Nevertheless, without a designated post-processing method, it is often extremely difficult to determine these correlations, considering that crash phenomenon is rather a sequence of chain events with complex interdependencies than uncoupled individual events. In this method, after the differences between a new simulation and the database were identified, the temporal and spatial information of these significant differences were recorded. The visualization of the time and spatial clustering of differences enabled further analysis of the effects of model changes. The method was realized with three steps.

- **Creation of a statistical database**
- **Comparison of simulation result with the statistical database**
- **Visualization of the comparison results**

For the creation of a statistical database, the qd-cae-python library was used for extracting the history output data from a "binout" file. Taking history output "secforc" [4] as an example, the contents of "binout" files and the data structure used in qd-cae-python are explained in Fig. 2.

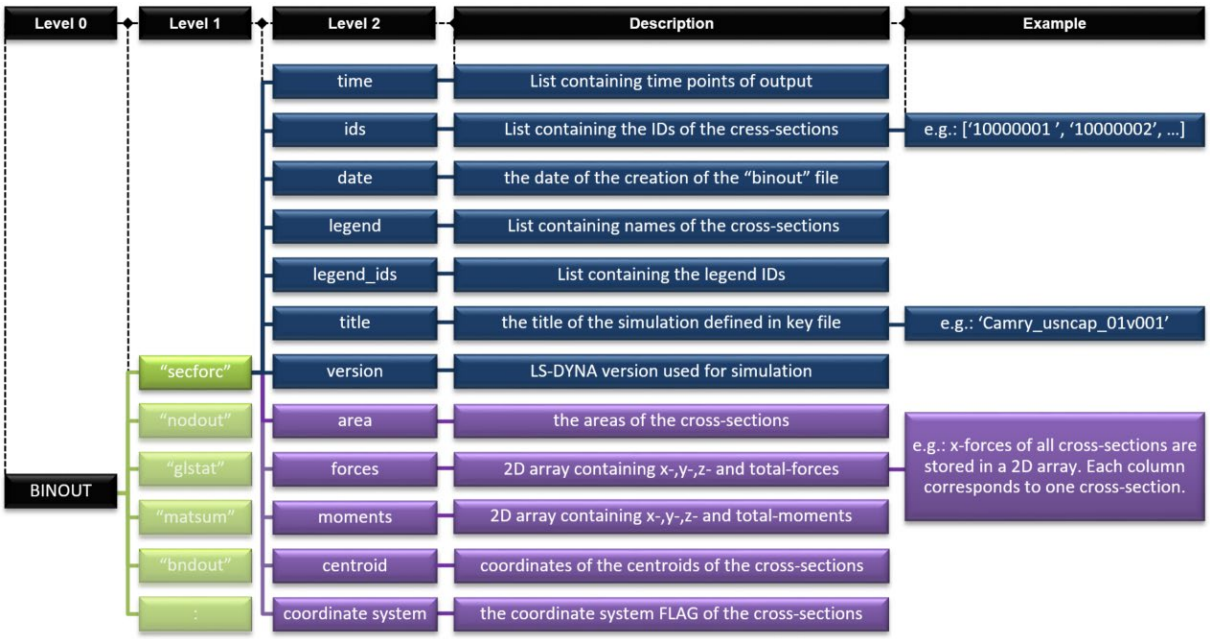

Fig. 2: Content of database "secforc" and the data structure used in qd-cae-python.

A "binout" file (Level 0 in Fig.2) contains several different types of outputs (Level 1). The database "secforc" contains the measurements of cross-sections defined using *DATABASE_CROSS_SECTION in LS-DYNA. In each database, other than numerical outputs, system information such as the LS-DYNA version used for simulation were also contained. The data (Level 2) in databases can be divided into two types, namely common data type for all databases (dark blue) and unique data for a specific database (purple). For instance, while all databases contain output time sequence data, only "secforc" measures the areas of the cross-sections.

Before the data were stored, the contents of the data were checked in order to filter out all non-numerical data. After the curves of each individual simulation were extracted from the "binout" file, they were stored in a dictionary in python. While the keys contained the information of database type and ID, the history output data were stored as the values of the corresponding keys. With this approach, dictionaries all had the same structure and therefore yielded the possibility of cross-dictionary comparison and mathematical operations. This laid the foundation for the creation and the maintenance of the database.

