Masterarbeit

Name:	Colin	Klein
Matr.-Nr.:	368642	

Thema: FAIR Sensor Health Monitoring of Flight Test Data

Betreuender Assistent: M. Sc. Matthias Bodenbenner

Aachen, den 15.06.2015

Diese Arbeit wurde vorgelegt am Werkzeugmaschinenlabor WZL, Lehrstuhl für Fertigungsmesstechnik und Qualitätsmanagement.

Aachen, 18. May 2023

V. Name - Tel. 0241-80 xxxxx

Master Thesis

for Ms./Mrs. Cand.-Ing. Erika Mustermann

Matriculation number: 081511

Topic: FAIR Sensor Health Monitoring of Flight Test Data.

The start of production for series production represents a major uncertainty and cost factor for both manufacturers and users of automated production systems. In particular, the poor planning of the production start-up requires new approaches that support the early safeguarding of the functionality and performance of automated production systems. Within the BMBF joint project Ramp-Up/2, the step from two-dimensional alphanumeric planning to an integral 3D-based digital verification of plant development and commissioning is aimed at. For this purpose, a kinematic 3D model of the production plant and all control components (NC/PLC) are simulated by virtual NC/PLC software modules and the mechanical behaviour of a machine is depicted by the Siemens Machine Simulator (MS). Based on this virtual production system, the aim of the plant development is to enable a preliminary verification of the production control software. In doing so, technical errors as well as operating and software errors are to be simulated and the reaction of the control software is to be analysed by means of diagnostic tools.

Within the scope of the work, concepts and tools are to be developed which enable the testing of the functionality of a production control software. In particular, the following questions are to be dealt with: which test cases can occur, how errors/tests can be reproduced, which data are necessary for the clear diagnosis of an error and to what extent tools can be used for error correction. Based on these considerations, a concept for information visualization is to be developed and realized exemplarily. The information content as well as the temporal sequence of the information flow within the production control software should be mapped and the possibility should be provided to reset the system into a freely defined state (time). The developed concepts are to be realized exemplarily on the basis of the control software cosmos4. The functionality and performance of the developed tools will be verified using an example scenario in the Integrated Manufacturing and Assembly System (IFMS) of the WZL.

In detail, the following subtasks have to be solved:

- Introduction with the leading software cosmos4
- Development of a comprehensive concept for error diagnosis and correction
- Exemplary realization of a scalable information visualization and recovery
- Documentation of the work

Prof. Dr.-Ing. Robert Schmitt

I Contents

II Acronyms

Symbol	Unit	Description
a_e	mm	Width
a_p	mm	Cutting depth
t		Number of teeth [Note: Sorting is alphabetical]
α		angle

Abbreviation	Description
DP	Polycrystalline diamond
ADC	Analog Digital Converter

III List of Figures

IV List of Tables

1 Introduction

Sensors fail. Failure rates often depend on the sensor as well as the complexity of the system into which the sensor is integrated. Historically, sensors (latin: *sentire*: perceive) are defined as entities that measure physical conditions. This information then gets transmitted by signals that are emitted by the sensors [DIN95; PEÑA23]. Aircraft sensors generally emit an electric current that gets transformed into a digital signal by an Analog to Digital Converter (ADC). This digital signal then flows through other stages until it reaches the Main Aircraft Bus where it can be used by the aircrafts systems.

Within this works scope the ISTAR aircraft (In-flight Systems Technology Airborne Research) gets examined. The ISTAR is a Falcon 2000LX business jet that is equipped with various additional sensors including strain gauges, accelerometers and changing experimental configurations. öö-

What?

The ISTAR aircraft is currently the aircraft generating the most data for the DLR's research aircraft fleet at the Brunswick Research Site. This fleet is Europe's largest, consisting of 12 aircraft employed for research purposes ranging from atmospheric research to testing operating limits that are pushing the aircraft operating envelope. The data acquisitioning system (DAQ) plays a focal role in the generated data since it is the central data hub within the airplane. Making the DAQ as well as the sensor data the central point of this work. In conjunction with the FAIR principles this work also focuses on reusability of the components of this work as well as interoperability with outside algorithms. Allowing the results of this work to be accessible by uploading them to the FAIR skystash data-platform and making condensing errors into tags and using conventions to catalyze digitalization to make the resulting data findable. Hence, making it the goal of this thesis to examine the ISTAR DAQ in detail to detect sensor behaviour anomalies and implement a clear infrastructure to fulfill FAIR principles.

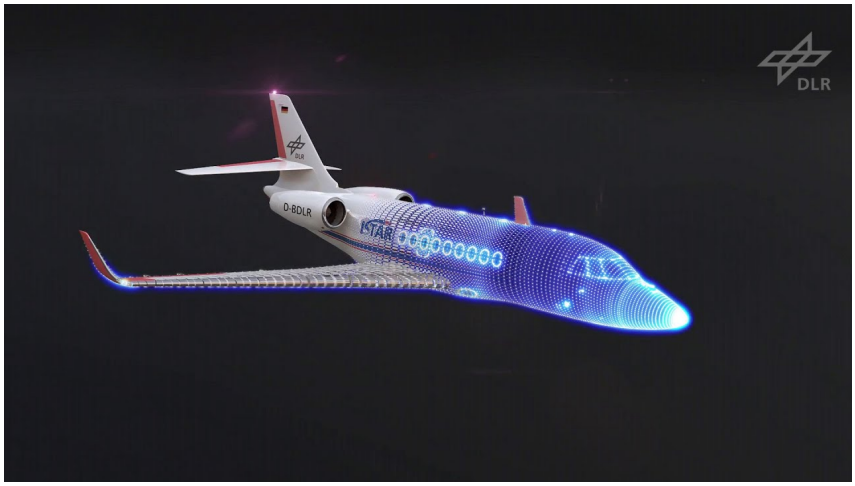


Figure 1.1: The ISTARs sensor data systems will be analysed within this work [DLR20]

Why?

It is then important to reduce manual overview of sensors. Currently, much manual postprocessing is needed to get an overview over the large datasets. Large datasets however are difficult to work with due to being too large for a single computer. Necessitating a server architecture for work with this size of data. Another benefit of this work may be to reduce downtime of the Aircraft due to sensors. A diagnostic tool for detecting sensor behavior can facilitate the processes and allow a quicker follow up to detect sensor errors since currently system information is distributed and not clearly perceivable.

To achieve this it is vital to increase the fault detection rate since sensor faults are frequently not detected). Commonly, a sensor- or system-failure may not be detected during a flight experiment but weeks after. This rescheduling within the aircrafts full timetable makes it even more economically important to implement a system to catch sensor faults and failures to avoid the need for an experiment to be reflown. A worst case scenario could be that a whole experiment needs to be repeated with the configuration needing to be recustomized and refitted which is a time- and funding intensive task. Additionally, it is important to increase reaction time. Often errors are detected weeks after the flight is over. With a software detecting errors directly after sensor data is extracted from the aircraft errors may be detected on the same day. allowing a quicker reaction maybe repeating the experiment during the affixed timeslot.

Safety critical errors may be detected earlier since the proposed routine has potential to go beyond the scope of commercially tested software that is generally employed within aircraft and generally rudimentary resulting from the desire to employ very stable software. Once, a baseline routine is in place. It is then possible to build a foundation for a reliability index for sensors. This is vital for big data operations within digitalization and the skystash project. This should be able to implemented by plug and play into this works results. Allowing for quick implementation of custom data quality algorithms.

Another motivation for this work is the fulfillment of the FAIR (Findable, Accessible, Interoperable, Reusable) principles. FAIR principles are the new standard within the DLR for research data management. The use within operations however are far from the fulfillment. Documents containing descriptions of sensors and setups are distributed among multiple employees obstructing any efforts to comprehend the already complex aircraft system and its inherent generation of data.

The proposed work is then nothing short of essential, building a tool for detection and avoidance of sensor failures to mitigate risks associated with experimental setups since aircraft hours are expensive and the amount of sensors is vast.

Origin of Data

The Flight Test Data by the ISTAR gets generated by a base DAQ system by IMC co. This base contains inputs from the main aircraft data bus, the experimental nose boom, experimental strain gauges and acc sensors as well as an experimental Inertial Measuring Unit (IMU) combined with a GNSS-System. In addition, complementary DAQs are added to fulfill given experiment requirements. In regard to system complexity it is notable that the main aircraft bus data originates from various sources, ranging from air data that gets measured by a

Figure 1.2: The DLRs research fleet [DLR18]

sensor working on an electric current which then gets transformed by an Analog Digital Converter (ADC) into a digital, discretized signal which then gets fed into the main aircraft bus system. Such a signal would then getstransmitted to the IMC DAQ making troubleshooting a multifaceted problem within this system.

How?

To now solve this extensive problem of SHM, it is possible to employ one of many existing methods in the field of Fault Detection and Mode Analysis (FMEA).

FMEA is a common problem in engineering disciplines and everywhere where systems become complex. This problem also gets facilitated by the work that is already happening within the digital twin program within the DLR. A data space is already in development and is supplied with Flight Test Data which aids this work. It is then necessary to allow clear software interfaces to facilitate future extensions and modes to allow reaching a quick overview over sensor errors. FMEA is a rich field in which much energy has already been invested into similar problems. A new challenge arises from the implementation into a dynamic digital system also keeping in mind to remain open for new changes and addons.

Context

This work happens within the DLRs effort to digitize and digitalize research data and update its research data management strategy. Within this project's scope, the development of skystash architecture is advanced to beyond prototypical level, allowing upload and distribution of research data by using the skystash cloud. Within the scope of future work lies the expansion of metadata management on a larger scale, metadata frameworks already exists but challenges arise from indexing and providing a digital replica of long paper trails. Aircraft are highly complex systems that amass a great amount of paperwork since even its tiniest screws require a highly documented certification process.

