The DX Competition 2025 and Its Benchmarks

Ingo Pill[1]
Institute of Software Engineering and Artificial Intelligence, TU Graz, Austria

Daniel Jung[2]
Department of Electrical Engineering, Linköping University, Sweden

Eldin Kurudzija[3]
Institute of Space Propulsion, German Aerospace Center (DLR), Köln, Germany

Anna Sztyber-Betley[4]
Warsaw University of Technology, Poland

Michał Syfert
Warsaw University of Technology, Poland

Kai Dresia
Institute of Space Propulsion, German Aerospace Center (DLR), Lampoldhausen, Germany

Günther Waxenegger-Wilfing
Institute of Space Propulsion, German Aerospace Center (DLR), Hardthausen am Kocher, Germany
Institute of Computer Science, University of Würzburg, Germany

Johan de Kleer[5]
c-infinity, Mountain View, CA, USA

Abstract

Fault diagnosis has been addressed in many research communities, leading to a variety of available fault diagnosis techniques. Deciding as a user which fault diagnosis methods are suitable for a specific application is thus a nontrivial task. Benchmarks can provide the community with a holistic understanding of the landscape of newly developed and available fault diagnosis methods when making this decision. After a long hiatus, we revived the DX Competition with three fault diagnosis benchmarks: SLIDe, LUMEN, and LiU-ICE. The purpose of the benchmarks is to inspire fault diagnosis research with challenging problems in cyber-physical systems relevant for industry. The benchmarks share a common code structure and we used similar performance metrics in order to simplify the adaptation of diagnosis system solutions to the different case studies.

2012 ACM Subject Classification Computing methodologies → Causal reasoning and diagnostics

Keywords and phrases Diagnosis, Algorithms, Evaluation

Digital Object Identifier 10.4230/OASIcs.DX.2025.14

Category DX Competition

Supplementary Material
Other (DXC'25 Homepage): `https://conf.researchr.org/home/dx-2025#Competition`
Other (DXC'25 Benchmarks, incl. Datasets and Instructions): `https://vehsys.gitlab-pages.liu.se/dx25benchmarks/`

[1] Chair DX Competition 2025
[2] Chair LiU-ICE Benchmark
[3] Chair LUMEN Benchmark
[4] Chair SLIDe Benchmark
[5] Co-Chair DX Competition 2025

36th International Conference on Principles of Diagnosis and Resilient Systems (DX 2025).
Editors: Marcos Quinones-Grueiro, Gautam Biswas, and Ingo Pill; Article No. 14; pp. 14:1–14:19
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Acknowledgements We would like to thank all of our colleagues who contributed to making the DX Competition 2025 happen and who worked with us on the benchmarks. This includes in particular Erik Frisk, Mattias Krysander, Tobias Lindell, Tobias Traudt, Jan Deeken, Justin Hardi, Stefan Schlechtriem, Michael Börner, Dmitry Suslov, Robson dos Santos Hahn, Sebastian Klein, Wolfgang Armbruster, Jan Haemisch, Christopher Groll, Max Axel Müller, and Vincent Bareiß.

1 Introduction

Reasoning about the root causes for an encountered problem is a common task. Whenever an order is not delivered, our car does not start, a program does not work, our multimedia system stops working, when we feel ill, and in many other situations we are interested in the reasons why something was or is not working as expected. Only once we know the source of the problem can we begin to effectively address and solve it to mitigate the issue.

Diagnosis algorithms address this need, in that they tell us exactly which sets of malfunctioning parts in a system can explain the unexpected behavior. The corresponding approaches are sometimes dedicated to very specific scenarios [33, 38] and exploit specific aspects of a diagnostic problem [35], but most concepts are general enough to be applicable to a wide variety of systems. Whenever we refer to systems, we do so in the most abstract sense in that we mean actually any artifact that we can reason about. The system targeted by a diagnostic process might thus be digital, logical, analog, mechanical, cyber-physical, biological, ecological, ethical, and economical, or it could also refer to, for instance, a social system, a supply chain, or a process.

Thus, many research communities have been working on concepts and algorithms for fault diagnosis, and they have been doing so based on a diverse set of underlying techniques. This led to a large variety of available approaches that range from symbolic [42, 34, 7] to sub-symbolic [25, 1], hybrid [28] and statistical [5, 36] ones, and which potentially aim at diagnosing a single [34] or multiple [39] scenarios. In rare cases, they target even the isolation of all faults that are present in a system [37] by generating and considering data that are hopefully representative enough. Deciding as a user which fault diagnosis methods are suitable for a specific application is thus a nontrivial task, and it is certainly not as straightforward as it might look at first glance. In particular, we have to take into account that all methods come with their own individual ramifications in terms of resource expenditure, required knowledge, and achievable performance. Each solution is based on hidden assumptions, see, *e.g.* [20], which affect the quality of the results computed in a given scenario.

One solution for providing the community with a holistic understanding of the landscape of available and newly developed methods is the use of well-formulated benchmarks. They allow the scientific community to propose different solutions to diagnostic systems and draw on a well-founded comparison option to evaluate their performance. As we shall discuss in Section 2, there is a variety of such benchmarks available. Individual papers tend to use only a subset of those, usually in combination with paper-specific benchmarks. Furthermore, we have to take into account that the computation hardware as well as available tools (like SMT/SAT/constraint solvers or simulators) change significantly over time. All of these aspects make it hard to maintain an accurate picture of old and new proposals. So, when Reiter argued in his seminal paper [42] that solvers are too slow to search for diagnoses directly (so not taking conflicts into account), he did not anticipate the technological evolution that we have experienced since then and that allowed the advent of corresponding solutions [27] with very competitive performance [30].

After a long hiatus, we thus revived the DX Competition[6], which is an important source for evaluating new and old algorithms and putting their performance into perspective. In 2025, we started DXC'25 with a set of three benchmarks that focus on three individual cyber-physical systems. As we explain in individual sections, the three case studies[7] combine different diagnosis tasks, and they feature different properties, as summarized in Table 1.

Table 1 Characteristics of the DXC'25 benchmarks.

	SLIDe	LUMEN	LiU-ICE
application	steam line	rocket engine	combustion engine
docker container available	Y	Y	Y
real/artificial data	A	A	R
natural/injected faults	I	I	I
attacks	Y	N	N
intermittent faults	N	Y	N
discrete/continuous	C	C	C
fault data/system data/ simulator	FD, SD	FD, SD, SIM	FD, SD
challenges	diag, nonlinear	diag, nonlinear, sim2real	diag, nonlinear

As we can see from the table, the benchmarks focus on persistent faults and cyber-attacks in a variety of continuous nonlinear systems. For some, a simulator is available so that a user can also create their own behavioral samples. For others, real and/or artificial data are provided. All the technical details are available from our DXC'25 benchmark repository, so that a reader may test their own solutions for all the benchmarks described in this paper.

The outline of this paper is as follows. First, related research and other fault diagnosis benchmarks are discussed in Section 2. Then, presentations of each of the three DX benchmarks are given: including a system description, a presentation of considered fault scenarios, and provided resources. The case study SLIDe is presented in Section 3, LUMEN in Section 4, and LiU-ICE in Section 5. A description of the benchmark implementation environments is presented in Section 6 and the evaluation metrics used in the benchmarks are summarized in Section 7. Finally, a summary is given in Section 8.

2 Related research

Various fault diagnosis benchmarks have been proposed. In contrast to text or vision processing, technical fault diagnosis research still suffers due to data scarcity. This is mainly attributed to two causes. First, industrial datasets often cannot be shared due to confidentiality concerns. Second, fault diagnosis is a field of anomaly detection, where the number of normal samples is significantly larger than the number of faulty samples. Therefore, the community can still largely benefit from developing new benchmarks.