The statistical database consisted of the cross-simulation mean and standard deviation (SD) of sampled simulation results. As illustrated in Fig. 3, the cross-simulation mean and standard deviation were calculated and updated using Eq. 1 and Eq. 2 respectively, when new results were added to database. In Eqs. 1-2, $\bar{x}$ indicates the mean, n corresponds to the number of simulation samples in the database and σ is the standard deviation.

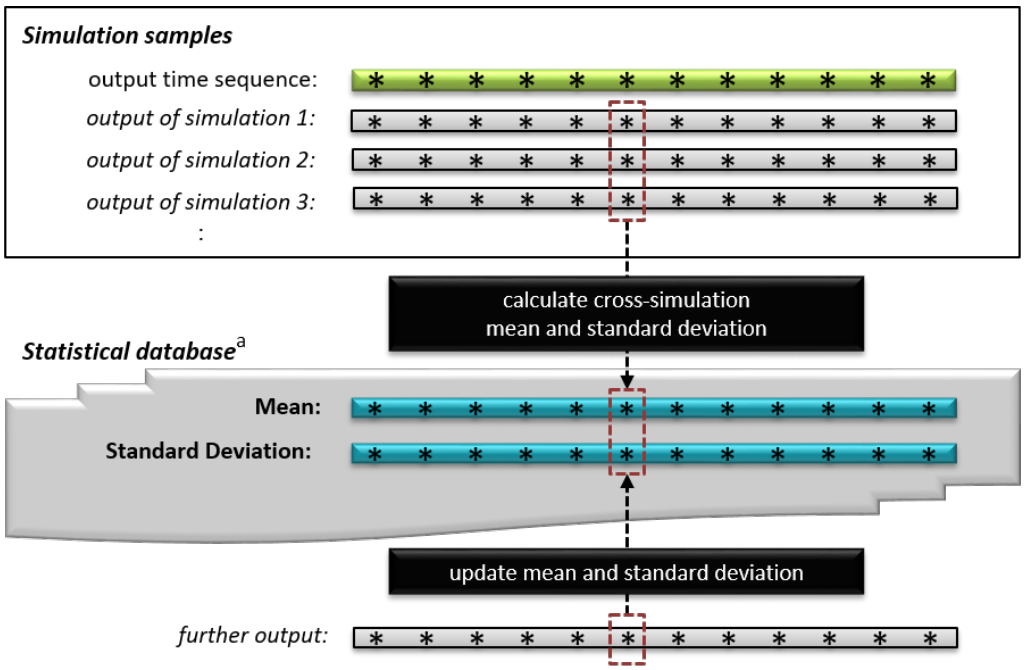

Fig. 3: Procedure of creation and update of statistical database with samples of simulation results.
[a] Statistical database contains cross-simulation mean and standard deviation.

$$\bar{x}_{n+1} = \frac{1}{n+1}(n\bar{x}_n + x_{n+1}) \quad \textbf{(Eq. 1)}$$

$$\sigma^2_{n+1} = \frac{n}{n+1}\left(\sigma^2_n + \frac{(x_{n+1}-\bar{x}_n)^2}{n+1}\right) \quad \textbf{(Eq. 2)}$$

After the database was created, in order to detect the significant differences of history outputs between the new simulation and the database, the differences between the new simulation and the mean in database were calculated at each output time points. These differences were then checked with two criteria at each time point. Firstly, if the difference at the current time point was greater than the local standard deviation. Secondly, if the difference had a certain significance on the global scale. The second criterion was verified by comparing the difference at the current time point with the maximum amplitude of the entire curve as shown in Fig. 4. The mean curve from the database was denoted as "glb_mean" and the interval on its both sides was equal to the standard deviation at each output time point. The region covered by the "interval" corresponded to the area of mean ± standard deviation. In the green region of the curve, the first criterion was fulfilled. However, the differences in this region were negligible considering their magnitudes compared with the maximum amplitude of the curve. On the contrary, the differences in the purple region fulfilled both of the criteria. In this study, ten percent of the maximum amplitude of the curve were used as the threshold value for checking the second condition during comparison.

With the current version of the method, the history outputs were not filtered before comparison. Curves of extremely high frequencies, for instance the x-acceleration output of an accelerometer fixed on the B-pillar from a full-frontal crash simulation (Fig. 5), led to unmeaning comparison results. Outputs of crash simulations are sometimes filtered with channel frequency classes (CFC) filters in post-processing phase in order to eliminate unphysical signals. In this study, the curves of extreme high frequencies were detected using Fast Fourier Transformation (FFT) and excluded from the analysis.