Digitalization and the digital twin are the inherent adjuncts of this work. A database with data already exists. now it is time to structure metadata allowing for this work to happen dynamically. Building upon this, further

metadata may be fed into the system allowing further analytics to be developed.

Summary

This work will examine the sensor data of the ISTAR aircraft, examining various algorithms and methods for FMEA. Then developing a software that allows dynamic implementations of various FMEAs to generate a dynamic backend facilitating development and finally generating reports on data quality for the aircraft systems. FAIR principles are vital for future legibility and exchangeability of data and hence become a central part of this thesis facilitating future use of the results generated within this work.

2 State of the art and a theoretical Background

Chapter **??** introduces the theoretic baselines for data acquisition, data metrics as well as standardized data formats in which to save metadata.

Topics to consider when starting the Sensor Health Monitoring process are mainly that of providing a structured overview of the Sensor Metadata which in itself consists of many layers as a dynamically generated set of metadata is desired. This should be able to accomodate changing Data Acquisitioning (DAQ) System configuration changes. Consideration is given to the SOIL data model and its' ability to accomodate the many demands that are expected of sensor data management. [BODE21]

The second major part to consider is that of physical crossrelations and d̈eep checks ẅhich are a experimental mode of checking for inconsistencies among the data. Major research and implementation work shall go into developing a dynamic model that is generated from the data and then checks back upon the data for possible discrepancies. This approach is chosen as it is estimated to be the most structured approach for a first prototype.

To solve this problem the infrastructure and general flow shown in figure **??** is proposed.

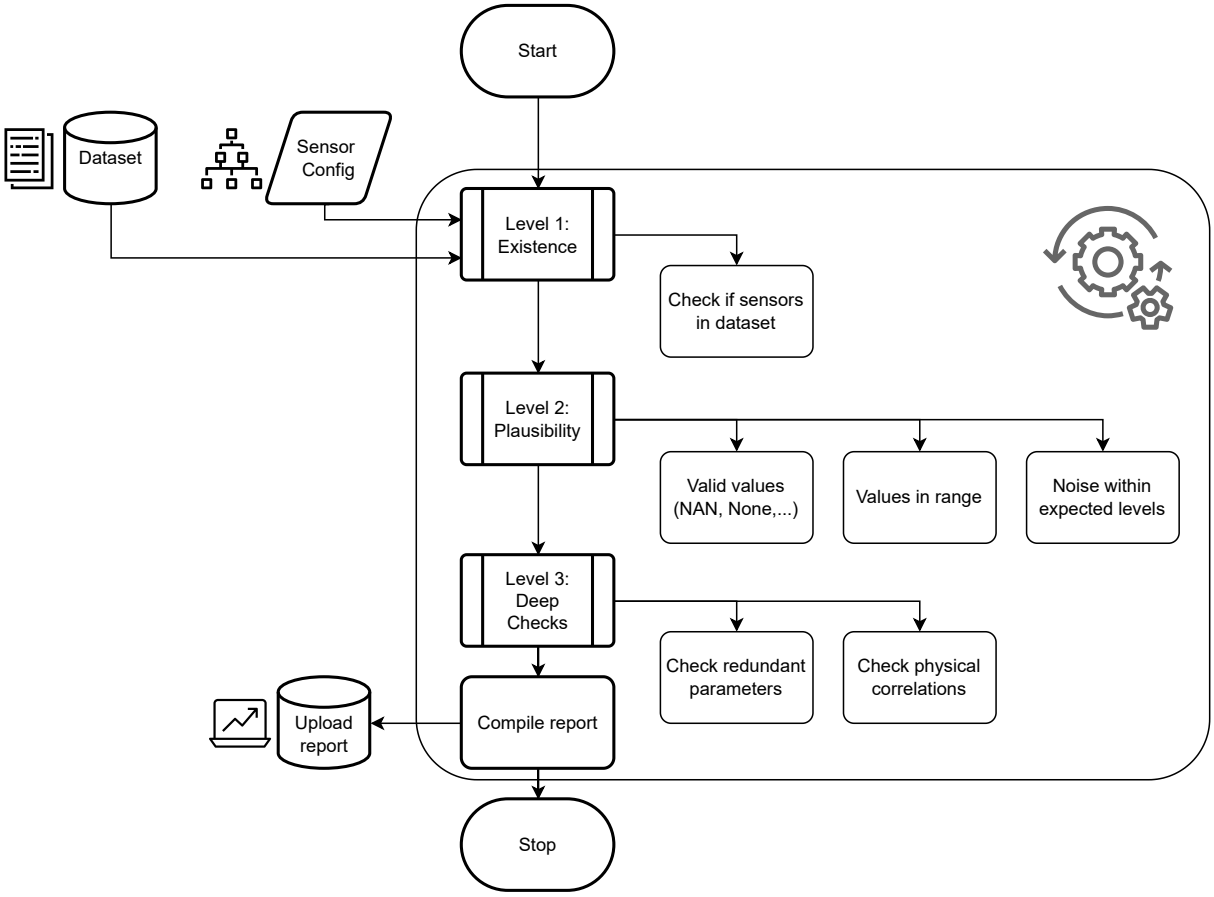


Figure 2.1: The data flow architecture proposed within this work. Starting with the configuration and dataset and finishing with a data quality report

2.1 Introduction

Motivating this chapter is the desire to understand the fundamental error mechanism. Definition for errors are adopted from Isermann [ISER06] defined in table **??**, the de-facto standard of Fault Diagnosis descriptions within the automation domain. Within this source errors are described as the difference between measurement and actual value as seen in figure **??**.

2.1.1 What are Errors?

Sensor are used to measure reality. But how precise do they measure? No sensor is without error, an inevitable property accompanying every sensor application. In the following, sensor errors are classified into categories upon which an understanding about data quality can be built that then can describe the data quality of the ISTAR datasets.

Error Type	Systemic	Random	Failure
Appearance	Deterministic	Stochastic	Stochastic
Occurence	Permanent	Permanent	Occasional
Compensation	Calibration	Filtering	Mixing

Table 2.1: Error Overview [HART22]

Filtering out sensor errors is the goal of this thesis. As described in figure **??** the error can be imagined as a disturbance added upon the original sensor value. Which then results in a skewed sensor measurement.

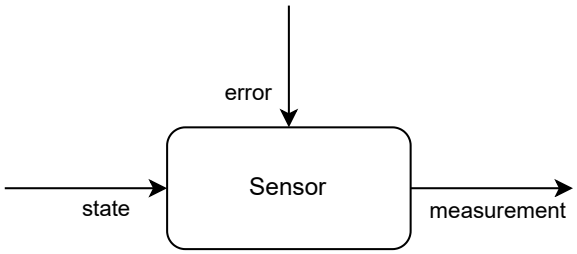


Figure 2.2: Sensor Signal

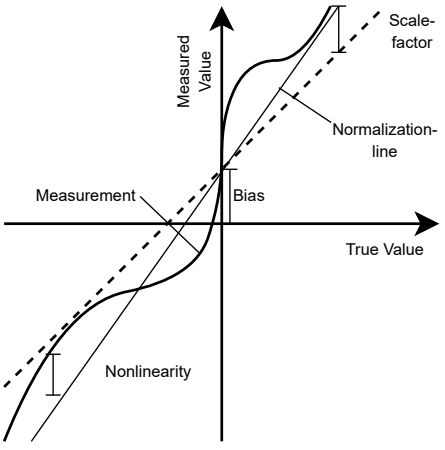


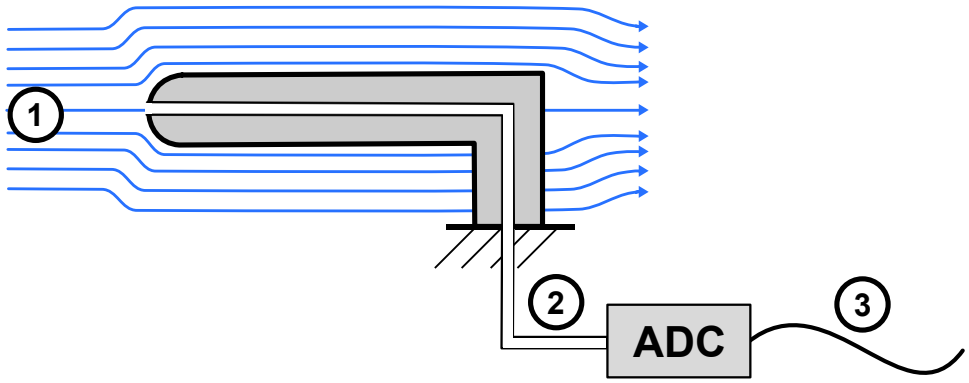
Figure 2.3: Systemic Sensor Errors [HART22; DIN95]

In Table **??** sensor errors are divided into 3 subcategories which are classified by their appearance representing their manageability. Deterministic Errors can be easily compensated since they are reproduceable. Stochastic Errors however are non-deterministic. Figure **??** gives an overview over systemic errors which generally are compensated by using calibration mechanisms in practical applications. The sensors within this work are assumed to be calibrated so this property is solely mentioned for completeness. However, perhaps some new understandings may be gained after a full SHM routine has been implemented since accuracy of calibration specifications is only as accurate as previously specified by the operator.

