SYSAI for System Health Management - a Statistical Framework for the Analysis of Diagnosis Systems

Yuning He[1] and Johann Schumann[2]

[1] *NASA Ames Research Center, Moffett Field 94035 CA, USA*
yuning.he@nasa.gov
[2] *KBR LLC, NASA Ames Research Center, Moffett Field 94035 CA, USA*
johann.schumann@nasa.gov

ABSTRACT

On-board failure diagnosis and health management systems (HMS) are crucial for the operation of complex autonomous aerospace systems. False alarms (false positives, FPs) or false negatives (FNs) can lead to lower system performance or even loss of mission or the autonomous vehicle. Therefore, a careful verification and validation (V&V) is important. Due to the high dimensionality of the system's state space, however, exhaustive testing of the HMS is usually not possible.

In this paper, we present how our SYSAI (System Analysis for Systems with AI components) framework can support intelligent analysis and testing of HMS on the system level. SYSAI's capabilities to efficiently explore high-dimensional state and parameter spaces and to identify diagnosability regions and their boundaries, makes a comprehensive analysis of the diagnosis system possible and can provide feedback to the designer. We will illustrate our approach using the ADAPT (Advanced Diagnostics and Prognostics Testbed) redundant power storage and distribution system.

1. INTRODUCTION

A Health Management Systems (HMS) on board an autonomous vehicle has to continuously monitor the system's components and behavior to detect anomalies and to identify and diagnose faults.

For systems with a high degree of autonomy, the on-board estimation of system health is extremely important. Only then, the autonomous system obtains knowledge about its current health status and capabilities. HMS therefore are key to support autonomous decision-making and contingency planning to ensure that the mission can be executed safely and successfully even in the presence of adverse events.

Numerous different approaches for fault detection, diagnosis, and system health management have been developed (Abid, Khan, & Iqbal, 2021; Gertler, 2021). They use vastly different techniques and algorithms but share one commonality: undetected or misdetected faults can lead mission failure and potential loss of the autonomous system. Unnecessary alarms (false positives) can hamper mission success but might have more severe consequences as well. In many cases, such situations comprise a safety risk, which even could jeopardize human life.

Therefore, the Health Management System of an autonomous vehicle need to be considered a safety-critical component, requiring careful design, verification and validation (V&V) and possibly certification.

Formal-methods-based approaches, like model-checking can be used for the verification of the discrete fault detection and diagnosis components (Cimatti, Pecheur, & Cavada, 2003). Realistic testing of the entire HMS in conjunction with the vehicle itself, as it is done in scenario-based testing faces large and high-dimensional search spaces that need to be explored during testing; exhaustive testing is not possible.

In this paper, we present, how our SYSAI (System Analysis using Statistical AI) framework can support the V&V of an on-board health management system. SYSAI (He, 2015; He & Schumann, 2020; He, Yu, Brat, & Davies, 2022) has been designed for the concise analysis of complex systems with AI components. It executes the system under test (SuT), the entire autonomous system with its environment or just a single component in a parametric way. The use of advanced surrogate models and active learning allows SYSAI to efficiently explore high-dimensional state and scenario spaces while automatically focusing on relevant regions like failure regions and their boundaries.

In this paper, we describe, how SYSAI can support automatic scenario testing of a HMS within its autonomous vehicle and operational environment, i.e., on the system level. This in-

cludes the analysis of signal filtering and thresholding within the HMS as well as temporal behavior of the system and its signals.

We illustrate our verification & validation approach using the ADAPT (Advanced Diagnostics and Prognostics Testbed) test bed, a power/distribution system, which was developed at NASA ARC, together with a simple discrete diagnosis system (e.g., based upon a diagnosability matrix).

The remainder of the paper is structured as follows: in Section 2, we present our case study, ADAPT, an existing model for an on-board power distribution system with a diagnostic system. Section 3 provides background of the SYSAI system and presents our customized SYSAI architecture to support V&V of diagnosis and health management systems. In Section 4, we discuss requirements for the analysis and V&V for health management systems, present customized analysis metrics, and present some illustrative analysis results. Section 5 covers related work before Section 6 summarizes and discusses future work.

2. CASE STUDY: ADAPT

In general, a health management or disgnostic system has a high-level architecture as shown in Figure 1: the ”plant”, which needs to be monitored and which has sensors that can measure (some of) its internal state. Such a plant could be a subsystem of a vehicle, e.g., a battery power distribution system for an electric UAS, or a hydraulic system with valves pipes, and tanks.