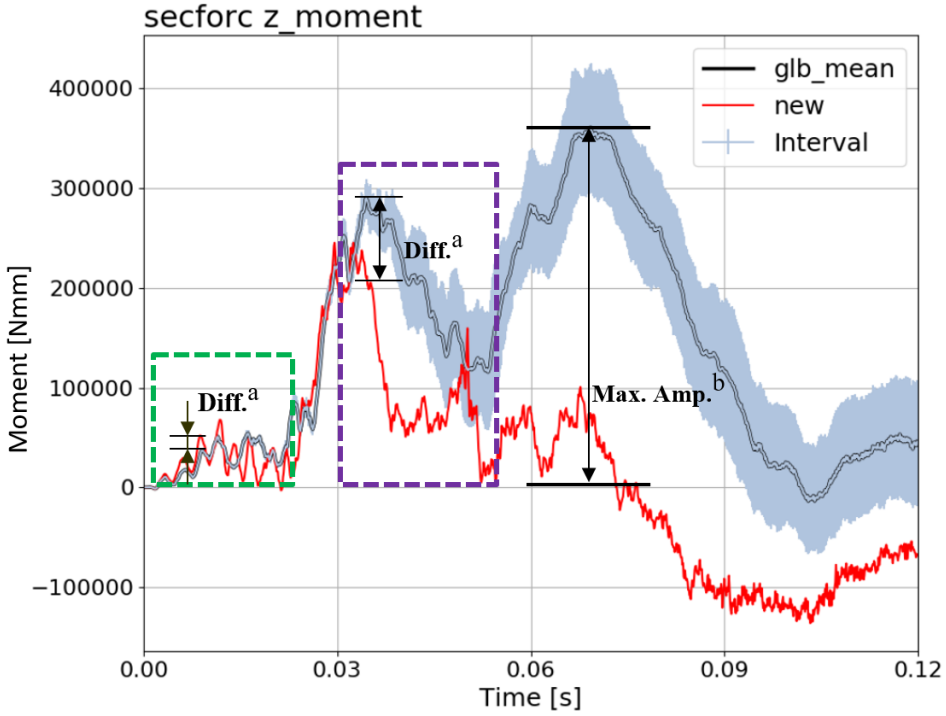

Fig. 4: Comparison of new simulation result with the statistical database using two criteria for identifying significant differences. [a] Differences between the newresult and the mean. [b] Maximum amplitude of the mean curve.

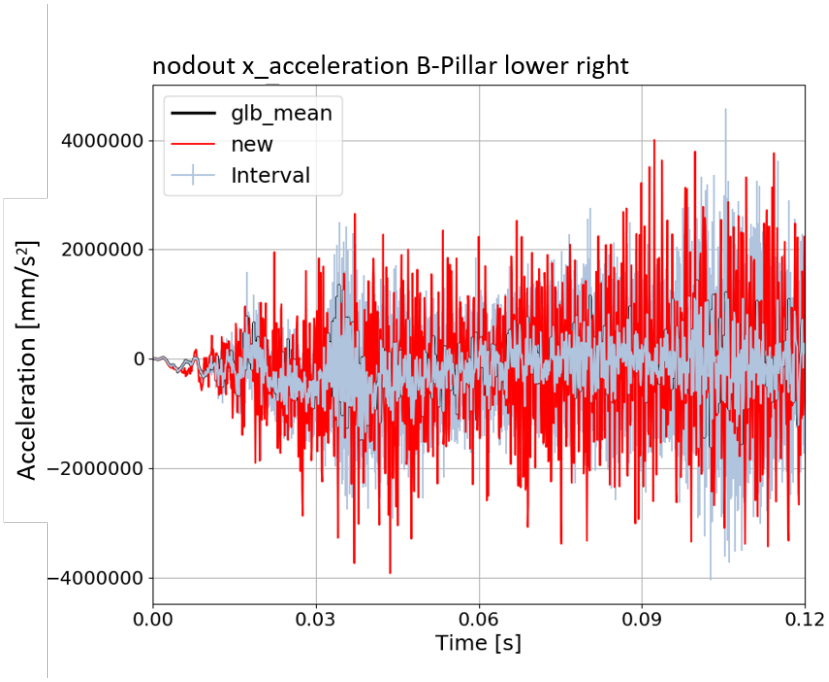

Fig. 5: Unfiltered acceleration measurement in x-direction from an accelerometer fixed on the right B-pillar in the lower position.