DX Community is grounded in logical and model-based approaches [42]. With the advances in machine learning, the community is integrating data-driven approaches. As pointed out in [56], industrial machine learning research (including fault diagnosis) must carefully follow principles to achieve reliable results. One of the crucial aspects is the use of

[6] `https://conf.researchr.org/home/dx-2025#Competition`
[7] available from `https://vehsys.gitlab-pages.liu.se/dx25benchmarks/`

held-out test sets. It makes competitions particularly useful for the evaluation of the proposed algorithms. The study in [19] showed that the performance of the solutions in the competition had a clear connection to the assumptions made about different faults in the design of the diagnostic system. Existing benchmarks, while helpful, carry the risk of overfitting to the test set. This effect was observed in the recent Safeprocess competition [17, 18] based on an internal combustion engine, where the results on the training data were generally overestimating the results on the held-out evaluation set.

There are several benchmarks that have been widely adopted in the fault diagnosis community. The Tennessee Eastman Process (TEP) [8] dataset is a simulation of a chemical process widely used for control and diagnosis research. DAMADICS benchmark [3] is a study of the intelligent industrial actuator. The CWRU dataset [45] serves for the comparison of fault diagnosis of rolling bearings. NASA Ames developed the Advanced Diagnostics and Prognostics Testbed [41] that has been used for benchmarks and competitions; see, e.g., [22]. NASA's Prognostics Data Repository[8] is a collection of datasets for prognostics and health management. It is a valuable resource for remaining useful life (RUL) prediction benchmarking. A simulation-based wind turbine benchmark is proposed for fault diagnosis and fault-tolerant control in [32]. The results of six participants in a competition using the wind turbine benchmark are summarized in [31].

Recently, a few benchmarks were proposed inside the DX community. A leak detection and localisation benchmark, including structural model of water distribution network, and simulated dataset, was proposed in [51, 50]. An ensemble of benchmarks based on simulated tank systems was presented in [2]. The set requirements for AI benchmarks in the domain of Cyber-Physical Production Systems were formulated in [11], additionally introducing a comprehensive benchmark, offering applicability on diagnosis, reconfiguration, and planning approaches. A TuLAUT (Theory and Teaching of Automation Technology) website[9] provides a curated collection of industrial datasets, including many fault diagnosis datasets.

The DX Competition has a history of successful editions [22, 21, 40, 47, 13], including synthetic track based on faults injected into ISCAS85 circuits, industrial tracks ADAPT and ADAPT-Lite, based on the Electrical Power System (EPS) testbed, software track, and thermal fluid track, which presented problems in a building's heating, ventilation, and air conditioning (HVAC) domain. DXC competitions gave rise to or helped evaluate numerous diagnostic algorithms, including FACT [43], HyDE [29], LYDIA [14], ProADAPT [26] RODON [24], and Wizards of Oz [16].

The range of problems covered in fault diagnosis benchmarks is extensive (from software and digital circuits to continuous processes from various domains), but it is still far from exhaustive. The benchmarks vary in complexity and the task (diagnosis, prognosis, planning). Due to data scarcity, primarily covering data with faults, many of the benchmarks rely on simulated data. There is a lack of benchmarks offering data and a structured process description.

3 SLIDe

SLIDe (Steam Line Intrusion Detection Benchmark) benchmark is devoted to the analysis of diagnostic algorithms for the detection and isolation of process faults and the detection of cyberattacks on a simulated fragment of the steam line of a fluidized bed boiler including

[8] https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/
[9] https://tulaut.github.io/

the third and fourth stage of superheaters. It includes challenging scenarios that exhibit sensor, actuator, and technological component faults as well as cyberattacks. To reflect the industrial nature of the benchmark, we provide only a qualitative description of the process with a list of measurements and a few prepared datasets representing different operating conditions, but only for fault-free and attack-free states.

The 2-stages steam line superheaters simulator models the processes within the boiler of a power unit. In each of these sections, there is an attemperator, a superheater, and a cascade controller – the main controller controls the temperature of the steam after the superheater, while the auxiliary controller, which controls the injection water valve, controls the temperature after the cooler. The schematic diagram of the process is shown in Figure 1.

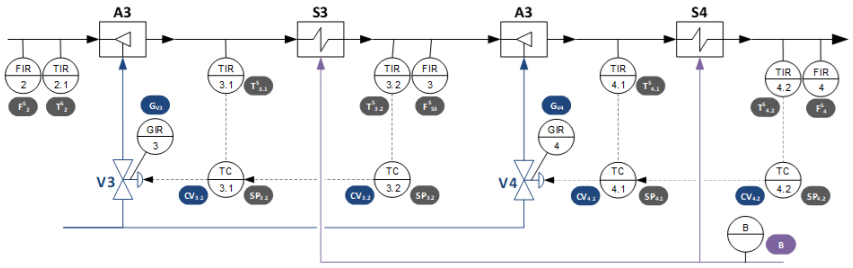
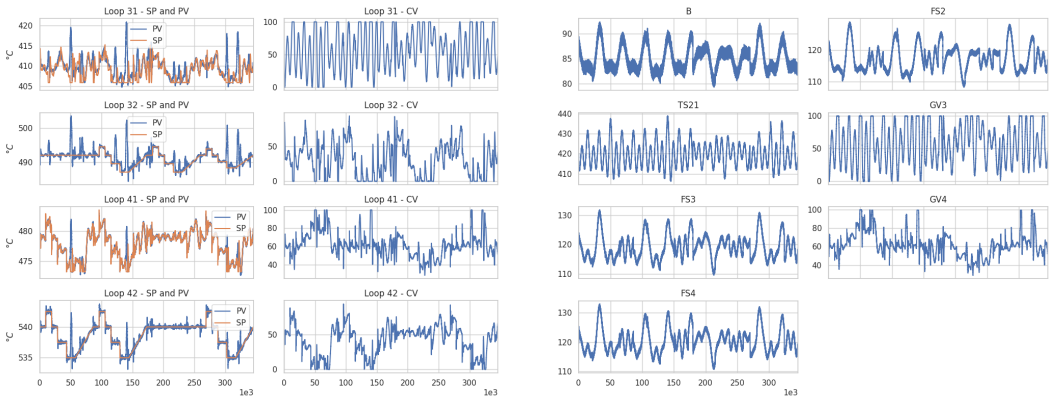


Figure 1 Process block diagram.

The benchmark simulator is implemented in Matlab. Control and measured variables are shown in Figure 2. B denotes the fuel inflow to the boiler, F steam flows, T temperatures, G positions of the injection valves, SP set points, and CV control signals. Figure 2 shows traces of control loops and process variables.



(a) Control loops.

(b) Process variables.

Figure 2 Control loops and process variables.

3.1 Process faults

The benchmark includes 16 process and sensor faults. The symbolic locations where process faults can be introduced are shown in Figure 3.

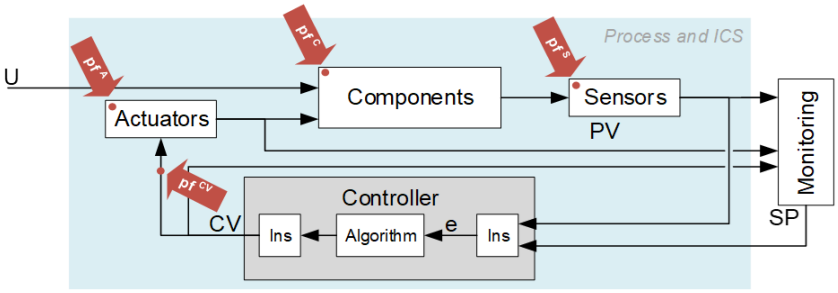


Figure 3 Symbolic designation of types and places of introduction of process faults.