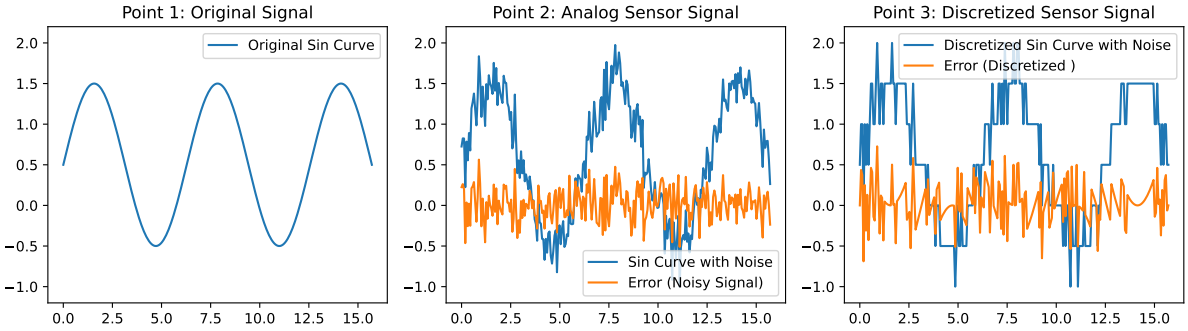
Errors that also occur continuosly are random errors that are however unpredictable in their behaviour. To compensate those errors, filtering can be applied e.g., High- or Lowpassfilters. The final error type is Sensor failure. This is characterised as a spontaneous event that coincides with the complete interruption of the sensor information stream.

Now the question poses if errors can be pinpointed to their origin from where such errors arise. Generally, single sensors carry inherent errors such as clogged static ports (spontaneous hysteresis) or sensor drift (spontaneous bias) but errors also occur within the complex aircraft system architecture itself. An exemplary process step is the Analog to Digital Conversion. Sensor outputs are primarily analog but are then digitized in order to get transported by the aircraft data bus. This process is accomplished using an Analog Digital Converter (ADC), effectively eliminating information from the system. The full process is shown in figure **??**.

Signal Discretization



(a) An exemplary pitot tube setup for measuring dynamic pressure



(b) Original signal transformed into measured and discretized signal

Figure 2.4: Simplified, exemplary signal processing flow for measurement of a dynamic pressure for real (1), analog sensor (2) and discretized sensor (3) values

The undisturbed system state at point 1 gets measured by a sensor and is transformed into the measured signal at point 2 with the error being represented by the orange plot. An ADC builds the next step of the signals journey and after having been digitized at point 3 the signal error grows even further.

Other stochastic error mechanism include but are not limited to sensor drift as well as random noise. In the following, qualifiers and descriptors are introduced to quantify these qualities.

2.1.2 Data Quality

Foundation for this work lies within a common understanding of its results. Standardized qualifiers are found within control theory [ISER11] as well as GNSS applications [TEUN17]. Standardization is also found within

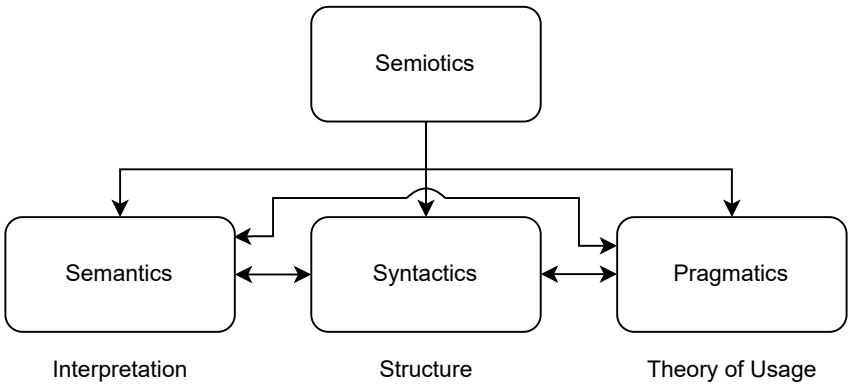


Figure 2.5: Semiology, according to Kutschera [KUTS75] and Shoemaker and Blackburn [SHOE87]

the ISO-Database. However, when beginning to describe data quality within metadata, the issue of metadata quality itself also arises. In the following, a brief overview is given over descriptors and their qualities based on norms.

Semantics

Semantics is a broad term due to its common use. One definition that has found some acceptance is the definition by Metzlers Lexikon which relies on theories by Blackburn and Kutschera. [SHOE87; KUTS75]

Semiotics (σημεῖον,semeion:sign) describes the theory of signs and their usage. Semiotics is divided into the areas semantics, syntactics and pragmatics. Within the definition at hand syntactics is given as the internal structure of signs within sign systems, pragmatics are defined as the theory of sign usage effectively thinking about how interaction with signs works. Finally, semantics define the relationship between signs and described objects. They work by allocating a structure/model to a predefined expressions

This gets developed further within ISO 8000-8: Data Quality [ISO15], concluding that syntactic qualities define the structure of the content language and semantic qualities define the structure of the content itself to allow verifying information. Whereas pragmatic qualities describe the use that can be extracted from the content itself hence allowing validation.

Contextually, the metadata syntax of this work will be in a JSON-format. The used semantics will be standardized to use the SOIL (SensOr Interfacing Language) structure and pragmatics will be quantified by the validity of the actual metadata.

Moving on from these fundamental concepts, the pragmatic data qualities will be defined in the following.

Metadata descriptors

Unambiguous terms are needed to provide clear and concise information within the sensor health monitoring data structure. This is especially important to create a common base of understanding. In the following, industry standards and definitions for data metrics are summarized.

Definitions based on a control theory background are shown in table **??**. These have been agreed upon in discussion with vdi/vde committees as well as the Reliability, Availability and Maintainability (RAM) dictionary. [ISER; DIN77]

These terms are divided into the contextual categories of primarily defining signal and state descriptors. Next up are functions for fault processing followed by model attributes that are employed to compare the actual system to. System properties can then be inferred by leveraging the previous descriptors for states, functions and models. Illustrating the fault detection process, figure **??** shows the method generating a residual which in turn enables interpretation of said residual which may contain information to isolate and identify the error.

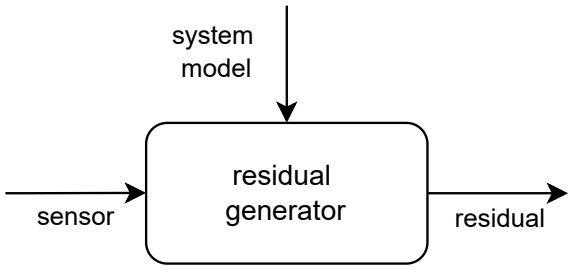


Figure 2.6: Sensor Signal Filter

Further definitions are also employed by the FAA [FAA08] adding integrity as an attribute for GNSS systems that is a quantifier for trust that can be put into the system's measurements. Providing users with warnings when measurements are expected to be unreliable. [FAA08, B.1.5, B.1.10]

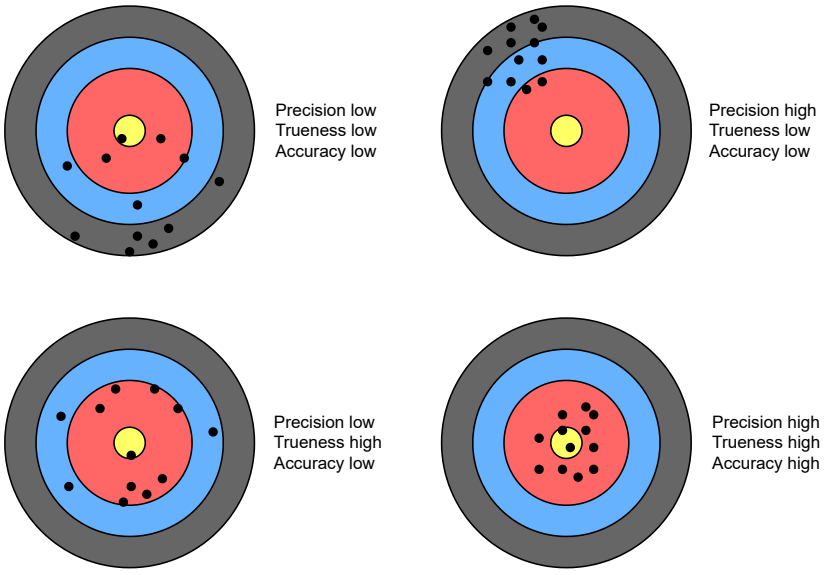


Figure 2.7: Dart boards displaying the Precision (as σ), Trueness (as μ) and Accuracy as their product [ISO97]

Following the GNSS domain, the terms accuracy and precision are important measures for quantifying results. For representation, a dart board is illustrated in figure **??**. It is then important to distinguish that good accuracy is the result of good precision as well as a good trueness. A mathematical description for the precision could be the standard deviation and the deviation of mean and actual value as the trueness. [ISO97][SMIT, S.33ff.]

Descriptor	Description
States and Signals	
fault	unpermitted deviation of one subset of the system
failure	permanent interruption
malfunction	intermittent regularity
disturbance	unknown, uncontrolled input
perturbation	input, leading to temporary departure from steady state
error	deviation between measurement and true value
	$y_e = \bar{y} - y$
residual	deviation between measurements and model-based calculations
	$\hat{y} = \bar{y} - y_m$
Functions	
fault detection	determination fault presence
fault isolation	Determination fault properties: kind, location, time of detection
fault identification	determination of size and time-variant behaviour of fault
fault diagnosis	includes fault detection, isolation, identification
monitoring	real-time determination of possible physical conditions and
	recognition, indication of behavioural anomalies
supervision	monitoring and taking actions to maintain operation during faults
protection	means by which potentially dangerous behaviours are suppressed if
	possible or how consequences are avoided
models	
quantitative	describe system in quantitative mathematical terms
qualitative	describe system in causalities and if-then rules
diagnostic	link specific inputs (symptoms) to outputs (faults)
analytical redundancy	determine a quantity in an additional way by using a mathematical process model
system properties	
reliability	ability to perform a function, measure $MTTF$, with λ as rate of failure per hour
safety	ability of a system not to cause danger to persons, equipment and environment
availability	$A = \frac{MTTF}{MTTF+MTTR}$
λ	rate of failure
μ	rate of repair
MTTF=$1/\lambda$	Mean time to failure
MTTR = $1/\mu$	mean time to repair

Table 2.2: Terminology for system properties [ISER11]

2.2 SHM Algorithms

Building now upon a solid foundation of descriptors, some previously implemented systems dealing with Monitoring and Supervision tasks are reviewed. The superset of these monitoring systems is generally described as Failure Mode and Effect Analysis (FMEA). In the following, some FMEA implementations are reviewed. Theoretic fundamentals for this works' actual implementation are then shown, checking sensor values against limits and then correlating sensor values with each other.