After a significant difference was recognized in a curve, the name of the curve, the time point when it occurred were recorded in a list. Furthermore, the magnitude of the difference itself and the ratio of difference to standard deviation at this time point were appended to the list for visualization purpose. The list was then sorted according to the magnitudes of the differences. After sorting, history outputs of the same type and ID, which appeared multiple times in the list were reduced to once. For instance, if a cross-section reported significant difference in force, the moment output of the same cross-section would also highly likely be significantly different. However, only one entry with the greatest difference was kept in the list. A scatter plot was then created with the time points of the significant differences as x-axis and the difference to standard deviation ratio as y-axis.

After the creation of time clustering, the corresponding spatial information were extracted from the same list, using the database type and the ID information in the name of the curve. For the visualization of the spatial information of these significant differences, GNS Animator v2.4.3 was used. The python API adopted since Animator v2.4.1 [4] was utilized to integrate the spatial visualization module.

Results

The method proposed in this paper was developed and used in the development process of a sports car. For demonstration purpose, non-confidential simulation results were prepared. In this case, the US New Car Assessment Program (USNCAP) full-frontal crash simulations with the open-source Toyota Camry LS-DYNA model [5] were performed as shown in Fig. 6. In this study, only structural behaviors were under consideration, therefore, no crash dummy models were included in the simulation model. Differences between all simulation models prepared were mainly minor thickness changes based on engineering judgment aimed at improving the crash performance. In total thirteen simulations were selected for the purpose of the demonstration of this method. A statistical database was created using the results of the first twelve simulations. Theoretically, there is no limitation on the number of simulation samples with this method. An automatic evaluation of the last simulation

result compared with the database was then performed and the temporal and spatial distributions of the significant differences were plotted.

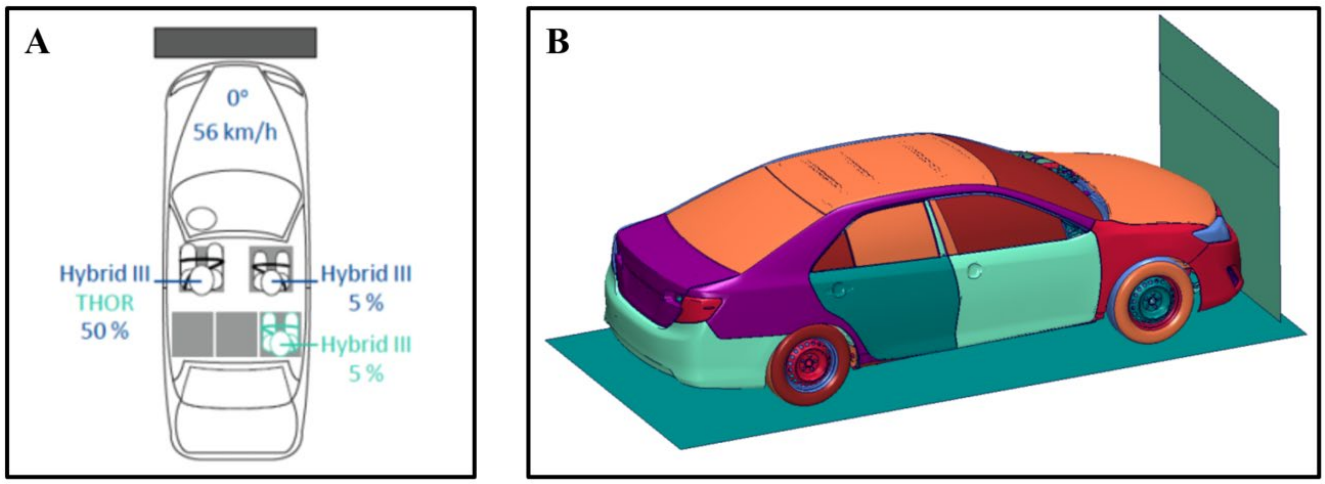

Fig. 6: Configuration of USNCAP full-frontal crash. **a** USNCAP full-frontal crash configuration with dummy specifications. Figure taken from Safety Companion 2020 [6]. **b** The open-source Toyota Camry LS-DYNA model [7] was used for USNCAP full-frontal crash simulation.