Process faults are divided into the following types according to the entry points:

- pf^S: Incorrect operation of the measurement signal path.
- pf^A: Faulty operation of the actuator.
- pf^{CV}: Control signal path malfunction.
- pf^C: Technological component fault.

3.2 Cyber-attacks

Each cyber attack is carried out according to a designed scenario – a specific method of attack. Such a scenario consists of elementary impacts on individual system elements and signals in communication channels called cyber faults. The symbolic locations for introducing cyber faults in the simulator are shown in Figure 4.

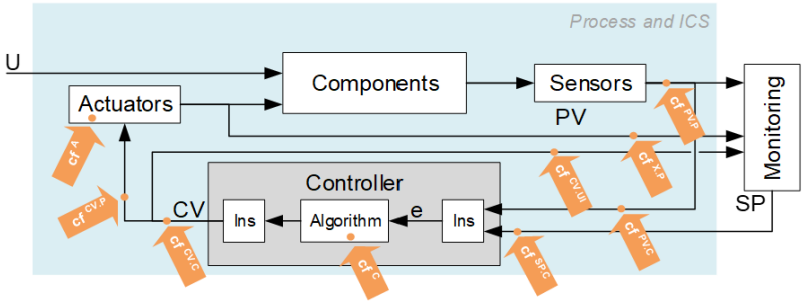


Figure 4 Symbolic designation of types and places of introduction of cyber faults.

We consider the following types of cyberattacks:

- cf^C – attack on the controller (change of operating mode, change of parameters),
- cf^SP – modification of set-points,
- cf^{CV} – modification of control variables,
- cf^{PV} – modification of controlled variables,
- cf^A – attack on the actuator (blockage, modification of operation, changes of operating parameters).

Cyber faults should be isolated to the specific control loop, *i.e.* the competitor's task is to detect cyber faults and say which control loop is affected. It is possible for more than one control loop to be affected by the same cyber-attack scenario. It is not necessary to isolate the cyber fault to the specific component.

3.3 Additional resources

Training datasets are available at the competition website[10]. Training and evaluation data from the previous version of the benchmark are available [48]. Exemplary algorithms for fault and cyber-attack detection and isolation can be found in [49, 53, 52].

4 LUMEN

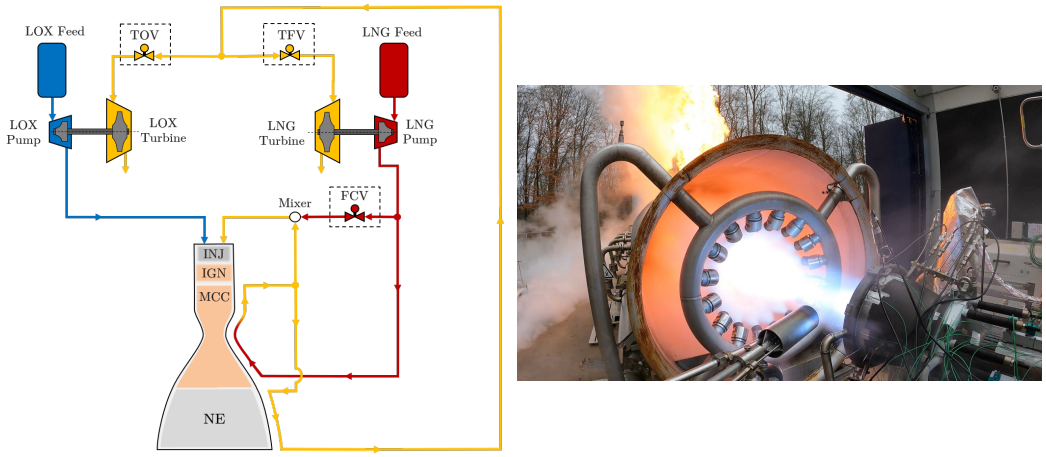
Early launch vehicles such as the American *Saturn 5* or the European rocket family *Ariane*, were expendable. Diagnostics therefore focused on pre-flight tests and post-test evaluations while no sophisticated system was used on-board. However, as the space industry shifts towards reusability and cost reduction, on-board diagnostic systems for health monitoring are essential for next-generation rocket engines. The *RS-25*, Space Shuttle's Main Engine (SSME), was the first reusable liquid rocket engine (LRE). For on-board diagnostics, dynamic limit-checks (redlines) for critical engine parameters, *e.g.* rotational speed of the turbopumps or combustion chamber pressure, were performed [6]. Those redlines were defined based on engineering judgement and experience. Although this simple method works well for most component faults, there are still severe problems: Sensor faults can cause unnecessary engine shutdowns and component faults can remain undetected. During the operation of rocket engines, sensors, such as pressure transducers and thermocouples, are subjected to high thermal and mechanical stresses, which make them susceptible to failure. This is underlined by the history of Space Shuttle ground test aborts, launch delays and the launch abort of flight STS-51-F caused by faulty sensors [4]. In addition, undetected component faults can result in catastrophic events [54]. For a more sophisticated diagnosis of the engine's health, vibration data was used on-board of the SSME to detect faults in the turbopumps which are the most common source of failure in rocket engines [6]. Newer reusable launchers such as SpaceX's *Falcon 9* also use sophisticated systems for detecting off-nominal conditions and initiating autonomous safe shutdowns [46]. The effectiveness of their health monitoring system was demonstrated on various flights, *e.g.*, during flight 84 where one of the nine engines on the first stage of the rocket was shutdown due to an anomaly and the mission was still successful. Following SpaceX's lead, Europe and other nations are actively researching and developing reusable rockets. The European rocket engine *Prometheus*, for example, is being optimised for reusability and cost reduction. The diagnosis of the the engine's health plays a vital role in achieving this goal [44]. The development and testing of these diagnostic systems involves component-level testing and ground tests, with the ultimate goal of ensuring their suitability in-flight.

LUMEN (**L**iquid **U**pper stage de**M**onstrator **EN**gine) is a modular pump-fed liquid oxygen (LOX) and liquid methane (LNG) rocket engine with 25 kN thrust, developed by the Institute of Space Propulsion of the German Aerospace Center (DLR). The operational envelope of LUMEN covers combustion chamber pressures from 40 bar to 80 bar and mixture ratios between 3.0 to 3.8. DLR has successfully completed two hot-fire test campaigns of LUMEN, demonstrating key capabilities such as stable combustion over a wide throttling range of 38 bar to 78 bar [10]. With this achievement LUMEN is now fully operational and available as a testbed for the development of intelligent control and diagnosis systems for health monitoring.

[10] `https://conf.researchr.org/home/dx-2025`

4.1 System description

This benchmark is proposed as a challenging problem for fault diagnosis of safety-critical technical systems with focus on the LUMEN engine. LUMEN, as shown schematically in Figure 5, is operated in an expander-bleed cycle. Both propellants are pressurized by separate turbopump units. While LOX is injected directly into the combustion chamber (MCC), LNG is first used for the regenerative cooling of the combustion chamber in a counterflow arrangement. The heated coolant flow is partially remixed with LNG to actively control the fuel injection temperature. The remaining cooling mass flow is further heated within the nozzle extension (NEM) and is then used to drive LOX and LNG turbines. Afterwards, the turbine exhaust is vented without being combusted. The generated thrust and therefore the operating point of LUMEN is defined by the combustion chamber pressure, the mixture ratio and the cooling channel mass flow. LUMEN's operating point is set by the position of the control valves TFV, TOV and FCV in open-loop. A more detailed description of LUMEN can be found in [9, 23, 55].



(a) Flow scheme of LUMEN.

(b) Thrust chamber of LUMEN during a hot-fire test.

Figure 5 The LUMEN benchmark.