While operating, the plant’s sensors provide measurements, which are sent to the health management systems on the right hand side of Figure 1. There, the signals are typically filtered, conditioned, preprocessed, and discretized before sent to the ”diagnostic reasoner”. Its task is to ‘‘make sense’’ of the signal settings and to produce hypotheses about the actual failure condition. There are numerous approaches for diagnostic reasoning, ranging from simple Boolean logic (e.g., the cFS/cFE Limit Checker (McComas, 2012)). diagnosability-based D-matrix reasoning (e.g., TEAMS/RT (Qualtech,)), Bayesian networks, or neural network based approaches.

Regardless of the actual diagnosis algorithm, the approaches have in common:

- diagnostic reasoning is based upon sensor measurements (signals) arising from the system to be diagnosed,
- usually a large portion of the internal system state is unobserved or unobservable,
- the diagnostic reasoner has to derive hypotheses about the system’s failure mode(s) based upon the information provided by the measurement signals and potentially external commands to the system, and
- the diagnostic reasoner is a complex system is often designed in a model-based manner, based upon design information, physical laws, and engineering models. Alternatively, the diagnostic reasoner can be developed using Machine-learning or AI techniques (Lei et al., 2020), which requires large amounts of system data to train the reasoner.

In this paper, we present a small case study of a complex on-board system, a redundant electrical power distribution system, as can be found on most modern aerospace systems.

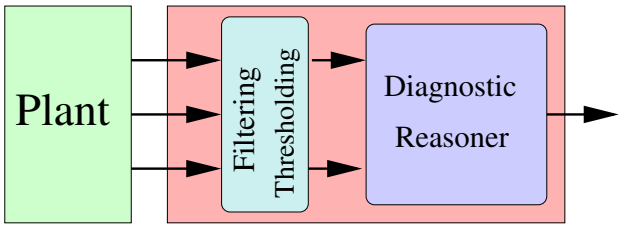

Figure 1. General architecture for a diagnosis application: The plant to be monitored (left); sensor signals are preprocessed/filtered and discretized before fed into the diagnostic reasoner.

2.1. Electric Distribution System

The ADAPT redundant electric distribution system has been developed as a demonstration system for an on-board electrical system. Figure 2 shows the overall architecture and its schematics: there are three different batteries, which serve as independent power-storage units (left column of Figure 2). The available electric power used by consumers in two different load banks (right). They could be lamps, fans, or other power-consuming components. Some of the loads use battery voltage (e.g., 12V), other loads operate on 110V AC, which is produced by one of the two power inverters (INV1, INV2). The entire system has numerous fuses and can be, during operation, reconfigured using relays. Information about the system’s state is provided by a number of voltage and current meters. The legend in Figure 2 shows the available types of signals. These signals produce analog values, which need to be discretized to yield discrete Boolean values for diagnosis (e.g., `U_BATT_LOW` might correspond to $U_{batt} < 11.2V$).

In this paper, we focus on a small excerpt of the circuit diagram ‘‘ADAPT-Lite’’ (enclosed by a dashed line), which routes the battery power from Battery 2 via a power inverter (12V to 110V) to an electric fan (load) in Bank 2. Other loads in Bank 2 can also be considered for our analysis.

A physical realization of the test bed (Poll et al., 2007) has been built and a detailed physicsbased simulation model in Simulink with simulated failure injection is available (NASA,). For each of the control components like fuses and relays, a failure of type ‘‘stuck open’’, ‘‘stuck closed’’, or ‘‘stuck at X%’’ can be injected at any time of the simulation. A simulation run produces measurement values for all individual simulated sensors and components as time series data.

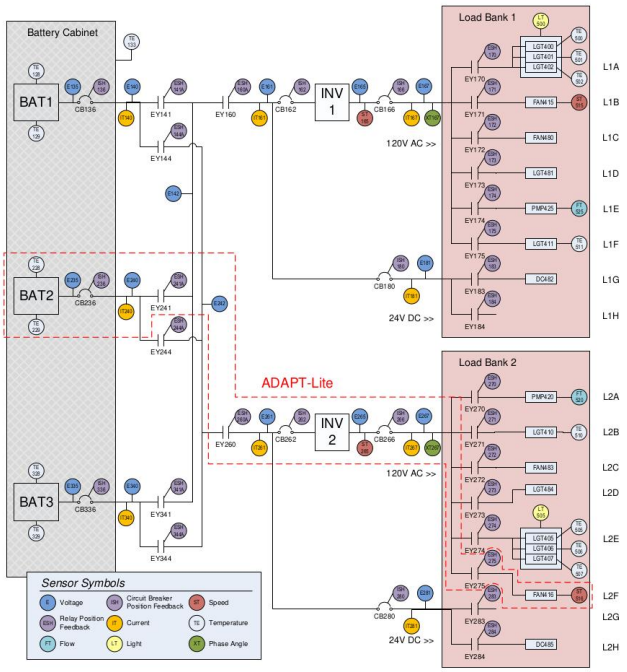

Figure 2. The ADAPT electric distribution system (from (Kurtoglu, Jensen, & Poll, 2009)). ADAPT-light schematics enclosed by a red dashed line.