The parts which underwent thickness change in the last simulation model v13 compared with the predecessor model v12 are highlighted in Fig. 7. Their part IDs and thicknesses in model v12 and v13 are shown in Table 1. Simulation v12 showed unideal deformation mode of the longitudinal members, which led to high firewall intrusion values and insufficient energy absorption of longitudinal members. These thickness changes in model v13 were countermeasures mainly aimed at improving crash performance of the longitudinal members.

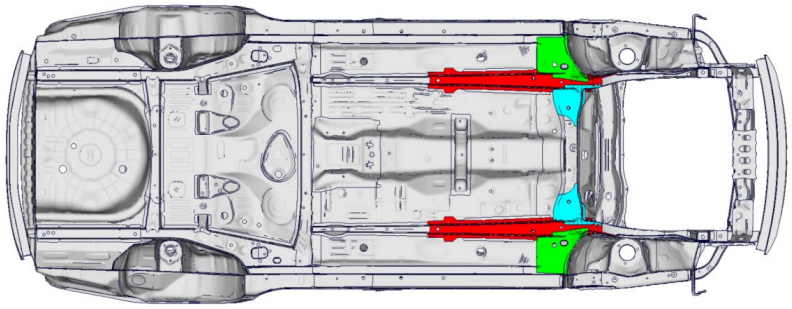

Fig. 7: Body-in-White colored in gray with parts of thickness changes highlighted.

Table 1: PIDs and thicknesses of parts of changes in simulation v12 and v13.

PID	Thickness in v12 [mm]	Thickness in v13 [mm]
10000016	1.653	1.9
10000116	1.397	1.6
10000247	2.02	2.2

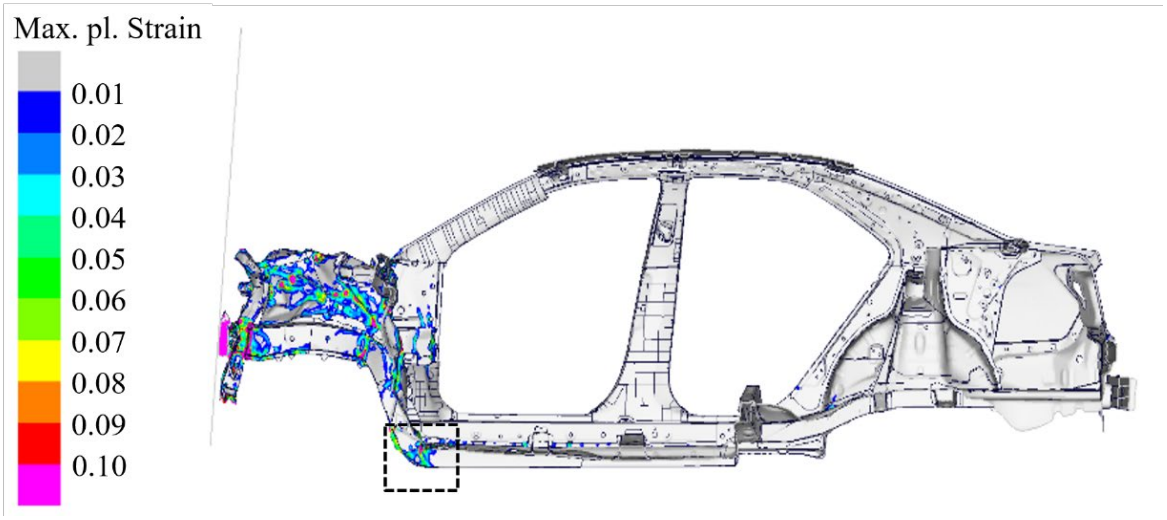

Fig. 8: Results of maximum plastic strain of Body-in-White in simulation v12. Buckling in the transitioning region (black box) of the longitudinal members led to insufficient energy absorption of the longitudinal members.

In Fig. 9, the time points of significant differences of history outputs between simulation v13 and the database were visualized. Furthermore, the raw curve data of four of the history outputs containing significant differences were plotted. Based on the type of the history outputs, colors were assigned for clearer visualization. In this example, only the top 15 curves sorted based on the magnitude of differences were considered for each history output type in order to avoid overlapping. The time cluster gave an overview of the temporal distribution of the significant differences. At each of these time points in the time cluster, differences in structural behaviors were highly likely to be expected in simulation v13 compared with its predecessors. The spatial cluster provided the information of the locations of these points on the vehicle.