2.2.1 Introduction to FMEA

Motivated by the desire to warn operators of system failure the FMEA has a relatively simple goal. To detect failures and faults and alert the user to avoid further damage/costs/downtime. This naturally establishes it as one of the pillars of automation. [ISER06] Since automated systems are only as smart as their creator, methods have to be put into place assuring their high level of quality, guaranteeing smooth operation which in turn spawns and motivates the entire field of FMEA. In recent years, especially systems based on machine learning approaches have come into play due to widely available cheap computing power as well as increasingly optimized training algorithms. Due to the scope of this work that is primarily trying to establish a pipeline for an automated FMEA on cloud data this however remains to be addressed at a later point.

In the following, state of the art non machine-learning approaches are introduced preceeded by a brief theoretic section on statistics for completeness. The FMEA methods then outlined feature the principal component analysis (PCA) used in live monitoring [XIAO06], Pseudo Transfer Functions (PTFs) in control theory [ALJA15], parity equations as well as grey box models using parameter tuning. [ISER06] In addition, some methods from the GNSS field are presented since much of GNSS research is public in contrast to most control algorithms for aircraft systems.

Statistical methods

[SMIT]

Definition Fault Detection + Fault Diagnosis (Fault-Diagnosis Applications)

Signal properties are examined for a given signal in figure **??**

Sensors generally produce errors within expected forms of output. -Noise -Measure using Covariance –> autocovariance -Offset -No response

Statistical representation of: Mean

[SMIT, S.13-17]

Time continuous mean

$$\mu = \frac{1}{T} \cdot \int_{T} x(t)dt \tag{2.1}$$

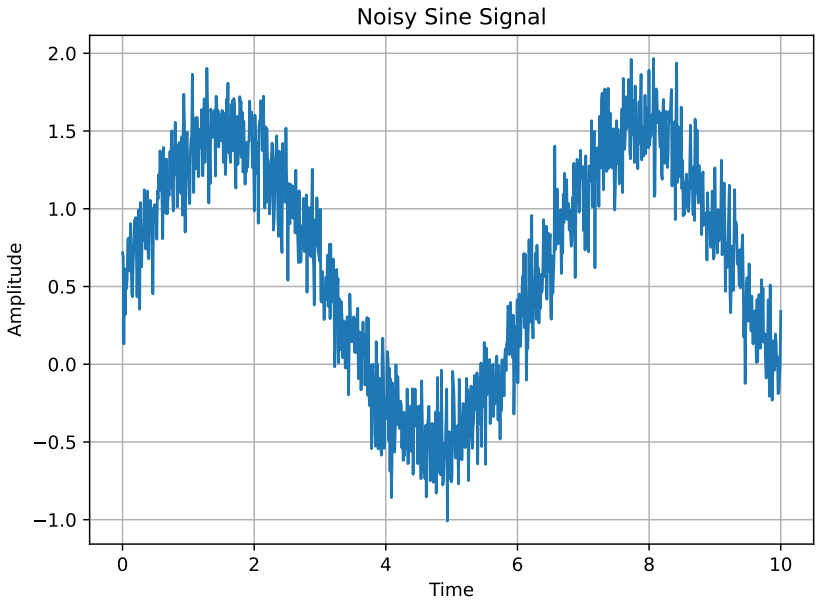


Figure 2.8: Noisy Sine Signal

Time discrete mean:

$$\mu = \frac{1}{N} \cdot \sum_{i=1}^{N} x_i \tag{2.2}$$

The variance σ^2 is a metric for the signal's behaviour. It expresses the mean squared deviation from the mean.

Time

$$\sigma^2 = \frac{1}{T} \cdot \int_T [x(t) - \mu]^2 dt \tag{2.3}$$

$$\sigma^2 = \frac{1}{N} \cdot \sum_{i=0}^{N} [x_i - \mu]^2 \tag{2.4}$$

The standard deviation is derived from the variance. Its value gets square-rooted to better represent the power (magnitude of the amplitude).

Standard Deviation

$$\sigma = \sqrt{\sigma^2} \tag{2.5}$$

Mean and the standard deviation don't represent the desired metrics in some use cases. Rather more important is a comparison between the two. Hence, the Signal-to-Noise ratio (SNR) is used to compare and condense the mean and standard deviation by dividing the mean by the standard deviation.

$$SNR = \frac{\mu}{\sigma} \tag{2.6}$$

Another parameter is the coefficient of variation (CV) which is the standard deviation divided by the mean and multiplied by 100%.

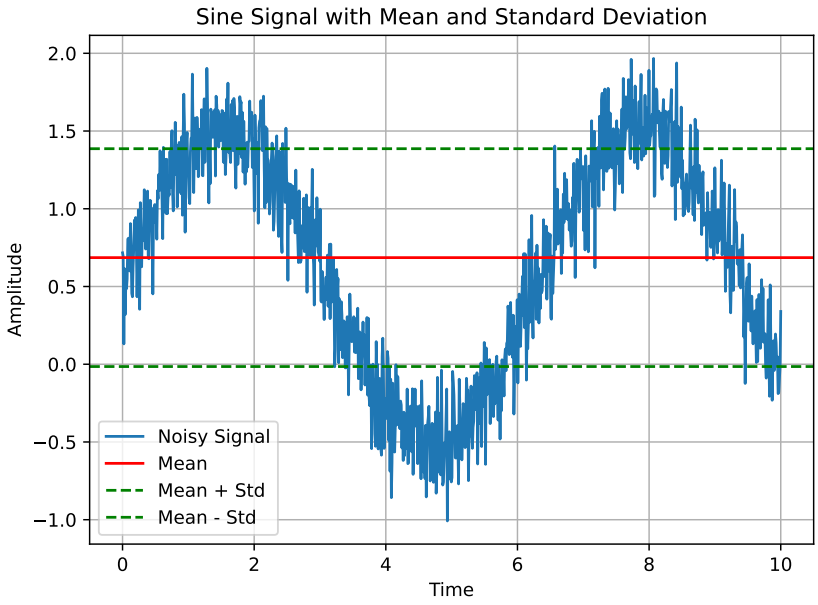


Figure 2.9: Sine Signal with mean and standard deviation

$$CV = \frac{\sigma}{\mu} 100\% \tag{2.7}$$

An arising problem based on the SNR and CV are however that they scale based on the mean value. Should the mean value lie at about 0 for e.g. a sensor of an aircraft control surface, the signal to noise ratio will be relatively high compared to an acceleration sensor in z axis with a constant offset of 1g

Practical example for mean and standard deviation in a given signal are overlayed in figure **??**

To evaluate a signal according to the quantities the next logical step for statistic Histogram

probability mass function

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.8}$$

Covariance time-continuous

$$\gamma(\tau) = \frac{1}{T} \cdot \int_T [x(t) - \mu] \cdot [x(t-\tau) - \mu] dt$$

Covariance time-discrete

$$\gamma(k) = \frac{1}{N} \cdot \sum_N [x_i - \mu] \cdot [x_{i-k} - \mu]$$

It follows then that the autocovariance for a variable is

$$autocov(x) = cov(x, x)$$

The Pearson coefficient as the correlation matrix is hence reached by scaling the covariance matrix by the standard deviation: [SMIT]

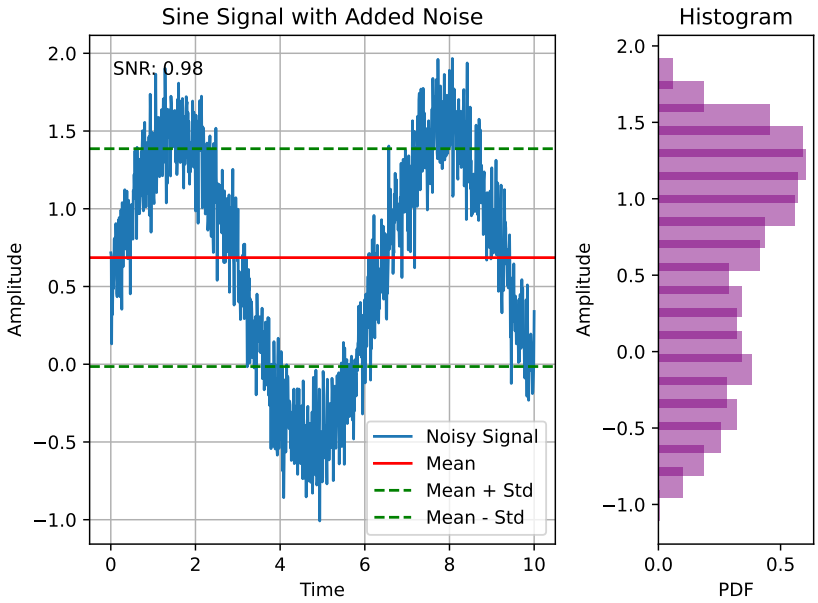


Figure 2.10: Sine Signal full analysis with mean, stdev, SNR and Histogram

$$\rho_{x,y} = corr(x,y) = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

With the limits of correlation being -1 for total antiproportional correlation and 1 for total correlation. A total independence for $\rho_{x,y} = 0$ not given since Pearson coefficient only detects linear correlations and not nonlinear behavior. Other correlation methods as rank-correlation (Spearman, Kendall) that detect change correlations are possible but are more complex in the implementation.