For our case study, we use a simple diagnosis system, a Diagnosability-matrix based diagnostic reasoner (Mahadevan, Lowry, Schumann, & Karsai, 2016). Other diagnostic reasoners that have been developed for ADAPT include, e.g., a Bayesian Network based system (Knox & Mengshoel, 2009).

3. THE SMART ANALYSIS FRAMEWORK SYSAI

SYSAI (System Analysis for Systems with AI components) (He & Schumann, 2020) is a flexible statistical learning framework for V&V and the analysis of complex and high-dimensional cyber-physical systems, which can have machine-learning based components. Figure 3 shows the high-level architecture of SYSAI analysis framework as it is configured for our purposes here. On the left-hand side, we have the ‘‘system under test’’ (SuT), consisting of the plant (system simulator) and the diagnosis system (see Figure 1). The SuT is surrounded by a scenario generator to control the SuT and a post-processing module for the diagnosis and system signals.

The core of SYSAI is shown in the middle and right side of Figure 3, its core being components for constructing a statistical learning model and performing analysis algorithms based upon Computer Experiment Design and Active Learning (He & Schumann, 2020; He, 2015, 2012), The SuT is executed given a set of parameters provided by the statistical learning model of SYSAI. The result of the test run is then used to incrementally construct the statistical model. The interface between SYSAI and the SuT is designed to be very flexible and generic and has been specifically designed for the analysis of diagnostic components on a system level.

More specifically, the architecture around the SuT consists of the following components:

- the simulator for the ”plant” under consideration
- a scenario generator that produces a temporal scenario, given parameter values produced by SYSAI. The scenario generator uses these parameter values to instantiate a time series of command values for the system.
- measurements and diagnostics output post-processor: the outputs of the diagnostics systems (usually: health state of components (as good, bad, suspect, or unknown), or health probabilities for each component is converted into a function value between 0 and 1, which then will be used by SYSAI to generate new test points. The measurement data from the sensors are also post-processed to yield a function value between 0 and 1 for use by SYSAI. Operations can include thresholding, filtering, or the use of customized metrics as discussed below.

For the representation and construction of the statistical model, SYSAI uses Dynamic Regression Trees (DynaTrees (Taddy, Gramacy, & Polson, 2011; Gramacy & Polson, 2011)), a dynamic Gaussian process model based upon Particle Filters. DynaTrees are regression and classification learning models with complicated response surfaces for on-line application settings. DynaTrees create a sequential tree model whose state changes over time with the accumulation of new data, and provide particle learning algorithms that allow for the efficient on-line posterior filtering of tree-states. A major advantage of DynaTrees is that they allow for the use of simple models within each partition. The models also facilitate a natural division in sequential particle-based inference: tree dynamics are defined through a few potential changes that are local to each newly arrived observation, while global uncertainty is captured by the ensemble of particles.

This surrogate model is initialized with available training data and incrementally refined using candidate data points that are produced by our active learning module. It evaluates the current surrogate model using a customized active-learning heuristics and suggests candidate data points that provide most information for model refinement. For these candidate points, the ground truth is obtained by executing the SuT.

SYSAI features customizable heuristics that allow the active learning to focus on particular characteristics of the model. Classical algorithms like ALM (MacKay, 1992) or ALC (Cohn, 1996) focus on under-explored regions in general of the domain space. Inspired by (Jones, Schonlau, & Welch, 1998) and work on contour finding algorithms, we loosely follow (Ranjan, Bingham, & Michailidis, 2008) and define our boundary-aware metric boundary-EI (He, 2015, 2012) that puts the