For the visualization of the spatial distribution of significant differences, firstly time clustering was divided into four domains with equal interval. Points within the same domain were visualized together. For each history output type, a unique way of highlighting was determined taking the functionalities of Animator into consideration. Outputs of "secforc" were indicated with the parts colored in blue. In addition, the locations of the cross-sections were highlighted with 3D impactor points. For the visualization of points of "matsum" type, parts were marked in red. As for outputs of "nodout", nodes were labeled with their IDs. Furthermore, all points in time clustering were also labeled with their names for better identification.

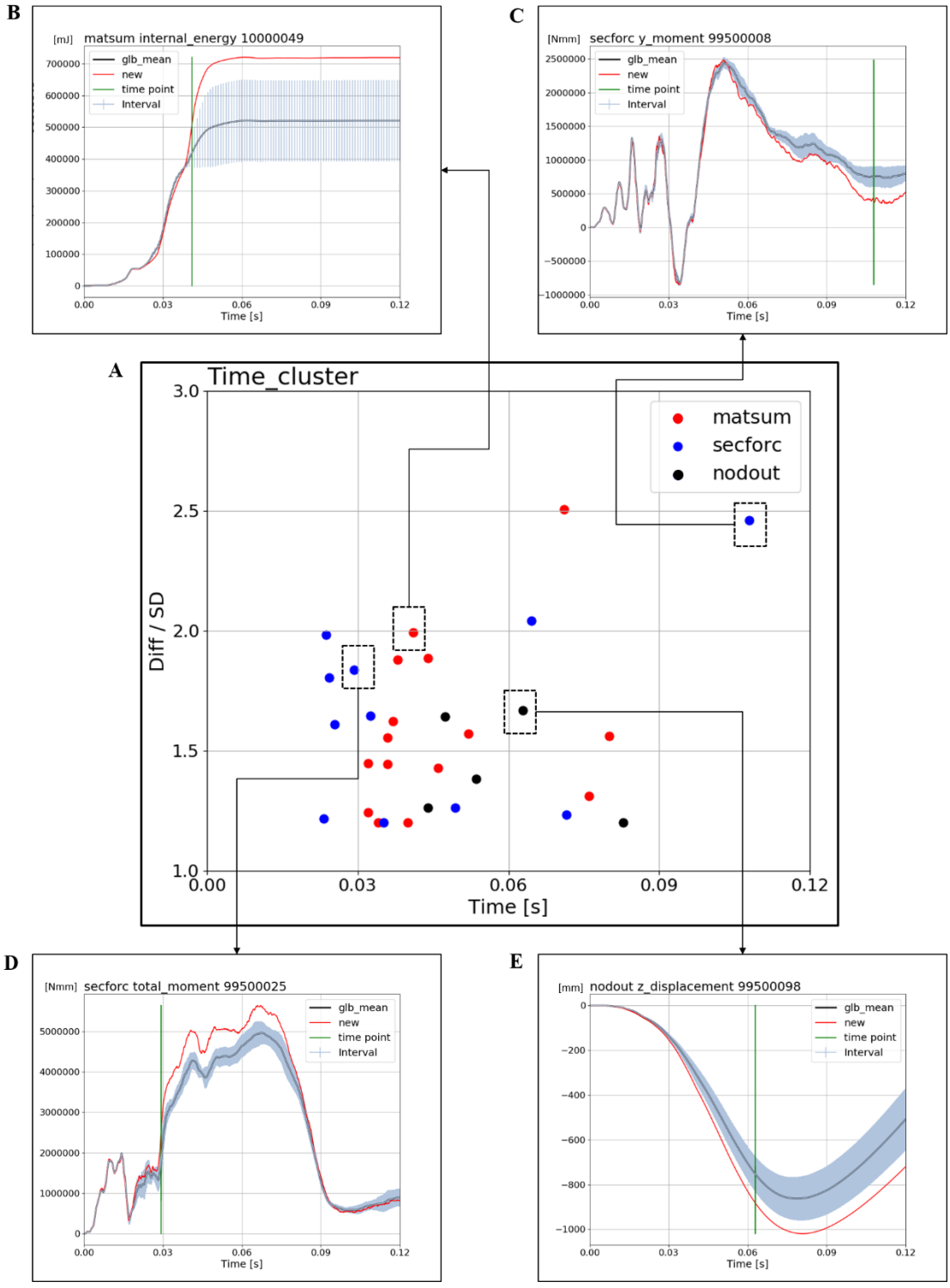

Fig. 9: Time clustering of significant differences of history output between simulation v13 and statistical database. **a** Temporal distribution of significant differences and ratio of differences to standard deviation. **b** Significant difference identified at about 0.04s in comparison of internal energies of a part. **c** Significant difference identified at about 0.108s in comparison of y moment of a cross-section. **d** Significant difference identified at about 0.03s in comparison total moment of a cross-section. **e** Significant difference identified at about 0.063s in comparison of z displacement of a node.