2.2.2 Check 1: Existence

The FMEA procedure is divided into three steps, separating the involvement of each algorithm. Metaphorically speaking shaping the inputs and outputs of the algorithm blackbox. Primarily, it is checked whether parameters exist in the database. This is checked by comparing the DAQ-configuration to the actually measured data. Within step two, single sensor signals are examined by themselves, saving ressources by iteratively working on each sensor. This approach also allows easy parallelization of tasks due to the encapsulation of each task. Step three allows the integration and correlation of different sensor signals. Allowing more complex FMEA systems.

Within the first step, the main challenge lies within the acquisition of sound configuration data. Errors may occur as early as the configuration step due to human error. This steps allows to get an overview over these first rough mistakes.

2.2.3 Check 2: Plausibility

Within the plausibility check the single parameter is checked against limits that shall be preconfigured for each parameter. Various methods are examined. The primary focus is checking the value against limits to detect extreme values that are unrealistic.

Noise algorithm:	Moving Average	FFT	STFT	WT
Implementation Effort	1	3	2	4
Result Quality	1	2	3	4
Real-Time Performance	4	2	3	1

Table 2.3: Comparison of Level 2 Noise approximation algorithms

Within the next step, the series shall be transformed with the noise or other attributes being extracted. These attributes can then in turn be checked against other limits.

For the noise, we examine various known methods to create a heuristic for noise approximation. Of course, noise approximation models range in complexity and detail from a simple moving window approximation into Fourier transforms (FFT, Fast-Fourier-Transform; STFT, Short-Time-Fourier-Transform), Wavelet Transform (WT) and into advanced works such as language-processing models [HEND08] which exceed the scope of this work and may be implemented at a later date.

These functions are evaluated into a ranking and displayed in Table **??**. After brief evaluation, the Short-Time-Fourier Transform is considered due to its comparability among sensor signals. Allowing to compare movement activity by amplitude.

The STFT algorithm is then implemented preliminary by using a short time window of 256 samples and then collapsing the resulting spectrogram by time. This results in an averaged amplitude value for each timestep that is detrended. Leading to a scalar value, upon which the noise of each value can be mapped.

2.2.4 Check 3: Parameter Correlation

Physically correlating and involving parameters into the anomaly detection of other parameter generally is the next logical step. It is primarily considered to use conventional non ML-approach using known correlations of parameters and perhaps allowing some greybox approaches for fine-tune limits and finding unknown correlation. A primary desire is to generate a model that is not hardcoded but easily expandable to satisfy various data-generators.

enter placeholder for image: Measurement equals signal + error and image: luenberger beobachter. Kalman Implementierung [LIE13] Parameter Correlation Studies [LI15]

Some traditional non-ML FMEA approaches are presented in the following paragraphs.

PCA

The Principal Component Analysis was first introduced by Karl Pearson in 1901 ([PEAR01]) and has since become a popular method to analyse datasets and detect correlations. The PCA works by reducing the dataset dimensions using covariance of parameters relative to each other. This results in a method that needs minimal user input but delivers a condensed data overview. The only input needed is the amount of dimensions into

which data shall be condensed into, the so called Principal Components (PCs).

The functionality, applications as well as usability within the context of this work are discussed below.

Functionality

The principal component analysis (PCA) is a method that allows to reduce the information parameters of a dataset into a few principal components or dimensions. Generally this is specified as the operation of multiplying the original dataset X_{Nxm} with the timesteps N and the measurement parameters m with the Permutation Matrix P_{mxr}. This results in the transformed principal components (PC) T_{Nxr} with r being amount of reduced dimensions or also the number of Principal Components. This transformation is defined as:

$$T_{Nxr} = X_{Nxm} P_{mxr}$$

P being the permutation matrix that transform the original data into the principal components. The overall strategy to generate a PC is shown in figure **??** as finding a function between two parameters x_1 and x_2 that maximizes the variance of data in a newly generated principal component. However drawbacks arise using this approach since the variance optimization only happens linearly and is not able to function for nonlinear correlations similar to a standard Pearson-Coefficient Correlation matrix. [HAND17]

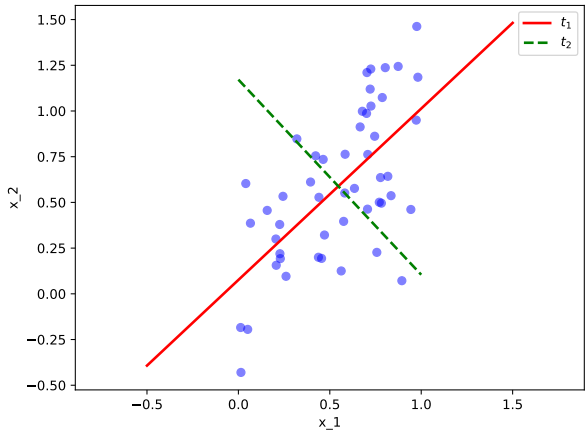


Figure 2.11: Parameters x_1 and x_2 get converted into principal components t_1 and t_2 by optimizing for minimal variance.

PCs can be created in any direction. Where input is needed however is the number of Principal components that shall be kept. A cutoff value can be defaulted to but this may not be feasible for multidimensional systems such as the flight test data that is at hand.

Eliminating a principal component becomes easy if parameters show a direct correlation such as in figure **??**. The gatherings of t_2 then mostly consist of noise and can be safely discarded.

Utilization

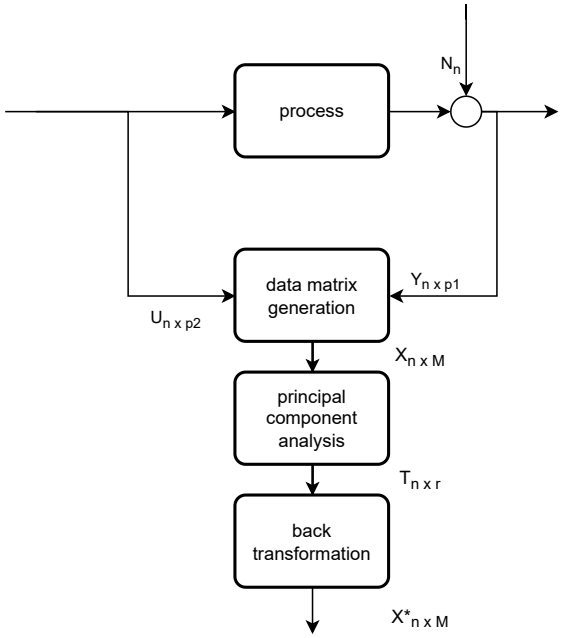


Figure 2.12: Known process input U as well as Sensor Measurement Y get collected into X and fed into a PCA. N indexes timesteps.[ISER06, p.268]

This analysis method is especially helpful for datasets for which correlations are unknown such as biomedicine in which marker correlation needs to be identified. But even for aerospace contexts it may prove helpful for large quantities of data in which correlations are unknown e.g. for parameter identification as well as perhaps aeroelastic applications in which vast amounts of acceleration sensors and strain gauges are present.

In these applications the PCA is a great tool to get a first overview over the data. For more refined analysis however it lacks detail since it generally uses linear optimization as and may be prone to overfitting for redundant input parameters. In these cases, it may be sensible to use optimized PCA approaches or an entirely different analysis system such as ML-approaches.

The PCA can be defined as a condensation of a dataset into fewer parameters. The so called principal components. These principal components can also be transformed back into the original parameters. This process can also be used to create a data model and then calculate the residual to possibly quantify the error. E.g. Isermann [ISER06] presents a process for an autotuning PCA FMEA that can be used for real-time Fault Detection applications.

Assessment

The principal component analysis is based on correlation principles and allows reduction of datasets into independent components. Applications in an aerospace context include detection of previously unknown correlations between parameters and complex interactions in aeroelastics or otherwise large datasets. It also may allow generation of rules to detect a deviation from previously correlated values. Usage however is associated with a certain effort since information needs to be preprocessed to a certain degree, reducing dimensions previous to processing in order to avoid feeding in redundant parameters. Upsides of the PCA are that it acts as a blackbox that quickly allows to reduce data onto a given set of dimensions, requiring relatively little user input.

PTF

Following the recipe for a modeled aircraft based on sensor data we try to simulate the parameters x and u of the aircraft with the sensor data y. Khaled shows that this approach works for linking the angular velocities for

the aircraft axes $omega_x$, $omega_y$, $delta_delta(driftangle)$ with $omega_z$. This is accomplished by tuning the Transmissibility function $\mathcal{T}$ using dynamically generated Markov-parameters. Essentially generating a type of correlation matrix for the parameters that gets tuned within the beginning of the measuring period that is then able to generate a residual, detecting errors. This approach is explained more closely in the most recent publication, examining the unknown dynamic systems of a roomba platoon. [KHAL22a; KHAL22b]

Functionality

The examined approaches include:

Transmissibility functions $\mathcal{T}$ that model the system output without having to take the unknown system input into consideration.

+TODO?: include more detailed examples

Utilization

Since this is a relatively new development, no broad implementations exist. It is to note however, that this method takes inspiration from structural health monitoring by transforming the structural health monitoring algorithms that work in the frequency domain into the time domain.

Assessment

This approach seems especially suitable for real-time implementations and may also be used in postprocessing. It comes however with the penalty of not being usable straight out of the box since no python implementations exist so far. For future work, this may be implemented to evaluate its capability against other algorithms. It also remains to be seen how well this algorithm performs on large datasets and how much preprocessing effort needs to be undertaken.

Parity Equations

Parity Equations work by modelling the physical state of a system in normal working conditions. They are based on knwon physical relationships between sensor measurements and generate a residual. This residual is then continuously checked during operations and once it reaches or exceeds a previously defined threshold it notifies the system's operator. [ISER11]

Functionality

The Parity workflow separates into the two steps of calculating the physical state and generating a residual and then interpreting the residual.