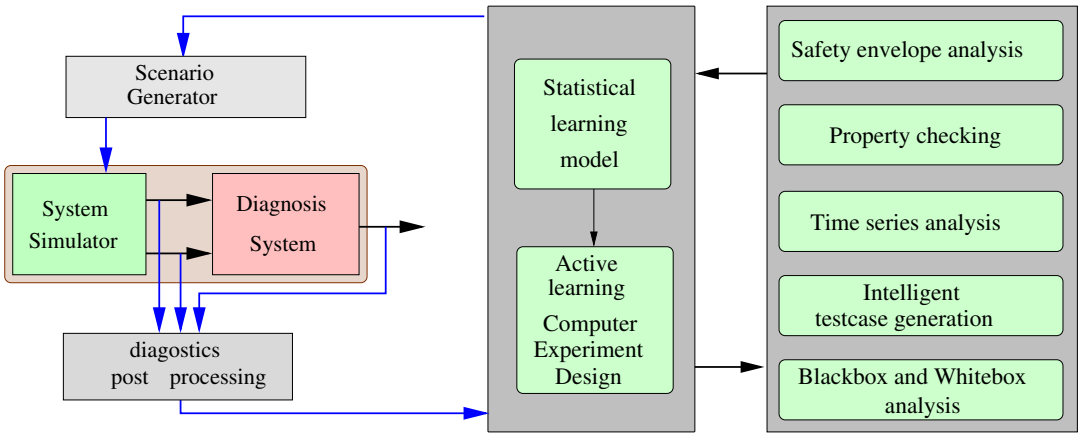

Figure 3. SYSAI architecture

focus of the search into ‘‘interesting’’ and potentially ‘‘troublesome’’ areas near safety boundaries. Here, the surrogate model therefore exhibits substantially more details than in other areas that are not of interest. This exploration is guided by the selected active learning heuristics and is able to cover the entire input space with a low number of data points. The SYSAI framework has been used for the analysis of several complex and safety-critical aerospace systems (He, 2015; He et al., 2022; He, Yu, Brat, & Davies, 2021).

In general, SYSAI supports the following analysis tasks:

- *statistical analysis of training and test data:* The feature supports the analysis of ML-based diagnosis systems that require training and test data. SYSAI can produce a detailed statistical analysis of the data sets used for training and evaluation of the ML-based diagnosis system (not used for this paper).
- *envelope analysis:* our framework can perform automatic analysis of failure regions, which indicate under which operational conditions the system produces failure modes that are correctly detected by the diagnosis system under test. Geometric shape modeling does not only identify but also characterizes regions with similar behavior and describes those regions in easy to understand geometrical terms. SYSAI thus helps to make the diagnosis system more explainable.
- *property checking:* our tool supports the automatic checking and analysis of properties and requirements concerning the diagnosis system. E.g., a property might require that a battery-low alarm is always raised in case of a weak battery but no overload conditions.
- *time-series analysis:* SYSAI can perform advanced time-series analysis in a high-dimensional parameter and state space. This analysis provides a deeper understanding of the system behavior and its dynamics. The tool also supports event prediction. In conjunction with specific metrics, introduced below, the behavior of a (discrete) diagnosis system can be studied in the presence of system dynamics (e.g., slow responding sensors, measurement decay, transients).
- *intelligent test-case and scenario generation:* SYSAI can efficiently generate relevant scenarios in high-dimensional spaces.
- *White-box analysis:* SYSAI usually performs analysis on the system level, i.e., considering the diagnosis system as a Black Box. However, it can be configured to have access to inner details of the diagnoser (white-box analysis), or to perform analysis of the diagnosis component individually.

In this paper, we focus our analysis on system behavior under failures. This means that the high-dimensional input space for our analysis can contain parameterized models of system or component failures, as well as off-nominal environmental and operational conditions.

4. SYSAI ANALYSIS

4.1. Analyses for Diagnosis Systems

For detailed analyses of the system behavior and the behavior of the diagnostics component, the high-dimensional space of failure modes must be explored. Typically this space is spanned by possible failures for each component. E.g., if our hydraulic system has 5 different valves, a 5-dimensional space is spanned; values along each axis correspond, for example, to the percentage a valve is stuck open. Since a systematic exploration is not possible, more effective methods need to be used to obtain a reasonable coverage on important parts of the space. (Mahadevan et al., 2016) use n-factor combinatorial exploration to restrict the number of test cases to be executed.

SYSAI uses active learning and a boundary-aware metric to focus on the exploration of ‘‘interesting’’ regions, e.g., boundary areas between different failure modes. In contrast to the

dynamic exploration of SYSAI, the approaches in (Mahadevan et al., 2016; Schumann et al., 2014) first generate all test cases and then executes them in a batch mode.