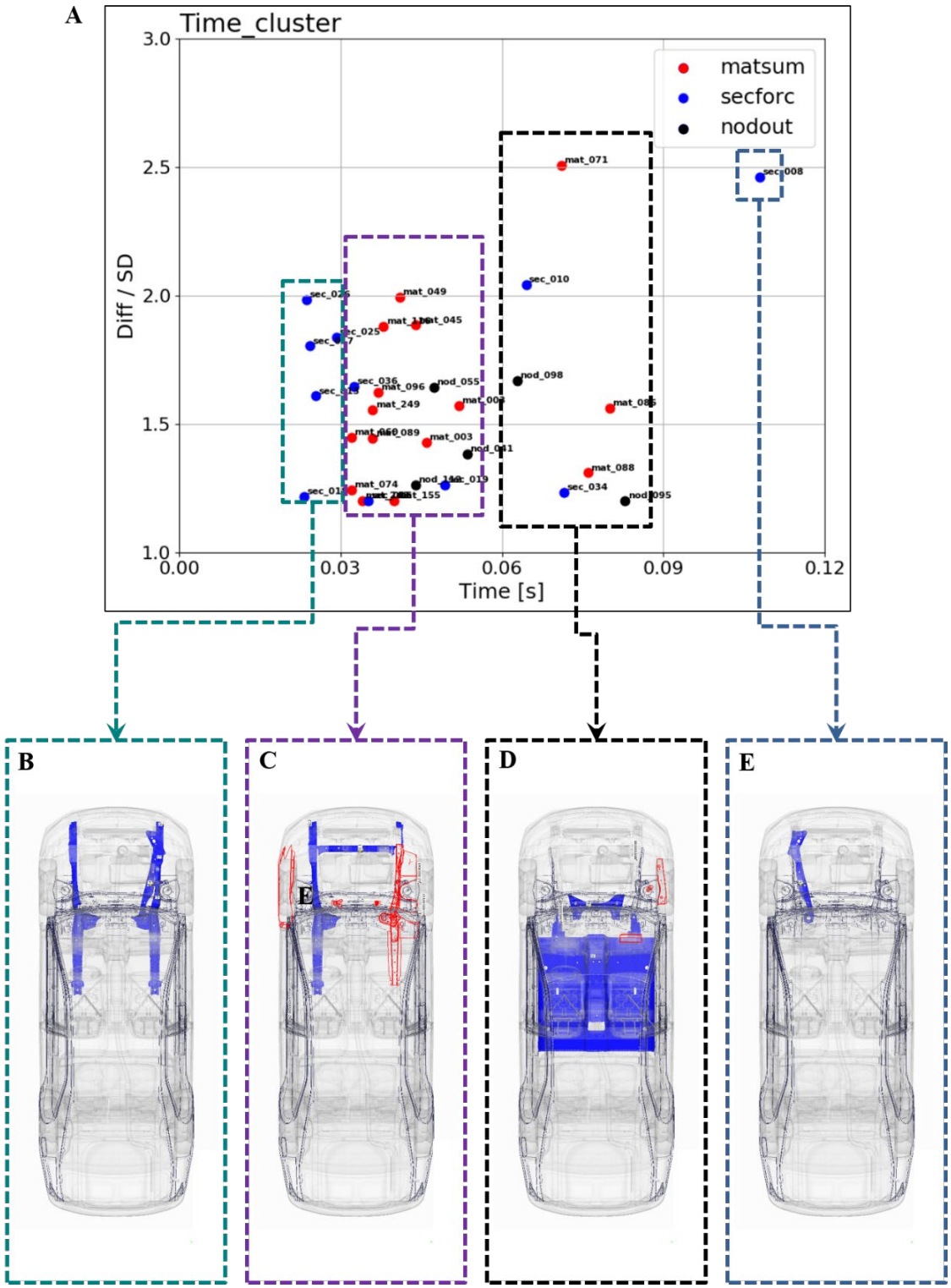

Fig. 10: Temporal and spatial plots of significant differences of simulation v13 outputs compared with statistical database. **a** Time clustering with the time domain divided into four equal intervals. **b** Corresponding regions of significant differences in 0-0.03s highlighted in vehicle model. **c** Corresponding regions of significant differences in 0.03-0.06s highlighted in vehicle model. **d** Corresponding regions of significant differences in 0.06-0.09s highlighted in vehicle model. **e** Corresponding regions of significant differences in 0.09-0.12s highlighted in vehicle model.

Discussion

In this work, we have presented a method for automatic evaluation of all history outputs of an LS-DYNA simulation using qd-cae-python. Currently, this method identifies the significant differences in history outputs of a new simulation by comparing them with a reference database. The method was demonstrated using simulation results of USNCAP full frontal crash of Toyota Camry. In this example, the "binout" file contained approximately 15,000 curves. Twelve simulation results were used for database creation. The method imposes no limitation on number of curves and simulation samples. The time cluster showed the time points of the significant differences with the degrees of these differences. The parts or regions of the vehicle, which were most likely responsible for these differences in history outputs were presented in the spatial cluster. Different structural behaviors were to be expected at these regions at the time point indicated in time cluster.
By combining the results of the temporal and spatial distributions of the significant differences, a sequence of different structural behaviors could be determined. Through more detailed analysis, the interdependencies between these structural behaviors could be identified. Furthermore, considering that these differences were the causes by the thickness changes, with further development of the method, it can be used for sensitivity analysis and robustness studies.

The method is under improvement and further development. Currently there is an open point in the step of history output comparison. One of the criteria used for significant difference detection was to compare the difference with ten percent of the maximum amplitude of the curve. This has yielded relatively promising identification of significant difference overall. However, delays in detections were observed in some cases. A more flexible and robust detection algorithm is needed.