4.2 Challenges

The development of fault diagnosis systems for LUMEN is complicated for various reasons, *e.g.* a limited amount of experimental data, measurement inaccuracies, nonlinear dynamics, the wide throttling range and strong coupling of all components. The placement of sensors is also constrained by extreme temperatures, high pressures and vibrations within the engine, which can damage instrumentation and lead to unreliable measurements. In addition, experimental data for fault scenarios can not be intentionally collected due to the inherent risk of a catastrophic failure. As a result of the extreme operating conditions close to the physical limits, the engine may also fail for unpredictable reasons. A number of challenges in developing a diagnosis system are defined by these difficulties.

The required robustness to unknown faults due to the lack of data for failure scenarios is one of the main challenges. The diagnosis system should detect any deviation from nominal operation as fast as possible but also reason based on the symptoms whether a detected fault can be isolated to known fault scenarios or is unknown. Another challenge poses the

scarcity of experimental data. Accurate reduced-order simulation models offer the possibility of generating representative data in controlled environments for both nominal operation and fault scenarios. However, resulting from unavoidable modeling errors in reduced-order models, the simulator is only an approximation of the real system. Closing this simulation-reality gap is not trivial and needs to be addressed in the development of diagnosis systems for rocket engines.

4.3 Provided resources

For developing the diagnosis system, a transient simulation model of LUMEN is provided. The simulator accurately reproduces experimental results in both transient and steady-state conditions with errors below 10 %. The simulator is based on differential-algebraic equations (DAE) and built with *EcosimPro*, the state-of-the-art modeling tool for space applications, and the European Space Propulsion System Simulation (ESPSS) library. The behavior of each component is defined by a set of geometrical and physical parameters such as the length, diameter and wall roughness of a pipe which cannot be changed in the simulator. For performing transient simulations, a Python interface is provided which can be used to adjust the position of the control valves and therefore change the operating point of LUMEN at each time step. The output of the simulator consists of noisy measurements at positions in which sensors are commonly placed within a flight-like rocket engine. In total, the output of the simulator consists of eight pressure measurements, *e.g.* the combustion chamber pressure, seven temperature measurements, *e.g.* the turbine inlet temperature, the rotational speed of the two turbopumps, the command and position signal of each valve as well as five mass flows at different positions, *e.g.* the injection fuel mass flow. As flowmeters are not used on board of rocket engines, the provided mass flows are calculated based on other measurements. The simulator can be used to generate both nominal trajectories and trajectories in which a fault is introduced at an user-defined time. In total, 15 sensor faults, three actuator faults and three components faults can be simulated.

To replicate the challenge of the adaption to the real system in this benchmark, we introduce another simulation model that is not provided to the participants. The modified simulator has a slightly different set of physical parameters and is a representation of the real system in this benchmark. This real system simulator is used for evaluating the performance of the diagnosis systems. To mimic the scarcity of available experimental data, a limited set of nominal trajectories which is generated with the real system simulator is also provided. In addition to the known fault scenarios, we use the real system simulator to generate trajectories for fault scenarios which are unknown to the participants *a priori*. If the symptoms of these faults differ from known faults, the diagnosis system should classify this fault as unknown. An overview of the provided resources and the evaluation process is given in Figure 6. The evaluation metrics are described in Section 7. For evaluating the diagnosis system solution, a Docker container with evaluation code is provided as described in Section 6.

4.4 Fault scenarios

Component, sensor, and actuator faults can be introduced in the simulation model at each time step. Sensor faults can be injected into all measurements by multiplying the measured variable by a fault factor $f \in [0.8, 1.2]$. As the simulation is performed open loop, sensor faults affect the measured signal and downstream calculations of the mass flows. Component and actuator faults, on the other hand, influence the operation of the entire engine as a result of strong coupling. The actuator fault is modeled as a stuck valve that does not change position according to the command signal. This fault can be introduced in TFV, TOV and FCV. In addition, three component faults with different magnitudes can be simulated:

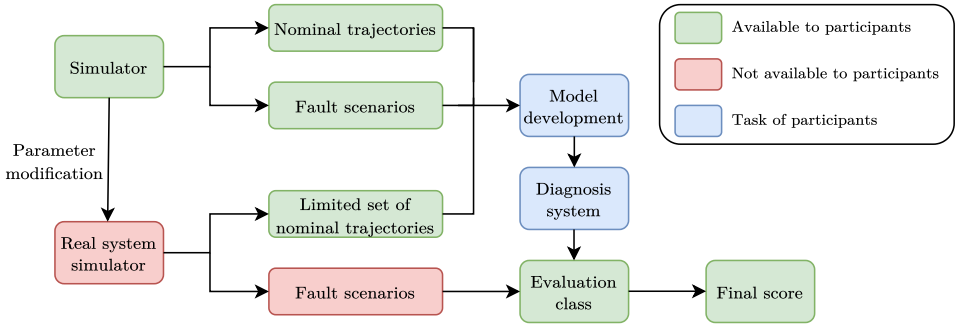


Figure 6 Overview of the evaluation process for the LUMEN benchmark.

- **Blockage of the turbine inlet nozzle**: This fault models a geometrical change in the flow area of the inlet turbine nozzles that can result from stuck particles.
- **Leakage**: This fault simulates a leakage mass flow downstream of the pump. It is modeled with an additional valve that can be partially opened, and thus introduce a leakage mass flow.
- **Increased pressure drop**: This fault simulates an additional pressure drop within the feedlines of the system. It is modeled with an additional valve that can be partially closed to increase the pressure drop.

To illustrate the described effects of each type of fault, Figure 7 shows the normalized sensor signals for the valve command for FCV, TFV and TOV, the fuel injection mass flow, the combustion chamber pressure and the rotational speed of the fuel turbopump (FTP).

5 LiU-ICE

The first version of the Linköping University Internal Combustion Engine (LiU-ICE) industrial benchmark was initially presented in [17]. The benchmark is proposed as a challenging industrial-relevant case study to support fault diagnosis research. Engine fault diagnosis is a nontrivial task that is complicated by nonlinear dynamic behavior, with slow and fast dynamics, a wide operating range, and both stationary and transient operation.

Developing a diagnosis system is complicated by system model inaccuracies, measurement uncertainties, and limited training data from relevant fault scenarios. The objective of the competition is to address these challenges by designing a diagnosis system for the air path of an internal combustion engine.

5.1 System description

The benchmark consists of operational data collected from an internal combustion engine test bench, see Figure 8a and a mathematical model, where the model parameters are unknown. Figure 8b shows a schematic of the modeled part of the system, which is the air path through the engine. The available sensor signals are as follows:

- y_{pic} – Intercooler pressure
- y_{Tic} – Intercooler temperature
- y_{pim} – Intake manifold pressure
- y_{waf} – Mass flow through the air filter
- y_{xpos} – Throttle actuator position