For a system and its diagnostic reasoner, the following analyses are important:

- Failure-mode region analysis: for a properly designed system, only certain combinations of signals should be recognized as an unambiguous failure mode F. With SYSAI's capability to explore multiple combinations of failure injections,we can determine regions, where F has triggered. Obviously, F should be triggered in regions, where the failure is supposed to occur. Other triggering regions can indication ambiguous failure modes, which requires improvements to the failure model of the diagnoser or additional sensors for disambiguation.
- Failure-mode boundary analysis: the relative size and location of the failure-mode boundary with respect to the designed failure mode region can give insight into performance and correctness of the failure models and diagnostic engine
- Threshold analysis: the selection of suitable thresholds is important for the good performance of the diagnostic system. SYSAI can perform analyses with different threshold settings and can calculate ROC curves for its performance.
- Sensitivity analysis: a proper diagnostics system shall be robust against small changes in failure conditions. E.g., a valve stuck at 50% should trigger the same failure mode as on stuck at 55% (although there might be non-linear effects, which might need to be considered).
- Temporal sensitivity analysis: many diagnostic reasoning systems (e.g., the Bayesian network or the D-matrix diagnoser) do not take into account temporal behavior of signals. However, the distinction between slow and fast changing signals and the existence of transients should be taken into account. With SYSAI, we will use customized signal metrics (see below) to address this issue.

4.2. Metrics

For the analysis of the diagnostics system, obviously the output (failure mode hypothesis) is important. With SYSAI we can, for example, find parameter regions, where a failure mode is not recognized (false negative) or a false alarm is raised. In general, the performance of a diagnostics system is measured using the typical metrics of True Positives (TP_D), False Positives (FP_D), False Negatives (FN_D), and True Negatives (TN_D). The Accuracy A is then calculated as

$$A = \frac{|TP_D| + |TN_D|}{|TP_D| + |TN_D| + |FP_D| + |FN_D|}$$

and a Comprehensive Diagnosis Metric (CDM), which is calculated as a weighted average of the A_k for each fault k. The weight w_k can be used to express the ''importance'' of fault k:

$$CDM = \frac{\sum_k A_k}{\sum_k w_k}$$

However, these metrics are very coarse and only cover a part of the picture. For a more detailed sensitivity and robustness analysis, the characteristics of individual signals need to be considered.

We therefore follow (Mahadevan et al., 2016) and add provide additional signal metrics that can help with the analysis of plant behavior and diagnostics system.

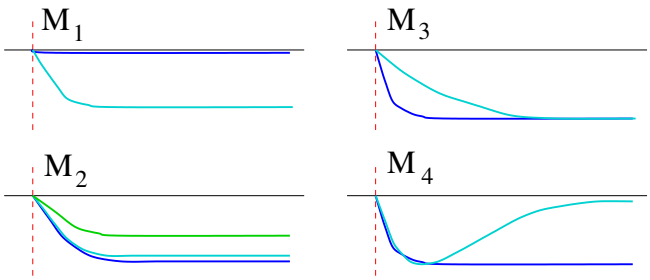

Figure 4. Signal metrics $M_1, ..., M_4$ for signals over time.

Figure 4 shows metrics $M_1, \ldots, M_4$ regarding the temporal development of a signal s with respect to a nominal signal s_0: The metric M_1 indicates, how much the signal value changes after the event, calculated as the relative change $|s - s_0|/s_0$. Only signals, which have a decent relative change should be used for diagnostic purposes. SYSAI can, for example, find regions, where an alarm cannot be triggered, because the signal change is not enough, despite the fact that the discrete reasoning system is working perfectly in this situation. SYSAI can determine suitable thresholds that allow for reliable diagnoses under given operational constraints, or can also check diagnosis reliability.

M_2 describes how much the signal s is changing in different failure scenarios. Only if there is a considerable difference in s between different scenarios, it is possible to distinguish diagnosis in these different cases using signal s. Resulting ambiguities require adjustment of filtering and thresholding. If the scenarios can be distinguished despite small signal variations of s, that might be an indication that signal s is not relevant for distinguishing the faults or that the diagnosis system likely has litte robustness.

M_3 is concerned with rise and fall times of signal s. Here, we usually consider the rise or fall times of s to 95% of the final value. This metric is important to determine, when the signals have stabilized and a reliable diagnosis can be expected. In particular, if signals with different rise or fall times are being combined during diagnostic reasoning, transients and other problems might occur.