Furthermore, at the moment the selection of simulation samples for database creation was conducted based on engineering judgment of similarity of the simulations. This approach is rather subjective. Since there are various aspects of a simulation, i.e. geometry, materials, boundary conditions, discretization methods, it is difficult to define a norm for determining the similarities between simulations, which accounts for all aspects. On a more abstract level, the simulations can be considered similar if the parts constituting the model are relatively comparable and the boundary conditions are the same, which means the same load case. Theoretically, the initial kinetic energies of all parts can be used as a signature of simulation for checking these two conditions. A more objective sampling method could therefore be, if for instance more than 95 percent of all initial kinetic energies of parts are the same, it is then reasonable to consider the two simulations similar with each other. This information is contained in database "matsum".

Outlook

A more flexible algorithm for significant difference detection is under consideration. The key capabilities of the algorithm is to detect large local differences and evaluate their magnitudes on a global scale at the same time. Based on these two criteria, the algorithm will then determine if the point should be considered as a significant difference. This actually also summarizes how the human eyes work in such situations. A neural network based detection algorithm is under consideration.

In this work, only direct history outputs were analyzed. Currently, no post-processed values were considered. However, for instance, the Head Injury Criterion (HIC) value, Vehicle Pulse Index (VPI) value or force-displacement properties of certain structural components are normally not direct history output. They are of great importance for evaluation of crashworthiness. In these cases, post-processing is then necessary.

As one of the next steps of this work, algorithms for identifying correlations of post-processed values and history outputs will be developed. Furthermore, filters will be selected and applied according to post-processing conventions on the history outputs.

References

[1] Github, qd python library for CAE, https://github.com/qd-cae/qd-cae-python
[2] C. Diez (2018) *qd-Build your own LS-DYNA Tools Quickly in Python*, 15th International LS-DYNA Users Conference
[3] LS-DYNA Manual R11.0 - Vol I - Section DATABASE, https://www.dynasupport.com/manuals
[4] *Animator4 version 2.4.3 Reference Manual – Python-Interface & API,* GNS mbH
[5] SafetyCompanion 2020, carhs.training GmbH
[6] D. Marzougui, D. Brown, C.K. Park, C.D. Kan, K.S. Opiela, (2014) *Development & Validation of a Finite Element Model for a Mid-Size Passenger Sedan,* 13th International LS-DYNA Users Conference