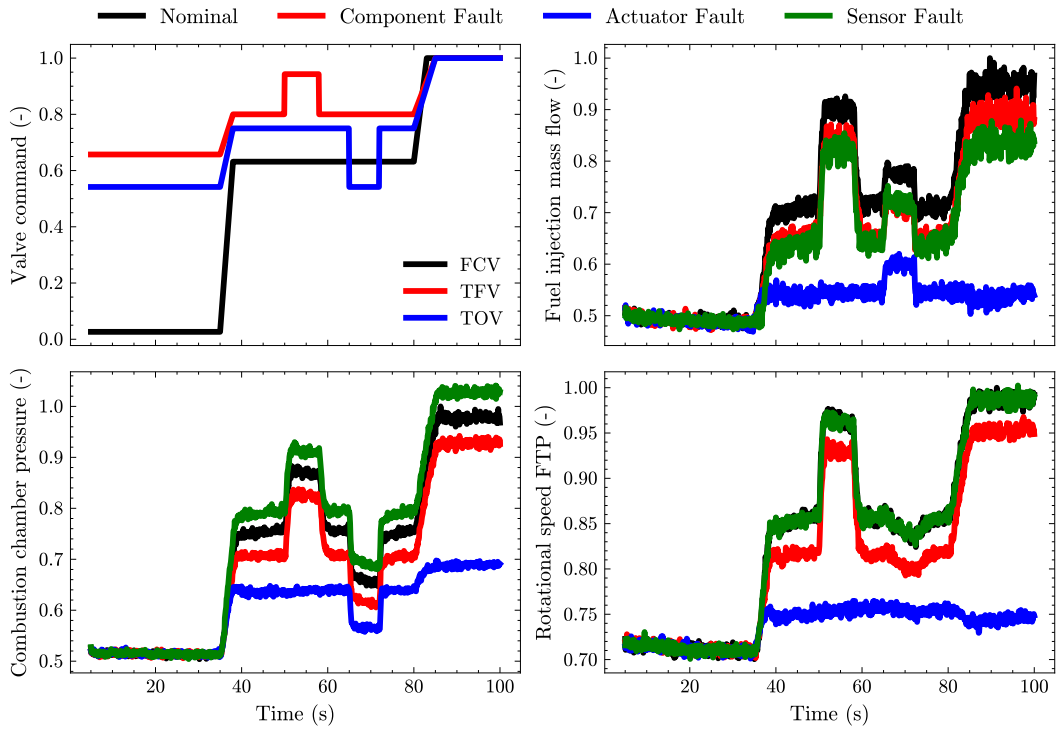


Figure 7 Examples of each fault type on normalized sensor signals. The fault is injected at different $t = 36$ s. The valve commands are identical for each fault.

- y_ω – Engine speed
- y_{pamb} – Ambient pressure
- y_{Tamb} – Ambient temperature

The known actuator signals are as follows:

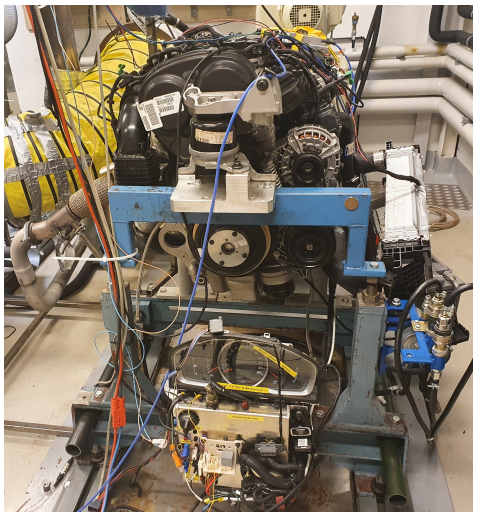
- u_{mf} – Requested injected fuel mass
- u_{wg} – Requested wastegate actuator position

The available signals represent a set of standard signals that are available in a production engine. Note that most signals are available in the air intake of the engine, see Figure 8b.

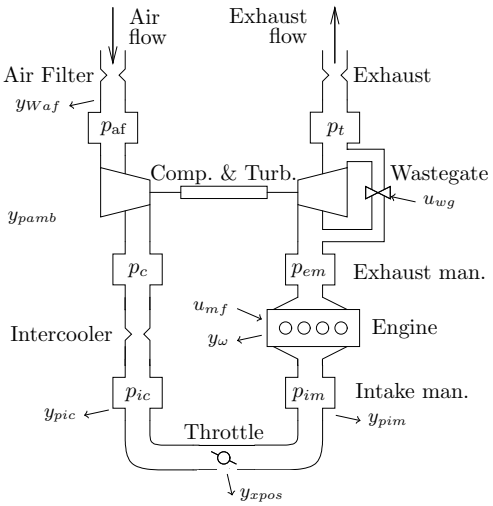
The airflow passes through an air filter before the compressor and the intercooler. A throttle is used to control the pressure in the intake manifold where air enters the cylinders, where it is mixed with fuel and ignited to generate torque. The exhaust gases pass through the exhaust manifold and the turbo that drives the compressor before leaving the exhaust. The wastegate is used to control how much of the exhaust gases pass through the turbo. The engine control unit makes sure that the engine provides the desired torque while controlling the stoichiometry of the air and fuel in the cylinder to optimize combustion and reduce emissions.

5.2 Fault scenarios

Faults are introduced during operation, either by opening a valve that represents a leak or by modifying a measurement signal in the engine control unit, representing a sensor fault. The faults considered are the following:



(a) The LiU-ICE test bench.



(b) A schematic of the air path of an IC engine.

Figure 8 The LiU-ICE benchmark.

- f_{ypic} – A fault in the inter-cooler pressure sensor y_{pic}
- f_{ypim} – A fault in the intake manifold pressure sensor y_{pim}
- f_{ywaf} – A fault in the air mass flow sensor y_{waf}
- f_{iml} – A leakage in the intake manifold

Sensor faults are injected as multiplicative as $y = (1 + f)x$ where y is the measurement signal, x is the measured variable, and $f \neq 0$ represents a fault that scales the measured signal. Since a sensor fault is introduced during operation, it can affect the operating conditions of the system through feedback loops. The magnitude of the leakage fault f_{iml} is defined by the diameter of the valve orifice.

5.3 Provided resources

In the benchmark, a mathematical model of the system and training data from various fault scenarios are provided. The mathematical model is in the form of semi-explicit differential algebraic equations (DAE) of index 1. The component models are similar to what is described in [12]. The provided model is implemented in the Fault Diagnosis Toolbox [15]. A structural representation of the provided model is shown in Figure 9 where the blue dots represent unknown variables, the red dots are fault signals, and the black dots are known signals.

5.3.1 Training data

The training data for this version of the LiU-ICE benchmark consist of 26 datasets and include different magnitudes of each fault. Each data set is sampled at 20 Hz. All training data sets in the benchmark have been collected using the Worldwide Harmonized Light Vehicle Test Procedure (WLTP). The WLTP cycle is approximately 30 minutes long and covers varying operating conditions and transient behavior that represent both urban and highway driving. Each fault is introduced after approximately two minutes into each corresponding dataset and is present for the rest of the cycle. A summary of the fault realizations is shown in Table 2. For sensor faults, the fault signal f can be both positive and negative, that is, the faulty signal is scaled up or down with respect to the true signal. Each data set starts with nominal operation, and the fault is injected after approximately two minutes.

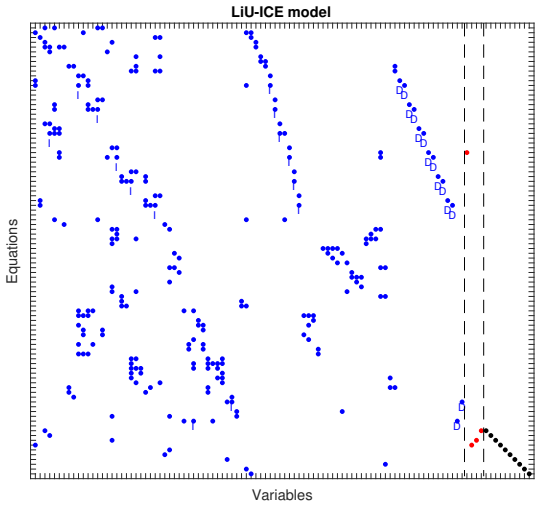


Figure 9 A structural representation of the engine model.

To illustrate the effects of each fault, Figure 10 shows the sensor signals y_{pic}, y_{pim}, and y_{waf} for one realization of each fault. The sensor faults in the plots are +10% and the leakage diameter is 6 mm. The injection of each fault is marked in the corresponding subplot where the fault is most visible. The sensor faults are marked in the corresponding signal and the leakage is highlighted in signal y_{pim} which measures close to the location of the leakage. In the figure, the signals have been translated in time, so they are synchronized in the drive cycles. Note that since the sensor faults are multiplicative, the same fault size for the different sensor faults results in different excitation in the signals. Since the fault f_{ywaf} is not visible in the plot, a zoomed in figure is shown in Figure 11.