Finally, M_4 indicates if the changes of signal s remains stable, if it is a transient signal, or a decaying signal. For analysis, we usually measure the time, signal s is changed more than 80% of the peak change. This metric is important, when diagnosis depends on a combination of signals. E.g., a fault mode requiring s_1 and s_2 being low might not be recognized properly if s_1 is dropping slowly, while s_2 is a transient signal dropping fast and recovering even before s_1 has dropped sufficiently to hit the threshold. Thus this situation would result in a missed alarm.

Other metrics can be defined and customized. For example, a metric concerning noise magnitude and characteristics can be used to analyze robustness of the diagnosis system under sensor noise. For a typical SYSAI analysis, these metrics, calculated for relevant signals would be combined with the desired diagnostic outcome to yield the function to be evaluated by active learning.

5. ANALYSES

Figure 5 shows typical results for a multi-dimensional parameter analysis. For visualization, here only two parameters are shown. SYSAI has been analyzing the space, spanned by two parameters, p1, and p2, e.g., battery currents for Battery 1 and 2. Failure mode F1 (shown in green) and F2 (in blue) can be easily recognized. In Figure 5A, both failure modes can be recognized unambiguously, whereas in Figure 5B, an overlap occurs (shown in red).In this area, the diagnosis always will be ambiguous, i.e., F1 or F2. Since such situations need to be avoided, changes to the system (e.g., additional or different sensors) or changes to the diagnostic models might be needed. Here, SYSAI analysis results can provide valuable feedback to the designers of the system and the diagnosis system.

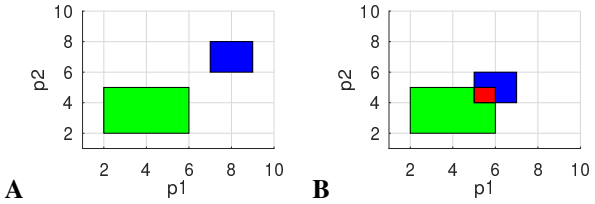

Figure 5. Typical diagnostic results of parametric analysis in two parameters p_1, p_2

In contrast to systematic, grid-based testing or Monte Carlo testing, SYSAI produces test cases that are close to the suspected boundaries of the regions of interest. This dramatically reduces the number of required simulation runs and makes efficient analysis in high-dimensional spaces possible.

Figure 6 shows how SYSAI explores the search space. For this experiment, the failure mode $U_{batt} < 24.1V$ is analyzed. In nominal mode, the system is using two batteries to drive the voltage inverter and loads as shown in Figure 2. All relays are closed, so the inverter is producing 120V power, which is consumed by potential loads. In our failure scenario, at $t_0 = 10s$, one of the batteries is disconnected. This causes that the entire power is drawn from only one battery, thus rapidly discharging it and lowering its output voltage U_{batt}. In this scenario, some of the loads are turned off at $t = t_0 + \Delta t$ (Δt between 5 and 35 seconds) to allow the battery to recover. With the analysis, we want to know, which initial batter voltages U_{batt} and Δ causes triggering the low-voltage failure.

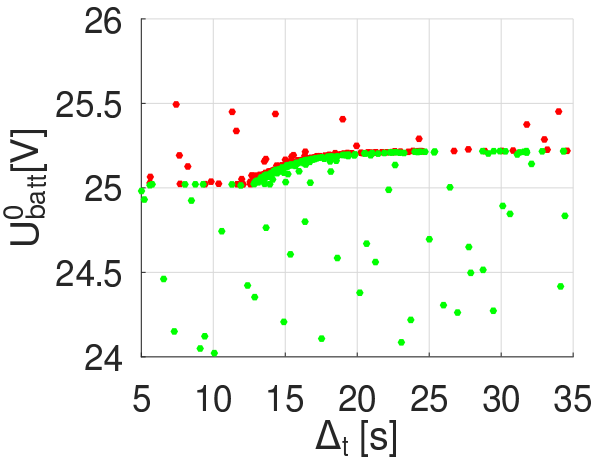

Figure 6. Failure mode regions over parameter U_{batt} and load cut-off time Δt

Figure 6 shows the result of this simple analysis: each dot corresponds to one experiment[1] Red dots correspond to ''low-voltage'' alarm not triggered. It can be seen clearly that the boundary regions is covered closely with test cases, whereas for other regions, only a sparse set of test cases is sufficient.

With this result, we see that sufficient battery voltages make a substantial different if the loads are turned off within less that 10 seconds or more than 10 seconds. We can also observe that the boundary line is non-linear.

Finally, Figure 7 show the raw point cloud produced by SYSAI if, in addition to the parameters above, also the amount of power cut W (0=no loads cut, 5=100% power cut) has been varied. Here again it can be seen that SYSAI is able to focus its exploration on the relevant part of the space.