Noise algorithm:	PCA	PTF	Grey Box Models	Parity Equations
Implementation Effort	3	4	3	1
System Knowledge	0	2	2	4
Reliability	2	3	2	4

Table 2.4: Comparing FMEA solutions by their respective usability

Utilization

Parity Equations are employed in systems in which the physical correlations are well-known. They allow fusion of various parameters into system state variables to then generate residuals for single parameters to determine the parameter errors.

Assessment

Since the relations of the aircraft systems and behavior have historically been subject of extensive research the Parity equations lend themselves to be considered within this context. Extensive tuning is however integral part of Parity Equations since physical correlations as well as residual thresholds need to be tuned to avoid amassing false positives.

Grey Box models

Another method for system modelling lies in grey box models. The "grey" attribute arises from the property that these systems have baseline constraints that may be set and some degrees of freedom such as tuning parameters. These tuning parameters are then tuned against a set of previously defined training cases to satisfy the constraints. This represents an approach that works with limited system knowledge and implementations such as the PTF work to this model.

FMEA Evaluation

An algorithm for evaluating data quality in level 3 must now be found based on these state of the art approaches.

Possible issues arise while comparing reliability. Especially in aerospace with multiple independent axes a false correlation may be detected due to circumstances. E.g. for a flight performing multiple consecutive landings it is possible for the Distance Measuring Equipment (DME) measuring distance to the airport to correlate with altitude and speed since all decrease and increase simultaneously. Simultaneously, employing no/low knowledge models enables rapid results eliminating the need for any degree of system knowledge.

In table **??** the introduced FMEA-schemas are evaluated on their viability to use within this work. While PTF as well as other Grey-Box models exist, they need considerable effort within their primary implementation and then their respective training. Since the research and development of a novel, optimal FMEA is also not focal

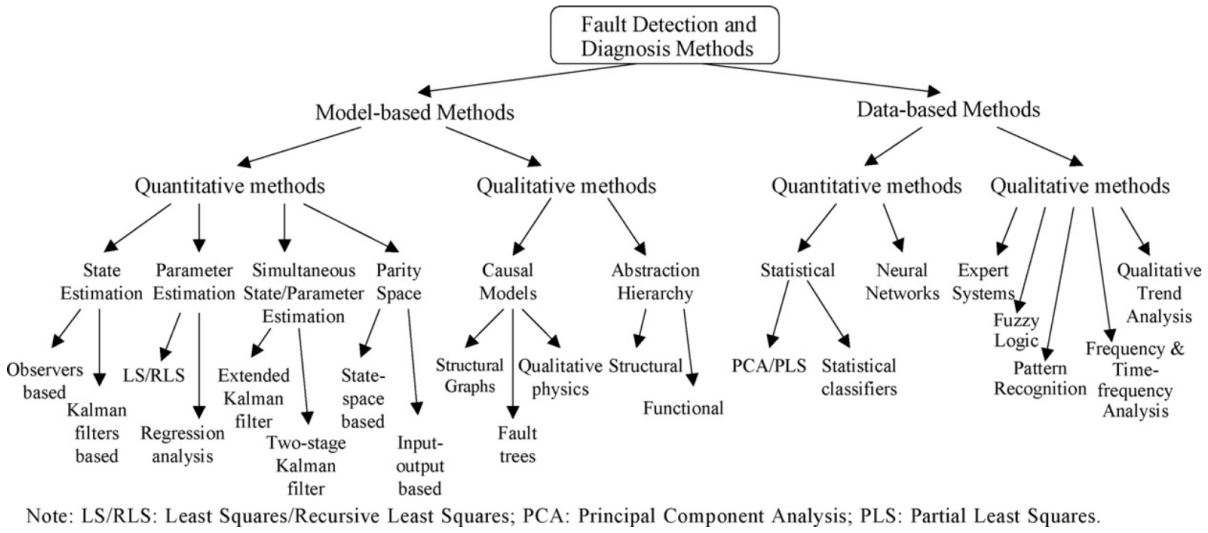


Figure 2.13: An overview over various kinds of FMEA [ZHAN08]

point of this work since it is rather more important to generate a holistic heuristic, solving first and foremost the basic problems associated with basic errors. Hence, Parity equations are chosen to be developed further within this work. Generating a dynamic model of the aircraft physical correlations that allows itself to be expanded for future use.

It also needs to be taken into account that this summary of FMEA methods barely scratches the surface. Far more FMEA variations exist as shown in figure **??**. It remains to be seen now, how well this work's architecture for integrating these FMEA may hold up in regard of testing these methods in future work.

Barometric Altitude

For the Parity model, the barometric altitude is introduced to compare experimental pressure sensors with the aircrafts barometric altitude.

The International Standard Atmosphere (ISA) is the conventional method of measuring an aircrafts altitude based on air pressure. [ISO75] The method can be understood as a function of pressure and reference pressure (differing based on weather) outputting an altitude. The assumptions for each layer of the atmosphere are shown in table **??**.

Based on these constants the general equation for pressure within an atmospheric layer with a nonzero temperature gradient is shown in equation **??**. [ISO75]

$$\frac{p}{p_0} = \left(1 + \frac{a \cdot (h - h_0)}{T\,(h_0)}\right)^{\frac{-g}{a \cdot R}} \tag{2.9}$$

Atmospheric Layer	Geopotential Altitude [km]	Temperature T at bottom [K]	Temperature gradient a [K/km]
Troposphere	-2-0	301.15	-6.5
Troposphere	0-11	288.15	-6.5
Tropopause	11-20	216.65	0
Stratosphere	20-32	216.65	1
Stratosphere	32-47	228.65	2.8
Stratopause	47-51	270.65	0
Mesosphere	51-71	270.65	-2.8
Mesosphere	71-80	214.65	-2

Table 2.5: International Standard Atmosphere [ISO75]

p = pressure at current altitude [Pa]
p_0 = reference pressure [Pa]
h = current altitude [m]
h_0 = reference altitude [m]
$T(h_0)$ = temperature at reference altitude[K]
a = ISA Temperature Coefficient depending on altitude $(-6.5)[K/km]$
g = gravity constant $(9.80665)[m/s^2]$
R = Ideal Gas Constant $(287.05287)[m^2/(K \cdot s^2)]$

Transforming this formula for altitude resolves into equation **??**

$$\rightarrow h = \left(\frac{p}{p_0}^{\frac{-a \cdot R}{g}} - 1 \right) \frac{T(h_0)}{a} + h_0 \tag{2.10}$$

Constant layer temperature

Resolving for pressure into equation **??** follows into the barometric equation for a constant temperature coefficient.

$$\frac{p}{p_0} = \exp\left(\frac{-g}{R \cdot T(h_0)} \cdot (h - h_o) \right) \tag{2.11}$$

The following equation **??** then resolves for the altitude based on pressure ratio within a layer with constant temperature.

$$h = ln(\frac{p}{p_0}) \cdot \frac{R \cdot T(h_0)}{-g} + h_0 \tag{2.12}$$

Logic Flow

The process to then actually calculate a barometric altitude is displayed in figure **??**. The input is p and p_0 with p being the measured pressure via the aircrafts static ports and p_0 being the reference pressure that is manually set by the pilot with the standard pressure being $p_{std} = 1013.25$hPa which varies depending to the weather.

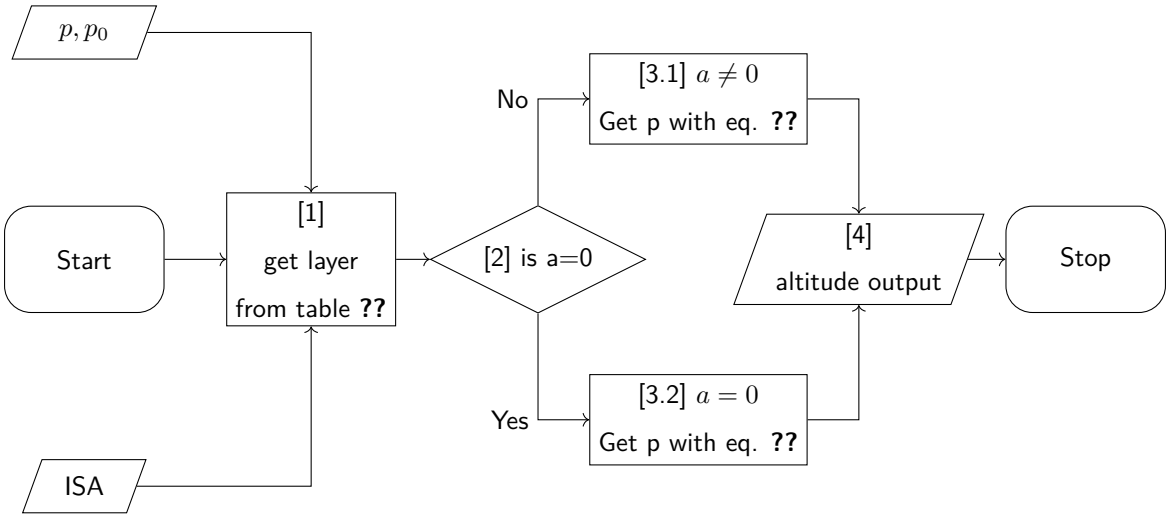


Figure 2.14: Process for determining barometric altitude from pressure and reference pressure only.

In step 1 the current atmospheric layer gets found by matching the MSL-pressure ratio $\frac{p}{p_0}$ to precalculated boundary values of layers based on table **??**. After determining the layer and the value for temperature coefficient a, the altitude can quickly be determined by inserting layer properties as well as the pressure ratio into their respective altitude equation (step 3.1 and 3.2 in flowchart **??**). In the final step 4 the altitude is returned.

GNSS

In addition to the barometric altitude, the Global Navigation Satellite System (GNSS) is also measured within the ISTAR.

For reference, the aircraft altitude generally is defined as the displacement of the aircraft from Mean Sea Level (MSL). It is important to note that the earth can generally be described as an ellipsoid due to its rotational shape in a geodetic context resulting in a very close approximation of the earth's shape. However, varying density levels of the earth's crust cause the elevation and sea level to deviate from the ellipsoid shape. This results in a lopsided model that is modeled in the WGS84 ([SLAT98]) system. This is also the altitude that the gps measures. And will be the reference altitude for the following calculations.