For the final evaluation, a set of secret test data will be used to evaluate all participating solutions. Note that the fault scenarios in the test data can be other driving cycles than the WLTP cycle.

Training data sets are available on the competition website. The previous version of the benchmark is described in [17].

6 Benchmark implementation environment

To simplify the implementation of diagnostic system solutions to the different benchmarks, a standardized data format and code structure in Python are used. Each benchmark provides a Docker container with a similar evaluation code and a template for the implemented diagnosis system.

Table 2 Summary of training datasets with fault scenarios.

Fault	Magnitudes
f_{ypic}	-15%, -10%, -5%, 5%, 10%, 15%
f_{ypim}	-15%,-10%, -5%, 5%, 10%, 15%
f_{yWaf}	-15%,-10%, -5%, 5%, 10%, 15%
f_{iml}	4 mm, 6 mm

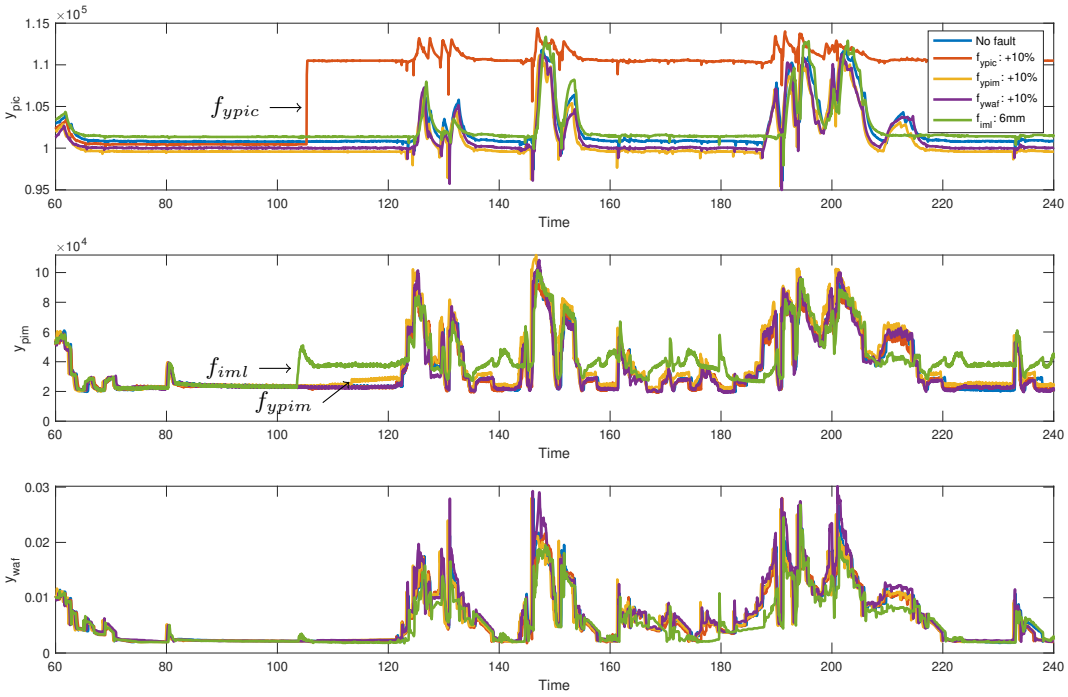


Figure 10 Examples of signals from different fault scenarios.

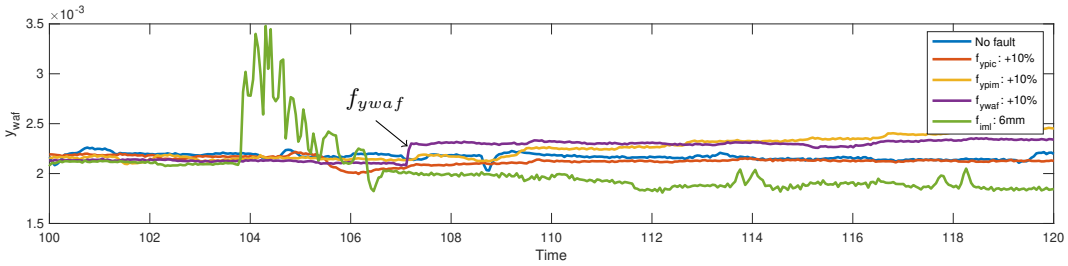


Figure 11 A zoom in of the y_{waf} signal from Figure 10 to show the sensor fault f_{ywaf}.

The diagnosis system solution should be implemented in Python in a class using the following template:

```
class DiagnosisSystemClass:
   def __init__(self):
       pass
    def Initialize(self):
      #initialize diagnosis system here
       pass
    def Input(self,sample):
      #Update diagnosis using new sample
       detection = [] # Set flag if fault is detected
       isolation = [] # Ranking of diagnoses
       return(detection,isolation)
```

found in `DiagnosisSystemClass.py`. Note that it is possible to update the class with functionality needed for the solution. It is important that the inputs and outputs to the above functions are not changed.

The evaluation code provided is found in the file `evaluate_diagnosis_system.py` as part of the benchmarks and calls the diagnosis system every time a new sample of data is available. The diagnosis system requires to return if a fault is detected and a ranking of the diagnoses is obtained. The evaluation of each fault scenario is stored in a csv file in the results folder in the container.

7 Evaluation metrics

There are various performance metrics that can be used for evaluation of diagnosis systems, where we outline in the following those chosen for DXC'25.

7.1 Diagnosis of faults

For each new sample, the diagnosis system should return a fault detection flag. When a fault is detected, the system provides a ranked list of diagnoses with decreasing posterior probability. The diagnosis system should have high fault detection accuracy and a low false alarm rate. At the same time, it is important to isolate the true fault to select a suitable countermeasure. The diagnosis system solutions are evaluated based on the following performance metrics:

- False alarm rate (FAR) – the percentage of samples in which the diagnosis system states that a fault is detected when there is no fault in the system.
- True detection rate (TDR) – the percentage of samples in which the diagnosis system states that a fault is detected when there is a fault in the system.
- Fault isolation accuracy (FIA) – the average probability given to the true diagnosis for all samples when a fault is correctly detected.

All performance metrics are between 0 and 1. The metrics are kept simple to simplify the comparison of different fault diagnosis solutions. The total score is calculated as a weighted sum of these performance metrics as follows:

$$\text{total score} = (\ (1\text{-FAR}) + \text{TDR} + \text{FIA}\ )/3.$$

where a higher value represents better performance. Note that a naive solution can achieve at least 0.3, *e.g.*, by not triggering any alarm.)

7.2 Diagnosis of cyberattacks

In the SLIDe benchmark, diagnosis of cybernetic faults is also evaluated based on the following performance metrics:

- False alarm rate (FAR) – the percentage of samples in which the diagnosis system states that a cybernetic attack is detected when there is no cyber attack in the system. We use 1 - FAR as a metric.
- True detection rate (TDR) – the percentage of samples in which the diagnosis system states that a cyber attack is detected when there is a cyber attack in the system.
- Cyber attacks isolation accuracy (CIA)

The isolation accuracy of cybernetic faults (CIA) is divided into two parts:

- True isolation rate (TIR) – the average probability assigned to the simulated attack vector,
- False isolation rate (FIR) – the average probability assigned to the loops that are not attacked. We use $1 - FIR$ as the isolation accuracy score.