6. RELATED WORK

Verification and testing of diagnosis and health management system is an important task and numerous approaches exist. V&V processes for ensuring the effectiveness and reliability of fault diagnosis systems are discussed in (Barua & Khorasani, 2012). Verification must ensure that the HMS software correctly implmements the specified requirements. Techniques

[1] SYSAI first performs an initial MC analysis with few tests (shown as circles) before the active learning starts.

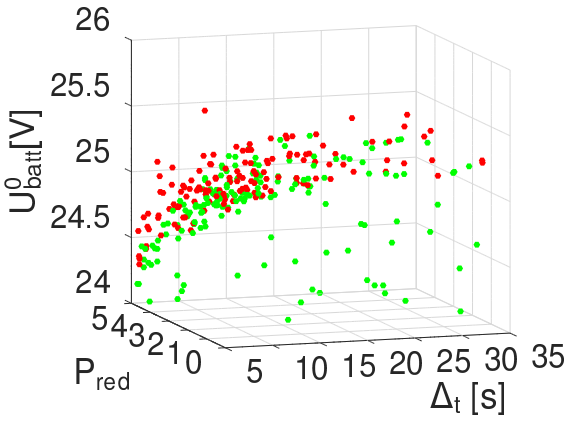

Figure 7. Failure mode regions over parameter U_{batt}, load cut-off time Δt, and amount of power reduction W

typically include static verification, in particular model checking (Clarke, Grumberg, & Peled, 2000). There, efficient algorithms are used to explore the space of a discrete diagnosis system, where the inputs are usually comprised of Boolean or discrete values (e.g., 'success', 'failure', 'unknown'). Other formal verification methods have been advocated for critical fault-tolerance algorithms to ensure system reliability in (Lincoln & Rushby, 1993). Reachability analysis is used in, e.g., (Su & Chen, 2019).

Our SYSAI framework is designed to support validation and the analysis of system diagnosability (see e.g., (Batteux, Dague, Rapin, & Fiani, 2011) for an overview). Here the main approaches are simulation-based testing and hardware-in-the-loop (HIL) validation. The authors emphasize the need to simulate rare but critical faults, such as sensor drifts or actuator malfunctions, to test the system's ability to detect and respond to faults that may not occur frequently but could have severe consequences. This requires elaborate mechanisms to inject simulated faults into both software and hardware components, e.g., for Simulink described in (Moradi, Van Acker, Vanherpen, & Denil, 2019). Typically, V&V processes involve testing the fault diagnosis models using synthetic data to ensure robustness and accuracy (Chen & Wu, 2007). In (Farsoni & Simani, 2021), validation of fault diagnosis techniques based on AI tools is discussed.

7. CONCLUSIONS

In this paper, we presented our approach, how the SYSAI tool and statistical framework can support advanced analysis and evaluation of systems with diagnostic components. SYSAI's capabilities to efficiently explore high-dimensional parameter spaces and to identify boundaries between regions of interest is an important prerequisite that enables practical analysis. In addition to evaluate the reasoner with respect to false positives and false negatives, the parametric SYSAI search can be used to analyze temporal system behaviors and might help to uncover weaknesses in the diagnostic models.

REFERENCES

Abid, A., Khan, M. T., & Iqbal, J. (2021). A review on fault detection and diagnosis techniques: basics and beyond. *Artificial Intelligence Review*, *54*(5), 3639--3664.

Barua, A., & Khorasani, K. (2012). Verification and validation of hierarchical fault diagnosis in satellites formation flight. *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)*. doi: 10.1109/tsmcc.2012.2187188

Batteux, M., Dague, P., Rapin, N., & Fiani, P. (2011). Diagnosability study of technological systems. doi: 10.1007/978-3-642-21822-4_20

Chen, Y.-Y., & Wu, G.-W. (2007). Fault-tolerant verification platform for systems modeled at high level of abstraction. doi: 10.1109/systems.2007.374697

Cimatti, A., Pecheur, C., & Cavada, R. (2003). Formal verification of diagnosability via symbolic model checking. In Proc. IJCAI (p. 363-369).

Clarke, E. M., Grumberg, O., & Peled, D. A. (2000). *Model checking*. Cambridge, MA, USA: MIT Press.

Cohn, D. A. (1996). Neural network exploration using optimal experimental design. *Advances in Neural Information Processing Systems*, *6*(9), 679--686.