The main existing altitudes are the:

1. Geodetic Altitude (GNSS)

2. Barometric Altitude (used in conjunction with reference pressure)

3. Inertial Altitude

4. Radar Altitude (can be used in conjunction with a terrain model to derive geodetic altitude)

Possible errors for each are: geodetic: inconvenient satellite placements, deflection of signals in the atmosphere and signal problems

Barometric Altitude: Possible errors due to drift and meteorological atmospherical pressure shifts, Reference Altitude.

Going into WGS84 and the GNSS Altitude however exceeds the scope of this work. In the following, the satellite altitude above Mean Sea Level (MSL) is considered as the reference altitude.

Fault detection algorithms within GNSS are described as Receiver Autonomous Integrity Monitoring (RAIM) are tried and tested within GNSS implementations since high accuracy positioning is valuable for various applications, reaching precisions of up to a few centimeters. Explaining the full function of position calculation exceeds this works' scope. To summarize however, GNSS inputs form an overdefined system of equations which needs to be compensated within some algorithm to form one position based on multiple inputs.

Summary

The fundamentals of potential algorithms to quantify system qualities have been presented in the past section. the Statistical fundamentals have been briefly outlined followed by FMEA implementations from statistics in the shape of Principal Component Analysis (PCA) ranging to implementations from control theory. Advanced models such as machine learning models have not been presented but are generally similar to PCA since they also work without system knowledge and also extract features similar to Principal Components.

The research within the general quality control field is generally very optimized with most of its research on analytical and traditional methods having been developed and implemented in the late 20th century [ISER11]. The new arising challenge faced within this work lies primarily in integrating these existing algorithms to allow evaluation of them. In the future such toolboxing approaches may allow quick and easy comparison of new experimental approaches by e.g. running new experimental ML-routines in such a toolbox to then validate them against existing tried and tested FMEA approaches within frameworks such as the ones developed within this work.

In addition to detecting faults it is also necessary for a SHM to display faults to the systems operator. This may happen in the shape of a display interface or a dashboard that generates fault and reliability ratings. Condensing information for operators is also a necessary part of SHM since suspicious occurences may be frequent and raw information about events is less helpful than already preprocessed data. This is what is now presented in the upcoming sections, first dealing with the structure in which the metadata will be handled and discussing the origin and destination of data in the SHM context.

2.3 SHM Metadata Structures

Data handling as well as structuring the data is essential for fulfilling FAIR criteria in terms of the accessibility and reusability. Thus necessitating a standardized data and metadata format. In the following, the FAIR standard and the motivation of data handling is presented and then its application in form of data format as well as the metadata format are introduced.

2.3.1 FAIR

Standardized data handling and following of conventions in regard of data structures always directly clashes with the desire to optimize data structures, making them more efficient, dynamic as well as developing them further. Were data structures once used as purely SQL (tabular format) this has now expanded into a wide array of data structures that are tightly knit into the needs and requirements of their application. Starting from tree structures, going into graph-theory and furthering into unstructured paradigms, these approaches represent the tradeoff between standardization and accurate data representation. The following paragraph discusses the use of FAIR principles to satisfy the standardization aspect while also allowing flexibility within the system to represent the underlying physical systems.

FAIR principles

Originating from a dutch alliance of biotechnology institutes in 2015, the FAIR Guiding principles emerged in 2016 as a general best practice guide for research data, referencing best practices

FAIR principles have been published in 2016 by Wilkinson et al. [WILK16]. They set the foundation for a standardized and open data culture. Within these principles lie values like open access for data, findable and well tagged datasets, interoperable data by using standardized formats and semantics which guarantee a reusability of data to generate a sustainable process for data usage. Effectively meaning that similar experiments do not have to be performed multiple times when well tagged and formatted data is freely available.

The full FAIR principles are shown in figure **??**.[WILK16]

FAIR essentially acknowledges that data analytics have come so far that the data analysis has become very effortless. So effortless however, that a major part of data science consists of formatting and acquiring data. To circumvent this issue in the future and also learn from past data it is then vital to guarantee FAIR principles. This allows to leverage current technology to automate analysis and enable detection of previously undetected correlations within datasets.

SI-Units

Units, especially within aerospace contexts, are far from standardized. Most scientific users prefer SI-units. However, the convention for Pilots and Flight Test Engineers remains imperial within units such as feet, Nautical Miles, knots and so on. Naturally SI-units are chosen for calculations since calculations are more efficient as

Findable	F1. (Meta)data are assigned a globally unique and persistent identifier
	F2. Data are described with rich metadata (defined by R1 below)
	F3. Metadata clearly and explicitly include the identifier of the data they describe
	F4. (Meta)data are registered or indexed in a searchable resource
Accessible	A1. (Meta)data are retrievable by their identifier using a standardised communications protocol
	A1.1 The protocol is open, free, and universally implementable
	A1.2 The protocol allows for an authentication and authorisation procedure, where necessary
	A2. Metadata are accessible, even when the data are no longer available
Interoperable	I1. (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
	I2. (Meta)data use vocabularies that follow FAIR principles
	I3. (Meta)data include qualified references to other (meta)data
Reusable	R1. (Meta)data are richly described with a plurality of accurate and relevant attributes
	R1.1 (Meta)data are released with a clear and accessible data usage license
	R1.2 (Meta)data are associated with detailed provenance
	R1.3 (Meta)data meet domain-relevant community standards

Figure 2.15: The FAIR principles [WILK16]

well as less prone to errors in SI-systems. Within this work, base SI-units will be used (e.g. m, s, N, kg, Pa).

2.3.2 Methods

Within this work it is then necessary to use standardized fair frameworks for data and metadata exchange. Since the skystash, a FAIR research data platform is already in development and in prototypical use within the DLR, the developed algorithm will be tested by working in conjunction with the skystash. To exchange data, the SOIL (SensOr Interfacing Language) framework will be examined for defining metadata of the aircraft and the FMEA logic itself.

Skystash

The skystash is a platform to distribute, analyze and visualize large flight data sets. It is currently in development within the DLR's digital twin project and aims to be a service platform for uploading and sharing the DLR's flight test data. Various requirements are posed from different stakeholders. And large data sets necessitate a platform that can handle these file sizes.

What are skystash goals to implement, how does it generally work? Mongodb datastructure. Fixed structure of Project/Collection/Series for all flights. Enrichment with json-metadata as well as binary objects is possible. MongoDB is accessible via api and web client to mitigate user errors within the database. Data series are saved as numpy errors. Nginx controls web traffic and accepts input via python API and the website and redirects it to ceycloak for authentification, angular frontend for displaying, and stashclient for accessing the database for performing data mutation or search operations using ist powerful search query.

Flight test data needs to be perceived within the context of its circumstances to extract all its information. Hence the need for a digital twin. Within this context all necessary data shall be compounded to set a central starting point for data exploration and analysis. The DigECAT project attempts to solve this issue by providing the skystash platform. Figure **??** shows all data sources that will be gathered. For this works context this

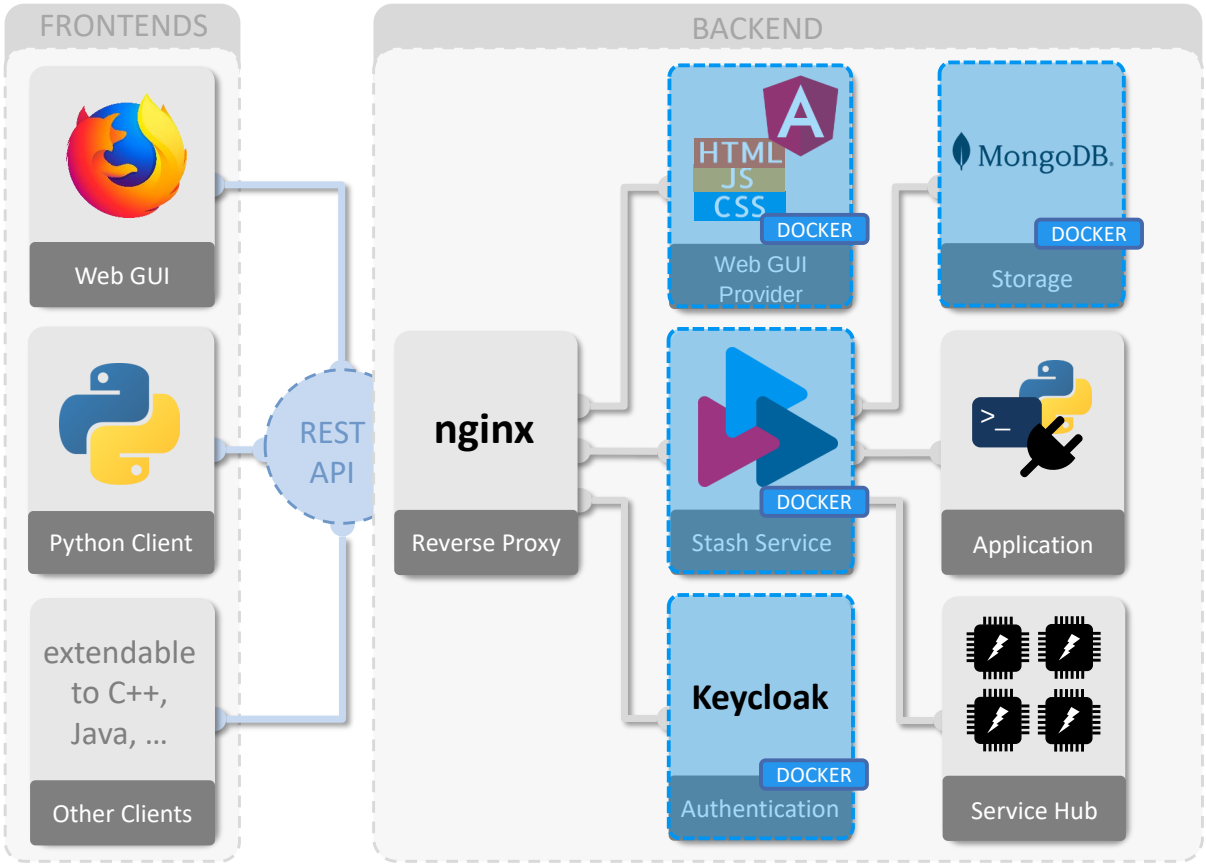


Figure 2.16: The Skystash architecture [ARTS]

means that necessary aircraft configuration data such as location of sensors is available (metadata) next to the actual generated data from the aircraft systems (data). This for the first time allows contextualizing flight data in one place. When previously various metadata and configuration info had to be manually collected, this whole process aims to minimize manual steps within the whole process.