The isolation accuracy score is computed as the harmonic mean of TIR and $1 - FIR$:

$$CIA = \frac{2 \cdot TIR \cdot (1 - FIR)}{TIR + (1 - FIR)} \tag{1}$$

The metrics used for cyber attacks differ in their approach to isolation accuracy because we also consider scenarios when multiple loops are attacked. The proposed metric better evaluates the cases where only some of the attacked loops are isolated correctly in contrast to only considering the probability of a correct diagnosis.

8 Summary

The three benchmarks described in this paper (and which are available from the DXC'25 benchmark repository[11]) served as a starting point for reviving the DX competition after a long hiatus. As we can easily deduce from the characteristics listed in Table 1, our aim was to provide an initial set of challenges that is diverse but also close enough to foster the testing of an approach for all benchmarks.

The benchmarks are continuously updated. Thus, we encourage prospective participants and interested readers to reach out to us for the latest versions. We also plan to complement the current benchmark set with additional ones that cover other types of system, such as discrete ones. Of particular interest will be extensions that cover additional diagnostic problems. This could include intermittent fault scenarios or the evaluation of a system's long-term performance (and degradation).

At the same time, we intend to expand the competition with challenges for approaches that integrate diagnosis with control, repair, and potentially prognosis. Evaluating these integrated approaches will, in particular, allow us to investigate the effectiveness of various diagnosis concepts regarding their integration into design approaches for intelligent systems.

Being able to analyze and, in turn, anticipate the exact needs for diagnostic support in the decision making of an intelligent autonomous system shall provide the community with the background to make educated design decisions towards enabling resilience in a system. That is, the ability to reasoning about problems and their mitigation at run-time enables resilient systems to maintain their functionality not only for anticipated fault scenarios, but also in situations and circumstances that could not be anticipated at design time.

References

1 J. L. Augustin and O. Niggemann. Graph Structural Residuals: A Learning Approach to Diagnosis, 2023. doi:10.48550/arXiv.2308.06961.

2 K. Balzereit, A. Diedrich, J. Ginster, S. Windmann, and O. Niggemann. An ensemble of benchmarks for the evaluation of AI methods for fault handling in CPPS. In *2021 IEEE 19th Int. Conf. on Industrial Informatics (INDIN)*, pages 1–6. IEEE, 2021.

3 M. Bartyś, R. Patton, M. Syfert, S. de las Heras, and J. Quevedo. Introduction to the DAMADICS actuator FDI benchmark study. *Control Engineering Practice*, 14(6):577–596, 2006.

4 T. W. Bickmore. Real-Time Sensor Data Validation. *Contractor Report, NASA-CR-195295*, 1994.

5 P. Chatterjee, J. Campos, R. Abreu, and S. Roy. Augmenting Automated Spectrum Based Fault Localization for Multiple Faults. In *32nd Int. Joint Conf. on Artificial Intelligence (IJCAI-23)*, pages 3140–3148, August 2023.

[11] https://vehsys.gitlab-pages.liu.se/dx25benchmarks/

6 M. Davidson and J. Stephens. Advanced health management system for the space shuttle main engine. In *40th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, 2004.

7 J. de Kleer and B. C. Williams. Diagnosis with Behavioral Modes. In *11th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1324–1330, 1989.

8 J. Downs and E. Vogel. A plant-wide industrial process control problem. *Computers & chemical engineering*, 17(3):245–255, 1993.

9 K. Dresia, M. Börner, W. Armbruster, S. Klein, T. Traudt, D. Suslov, J. Hardi, G. Waxenegger-Wilfing, and J. C. Deeken. Design and control challenges for the LUMEN LOX/LNG expander-bleed rocket engine. In *34th Int. Symposium on Space Technology and Science (ISTS)*, 2023.

10 K. Dresia, T. Traudt, M. Börner, D. Suslov, W. Armbruster, R. dos Santos Hahn, E. Kurudzija, C. Groll, M. A. Müller, S. Klein, J. Hämisch, J. Deeken, J. Hardi, and S. Schlechtriem. Hot-fire testing and system analysis of the LUMEN liquid upper stage demonstrator engine. In *3rd Int. Conf. on Flight Vehicles, Aerothermodynamics and Re-entry (FAR)*, 2025.

11 J. Ehrhardt, M. Ramonat, R. Heesch, K. Balzereit, A. Diedrich, and O. Niggemann. An AI benchmark for diagnosis, reconfiguration & planning. In *2022 IEEE 27th Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, 2022.

12 L. Eriksson. Modeling and control of turbocharged SI and DI engines. *Oil & Gas Science and Technology-Revue de l'IFP*, 62(4):523–538, 2007.

13 A. Feldman, J. de Kleer, T. Kurtoglu, S. Narasimhan, S. Poll, D. Garcia, L. Kuhn, and A. van Gemund. The diagnostic competitions. *AI Magazine*, 35(2):49–54, 2014. `doi:10.1609/AIMAG.V35I2.2532`.

14 A. Feldman, G. Provan, and A. van Gemund. The Lydia approach to combinational model-based diagnosis. *Proc. Int. Workshop on Principles of Diagnosis*, 9:403–408, 2009.

15 E. Frisk, M. Krysander, and D. Jung. A toolbox for analysis and design of model based diagnosis systems for large scale models. *IFAC-PapersOnLine*, 50(1):3287–3293, 2017.

16 A. Grastien and P. Kan-John. Wizards of Oz description of the 2009 DXC entry. *Proc. Int. Workshop on Principles of Diagnosis*, 9:409–413, 2009.

17 D. Jung, E. Frisk, and M. Krysander. The LiU-ICE benchmark–an industrial fault diagnosis case study. *arXiv preprint*, 2024. `arXiv:2408.13269`.

18 D. Jung, E. Frisk, M.s Krysander, A. Sztyber-Betley, F. Corrini, A. Arici, N. Anselmi, M. Mazzoleni, J. Xu, S. Mo, Z. Xu, C. Yang, Z. Du, H. Safaeipour, M. Forouzanfar, V. Mirahi, A. Pinnarelli, V. Puig, Q. Deng, Y. Liu, J. Liu, H. Ke, W. Zhu, S. Merkelbach, M. Ahang, and H. Najjaran. A fault diagnosis benchmark of technical systems with incomplete data – six solutions. *Control Engineering Practice*, 2025. to appear.

19 D. Jung, H. Khorasgani, E. Frisk, M. Krysander, and G. Biswas. Analysis of fault isolation assumptions when comparing model-based design approaches of diagnosis systems. *IFAC-PapersOnLine*, 48(21):1289–1296, 2015.

20 D. Jung and M. Krysander. Assumption-based Design of Hybrid Diagnosis Systems: Analyzing Model-based and Data-driven Principles. In *Annual Conf. of the PHM Society*, 2024.

21 T. Kurtoglu, S. Narasimhan, S. Poll, D. Garcia, L. Kuhn, J. de Kleer, and A. Feldman. Second international diagnostic competition (dxc'10), 2010.

22 T. Kurtoglu, S. Narasimhan, S. Poll, D. Garcia, L. Kuhn, J. de Kleer, A. van Gemund, and A. Feldman. First international diagnosis competition-DXC'09. *Proc. Int. Workshop on Principles of Diagnosis DX*, 9:383–396, 2009.

23 E. Kurudzija, K. Dresia, J. Martin, T. Traudt, J. C. Deeken, and G. Waxenegger-Wilfing. Virtual sensing for fault detection within the LUMEN fuel turbopump test campaign. In *9th Edition of the Space Propulsion Conference, Glasgow, Scotland.*, May 2024.

24 K. Lunde, R. Lunde, and B. Münker. Model-based failure analysis with rodon. In *ECAI 2006*, pages 647–651. IOS Press, 2006.