Farsoni, S., & Simani, S. (2021). Validation of fault diagnosis techniques based on artificial intelligence tools for a wind turbine benchmark. In *2021 5th international Conference on Control and Fault-tolerant Systems (SYSTOL)* (p. 157-162). doi: 10.1109/SysTol52990.2021.9595291

Gertler, J. (2021). Fault detection and diagnosis. In *Encyclopedia of Systems and Control* (pp. 764--769). Springer.

Gramacy, R., & Polson, N. (2011). Particle learning of Gaussian process models for sequential design and optimization. *Journal of Computational and Graphical Statistics*, *20*(1), 467--478.

He, Y. (2012). *Variable-length functional output prediction and boundary detection for an adaptive flight control simulator* (Doctoral dissertation). University of California at Santa Cruz.

He, Y. (2015). Online detection and modeling of safety boundaries for aerospace applications using active learning and bayesian statistics. In *2015 International joint Conference on Neural Networks, IJCNN 2015* (pp. 1--8). IEEE. Retrieved from `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7256526` doi: 10.1109/IJCNN.2015.7280595

He, Y., & Schumann, J. (2020). A framework for the analysis of deep neural networks in aerospace applications using bayesian statistics. In *Proc. IJCNN, WCCI*.

He, Y., Yu, H., Brat, G., & Davies, M. (2021). Statistical

learning framework for safety and failure analysis of a DNN-based autonomous aircraft system. In Proc. International Conference on Machine Learning Applications (ICMLA), IEEE.

He, Y., Yu, H., Brat, G., & Davies, M. (2022). System and safety analysis for autonomous center line tracking with SYSAI. In SciTech 2022,

Jones, D., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black box functions. *Journal of Global Optimization*, *13*, 455--492.

Knox, W., & Mengshoel, O. (2009). Diagnosis and reconfiguration using bayesian networks: An electrical power system case study. In *Proc. IJCAI.*

Kurtoglu, T., Jensen, D., & Poll, S. (2009). Systematic benchmarking of diagnostic technologies for an electrical power system. InIEEE Aerospace Conference (p. 1 - 9). doi: 10.1109/AERO.2009.4839623

Lei, Y., Yang, B., Jiang, X., Jia, F., Li, N., & Nandi, A. K. (2020). Applications of machine learning to machine fault diagnosis: A review and roadmap. *Mechanical Systems and Signal Processing*, *138*, 106587. doi: https://doi.org/10.1016/j.ymssp.2019.106587

Lincoln, P., & Rushby, J. (1993). The formal verification of an algorithm for interactive consistency under a hybrid fault model. doi: 10.1007/3-540-56922-7_24

MacKay, D. J. C. (1992). Information--based objective functions for active data selection. *Neural Computation*, *4*(4), 589--603.

Mahadevan, N., Lowry, M., Schumann, J., & Karsai, G. (2016). Dver: A tool chain for cross-validation and perfection of discrete model-based diagnostic systems. In *2016 IEEE aerospace conference* (p. 1-15). doi: 10.1109/AERO.2016.7500913

McComas, D. (2012). NASA/GSFC's Flight Software Core Flight System. In *Flight software workshop.*

Moradi, M., Van Acker, B., Vanherpen, K., & Denil, J. (2019). Model-implemented hybrid fault injection for simulink (tool demonstrations). In (p. 71-90). doi: 10.1007/978-3-030-23703-5_4

Poll, S., Patterson-Hine, A., Camisa, J., Garcia, D., Hall, D., Lee, C., ... Koutsoukos, X. (2007). Advanced diagnostics and prognostics testbed. In *18th international workshop on principles of diagnosis.*

Qualtech. *TEAMS designer.* Retrieved from `http://www.teamqsi.com/products/teams-designer/`

Ranjan, P., Bingham, D., & Michailidis, G. (2008). Sequential experiment design for contour estimation from complex computer codes. *Technometrics*, *50*(4), 527--541.

Schumann, J., Gomez-Gonzalez, V., Mahadevan, N., Lowry, M., Robinson, P., & Karsai, G. (2014). A tool chain for the v&v of nasa cryogenic fuel loading health management. In *Annual conference of the phm society, 6(1).*

Su, J., & Chen, W.-H. (2019). Model-based fault diagnosis system verification using reachability analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(4), 742-751. doi: 10.1109/TSMC.2017.2710132

Taddy, M. A., Gramacy, R. B., & Polson, N. G. (2011). Dynamic trees for learning and design. *Journal of the American Statistical Association*, *106*(493), 109-123.

VirtualADAPT. Retrieved from `https://github.com/nasa/VirtualADAPT`