25 I. Matei, M. Zhenirovskyy, J. de Kleer, and A. Feldman. Classification-based Diagnosis Using Synthetic Data from Uncertain Models. *Annual Conference of the PHM Society*, 10(1), 2018.

26 O. J. Mengshoel. Designing resource-bounded reasoners using bayesian networks: System health monitoring and diagnosis. In *Proc. of the 18th Int. Workshop on Principles of Diagnosis (dx-07)*, pages 330–337, 2007.

27 A. Metodi, R. Stern, M. Kalech, and M. Codish. Compiling Model-Based Diagnosis to Boolean Satisfaction. In *26th AAAI Conf. on Artificial Intelligence*, pages 793–799, 2012.

28 L. Moddemann, H. Steude, A. Diedrich, I. Pill, and O. Niggemann. Extracting Knowledge using Machine Learning for Anomaly Detection and Root-Cause Diagnosis. In *29th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, 2024. to appear.

29 S. Narasimhan and L. Brownston. HyDE – A general framework for stochastic and hybrid modelbased diagnosis. *Proc. Int. Workshop on Principles of Diagnosis*, 7:162–169, 2007.

30 I. Nica, I. Pill, T. Quaritsch, and F. Wotawa. The Route to Success – A Performance Comparison of Diagnosis Algorithms. In *23rd International Joint Conference on Artificial Intelligence*, pages 1039–1045, 2013.

31 P. Odgaard and J. Stoustrup. Results of a wind turbine FDI competition. *IFAC Proceedings Volumes*, 45(20):102–107, 2012.

32 P. Odgaard, J. Stoustrup, and M. Kinnaert. Fault-tolerant control of wind turbines: A benchmark model. *IEEE Transactions on control systems Technology*, 21(4):1168–1182, 2013. doi:10.1109/TCST.2013.2259235.

33 I. Pill and T. Quaritsch. Behavioral diagnosis of LTL specifications at operator level. In *23rd Int. Joint Conf. on Artificial Intelligence*, pages 1053–1059, 2013.

34 I. Pill and T. Quaritsch. RC-Tree: A variant avoiding all the redundancy in Reiter's minimal hitting set algorithm. In *IEEE Int. Symp. on Software Reliability Engineering Workshops (ISSREW)*, pages 78–84, 2015.

35 I. Pill, T. Quaritsch, and F. Wotawa. Parse tree structure in LTL requirements diagnosis. In *2015 IEEE Int. Symp. on Software Reliability Engineering Workshops*, pages 100–107, 2015.

36 I. Pill and F. Wotawa. Spectrum-Based Fault Localization for Logic-Based Reasoning. In *2018 IEEE Int. Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 192–199, 2018.

37 I. Pill and F. Wotawa. Exploiting observations from combinatorial testing for diagnostic reasoning. In *30th Int. Workshop on Principles of Diagnosis*, 2019.

38 I. Pill and F. Wotawa. Extending Automated FLTL Test Oracles with Diagnostic Support. In *IEEE Int. Symp.on Software Reliability Engineering Workshops*, pages 354–361, 2019.

39 I. Pill and F. Wotawa. Computing Multi-Scenario Diagnoses. In *31st Int. Workshop on Principles of Diagnosis*, 2020.

40 S. Poll, J. de Kleer, R. Abreau, M. Daigle, A. Feldman, D. Garcia, and A Sweet. Third international diagnostics competition–DXC'11. In *Proc. of the 22nd Int. Workshop on Principles of Diagnosis*, pages 267–278, 2011.

41 S. Poll, A. Patterson-Hine, J. Camisa, D. Garcia, D. Hall, C. Lee, O. Mengshoel, C. Neukom, D. Nishikawa, J. Ossenfort, et al. Advanced diagnostics and prognostics testbed. In *DX Int. Workshop on Principles of Diagnosis*, pages 178–185, 2007.

42 R. Reiter. A Theory of Diagnosis from First Principles. *Art. Intelligence*, 32(1):57–95, 1987. doi:10.1016/0004-3702(87)90062-2.

43 I. Roychoudhury, G. Biswas, and X. Koutsoukos. Designing distributed diagnosers for complex continuous systems. *IEEE Trans. on Automation Science and Engineering*, 6(2):277–290, 2009. doi:10.1109/TASE.2008.2009094.

44 P. Simontacchia, R. Blasi, Edeline E., S. Sagnier, , A. Espinosa-Ramos, J. Breteau, and P. Altenhöfer. PROMETHEUS: Precursor of new low-cost rocket engine family. In *Proc. of the 8th European Conf. for Aeronatuics and Space Sciences (EUCASS)*, 2019.

45 W. A. Smith and R. B. Randall. Rolling element bearing diagnostics using the case western reserve university data: A benchmark study. *Mechanical Systems and Signal Processing*, 64-65:100–131, 2015.

46 SpaceX. *Falcon User's Guide*. Space Exploration Technologies Corp., September 2021. (visited on 05/09/2025). URL: `https://www.spacex.com/media/falcon-users-guide-2021-09.pdf`.

47 A. Sweet, A. Feldman, S. Narasimhan, M. Daigle, and S. Poll. Fourth international diagnostic competition–DXC'13. In *Proc. of the 24th Int. Workshop on Principles of Diagnosis*, pages 224–229, 2013.

48 M. Syfert. Cyber-attack scenarios for super-heaters system, March 2023. `doi:10.5281/zenodo.7612269`.

49 M. Syfert, P. Wnuk, A. Sztyber-Betley, and M. Pobocha. The Model of Ongoing Diagnosis of Process Faults and Detection of Cybernetic Attacks for a Steam Line. *Acta Physica Polonica A*, 146(4):438, 2024.

50 A. Sztyber, E. Chanthery, and L. Travé-Massuyès. Benchmark for fault diagnosis of water distribution network. In *34rd Int. Workshop on Principle of Diagnosis – DX 2023*, pages 1–8, 2023.

51 A. Sztyber, E. Chanthery, L. Travé-Massuyès, and C. G. Pérez-Zuñiga. Water network benchmarks for structural analysis algorithms in fault diagnosis. In *33rd Int. Workshop on Principle of Diagnosis – DX 2022*, 2022.

52 A. Sztyber, Z. Górecka, J. M. Kościelny, and M. Syfert. Controller modelling as a tool for cyber-attacks detection. In Z. Kowalczuk, editor, *Intelligent and Safe Computer Systems in Control and Diagnostics*, pages 100–111, Cham, 2023. Springer International Publishing.

53 A. Sztyber-Betley, M. Syfert, J. M. Kościelny, and Z. Górecka. Controller Cyber-Attack Detection and Isolation. *Sensors*, 23(5), 2023. `doi:10.3390/S23052778`.

54 A. E. Tischer and R. C. Glover. Studies and Analyses of the Space Shuttle Main Engine. *Contractor Report, NASA-CR-183593*, 1987.

55 T. Traudt, W. Armbruster, C.r Groll, R. H. Dos Santos Hahn, K. Dresia, M. Börner, S. Klein, D. Suslov, E. Kurudzija, J. Haemisch, M. A. Müller, J. C. Deeken, J. Hardi, and S. Schlechtriem. LUMEN, the test bed for rocket engine components: Results of the acceptance tests and overview on the engine test preparation. In *9th Edition of the Space Propulsion Conference*, May 2024.

56 D. Vranješ, J. Ehrhardt, R. Heesch, L. Moddemann, H. S. Steude, and O. Niggemann. Design Principles for Falsifiable, Replicable and Reproducible Empirical Machine Learning Research. In I. Pill, A. Natan, and F. Wotawa, editors, *35th Int. Conf. on Principles of Diagnosis and Resilient Systems (DX 2024)*, volume 125 of *Open Access Series in Informatics (OASIcs)*, pages 7:1–7:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. `doi:10.4230/OASICS.DX.2024.7`.