[ARTS]

This work builds upon the python api and tries to implement and test an algorithm that checks the data for the istar aircraft [MEYE20]

Metadata exchange format

Aiming for a standardized machine-readable data format the SensOr Interfacing Language (SOIL) is examined. The aircraft data as shown in figure **??** is varied in format and shape. For this work however we need a data format that is able to handle and represent the intricate details of the aircraft precisely. For the various sources, a metadata enrichment step might be practicable. Effectively, this would act as a metadata importer that fuses metadata and also translates it into SOIL.

Use of FAIR in the implementation

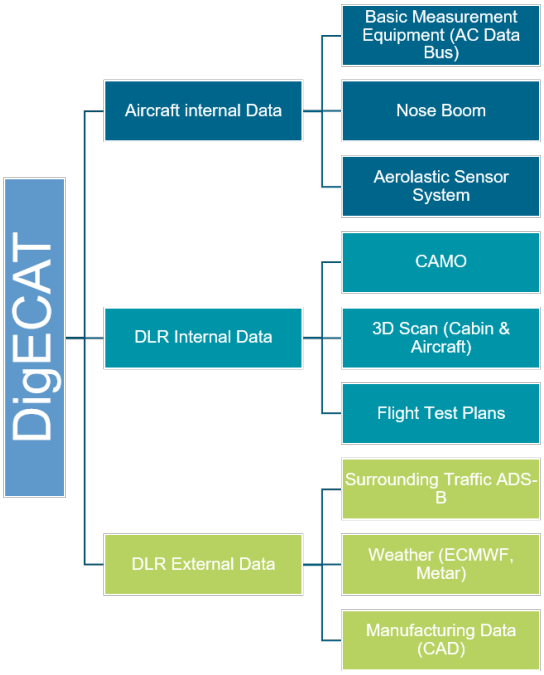


Figure 2.17: Generation of flight test data [ARTS]

To now fulfill FAIR principles, we examine the interfaces in the architecture. The generated flight test data is available and metadata is already in a JSON-file format. To further specify and standardize metadata we can translate metadata into SOIL to further Interoperability. This enriched metadata then can be fed into the FMEA step. Exiting the FMEA-step, the Analysis results can also be defined within a SOIL/JSON format and then get opened by other applications to access data quality results and perform further analysis steps, increasing Reusability.

By implementing standardized JSON-data structures for the metadata including dataset and report, interoperability as well as reusability are guaranteed. To increase Findability, the skystash architecture is used which generates a unique identifier for every dataset. [MEYE20]. By enriching metadata, findability as well as accessibility is guaranteed. And finally, converting to SI-units guarantees accessible, interoperable and reusable data.

2.4 SHM results config

Within the last step of the FMEA, the generated report results need to be analysed which in turn might necessitate change of the FMEA parameters. To facilitate the process and lower the barrier of entry, the possibility of an interactive report/dashboard is examined with the ultimate goal to output a report that is clear, concise and standardized.

2.4.1 Challenges

The requirements for such a report are firstly motivated by necessity. It is vital to show error detection, embed the report into the toolchain and find a format to communicate metadata as well as SHM findings. Such a report tool could then also be used in the development of new algorithms since it generates quick feedback and precise feedback, aiming to display various Indicators for data quality and FMEA quality.

2.5 Summary

3 Problem Statement

A roadmap is important for any larger undertaking. The following outlines the important stops along the route of the upcoming work.

Primarily, it shall be said that a large problem with large-scale measuring systems is the sheer scale and inefficiency of manually checking for errors. Since errors in aviation contexts mostly have fault rates of $10^-6/h$ it allows for few errors even with large datasets of up to 5000 sensors. However, many sensors in this application are built in experimentally and can be assumed to be at a fault rate of $1/h$.

This work then shall start with the skystash architecture in mind. Firstly, it needs to be assured that the available data is sensibly imported into the skystash. A second hurdle lies within the metadata treatment. Metadata needs to be parsed from the proprietary .imcexp format. Steps undertaken will then involve uploading the available data to the skystash without modification, forming the first step of data provision.

Metadata enrichment follows as the next step, enriching the DAQ metadata with additional sensor information, SHM information such as boundaries for level 2 as well as level 3 tags for mapping parameters onto categories. Within this step all this data is then taken and transformed into a standardized format (like the SOIL format) that enables a standardized interface to the data processing step.

Data processing builds the next step. The previously standardized, enriched metadata is used to check levels 1, 2 and 3. It is

The work allows itself to then be

- prototype for a complex sensor health monitoring structure. -employing standardized data classes -metadata semantics (SOIL) -pandas series -employing standardized interfaces for modular expansions

-structuring

▪ Precise detection of malfunctions and perturbances ▪ Reporting tool that displays relevant info and generates metrics Plug and play philosophy

3.1 Concise Problem Proposal

existing proposal:

Sensor Health Monitoring

Sensors fail. And have unexpected outputs. Not catching these faults can sometimes come at great costs when a research campaign proves to be useless when no sensible data has been produced. And even greater costs when the data acquisitioning systems have been changed and reconfigured for the next campaign. To solve this, an automatical fault detection/Health Monitoring of sensors shall be investigated and developed.

Motivation and problem

To create checks, the sensors have to be checked against a configuration database. As straightforward as it sounds this proves to have some challenges to it as the existing configuration formats need to first be understood and then transformed into a format in which they can be read by the python code. Of course, this step needs to be dynamic and expandable for different configurations since the aircrafts' configuration may change frequently. A nomenclature and procedure can be formulated upon segmenting the problem of faulty sensors into different fault catching steps. 1. Production of data 2. Production of data that is within an expected range of values 3. Production of data that acts sensible in reference to other sensors Since metadata regarding Steps one and two like the Sensor names as well as their ranges can be defined in the configuration database, the main work to solve these procedures lies in the creation of a thorough configuration management. Systems which simulate a dynamic model as well as neural networks or other algorithms can be implemented in step 3 for finding correlations between sensors and detecting perhaps previously unknown correlations.

Approaches

Sensor Health Monitoring is a multilayered problem whose challenges not only lie in the single operational tasks themselves but also the integration into the existing infrastructure of the sensors. Since the DLRs aircraft sensor configurations are subject to change a database shall be devised that does not create an additional location for configurations to be updated. Thus, a dynamical sensor health monitoring over changing sensor configurations is proposed that at least shall be designed with the architecture in place to enable such a design Changing configurations may be incorporated into a standardized sensor configuration management system. Research has to be conducted on how this database of sensor metadata is going to be implemented in the best way. The SOIL system shall be examined closer for this task and rated and perhaps modified according to its' abilities. Once a sound sensor configuration database is found, simple checks on SHM Level 1 and 2 should be conducted without great effort. For Level 3 approaches, a systemic approach will have to be found that interfaces the sensors to a physical correlation model and then back to themselves. Simple physical correlations can be a starting point for more complex algorithms like control system approaches based upon Pseudo Transfer Functions. These would be able to reference the State Space variables to the sensor outputs without knowing the systems inputs. Also possible is the use of neural networks. However, the control systems approach is chosen first as it offers more transparency on its logic and hence facilitates the debugging process. Once satisfying solutions are reached in a simple version of the Level 3 approaches, additional sensors may be added while recursively checking that solutions remain sensible. A possible strategy would be to rebuild the approach from Aircraft Sensor Health Monitoring for omega$_z derived from omega_x, omega_y and drift angle$: $delta_b eta. Further use of this model could be expanded once it has been proven to work.$

Validation

Debugging a fault detection for aircraft with around 400 sensors will certainly prove tedious. An approach using about 10 sensors might be easier to validate. Steps 1 and 2 will not need much validation after verifying the relatively simple logic. Step 3 however will create a physical model. This model can be validated by using known physical correlations to reference sensors to the simulated model and calculate the difference over an entire flight. This in turn can be investigated to validate the method. Further validation methods may be employed later on once more is known about the topic.

3.2 Methodology

Since SHM is a multifaceted problem that represents a grand undertaking with many moving parts and optimizable parameters, algorithms and systems that may be implemented, it is firstly considered necessary to develop an ecosystem and a process in which SHM algorithms may be tested, evaluated and optimized. To develop said ecosystem, the turbine model for development of Human-Machine Systems is featured [FLEM22]. Its process can be applied to this work without much changes. Starting from Ideation stage that represents the beginning phase as well as the first Chapter of this work. Within Ideation phase the project orientation takes place. Answering the question of which goals are considered viable within the given constraints of time and manpower as well as ressources such as knowledge and available tooling. After Ideation Phase the vertical movement representing creativity and innovational ideas is centered into a level-headed assessment of the problem containing goals and boundary conditions. Using these condensed ideas and values, the Research and Technology Assessment Phase is kicked off, represented by Chapter 2 of this work in which various technologies are examined for usage within the given problem context. Based upon a wide and thorough research feasible ideas and approaches can now be arranged within an infrastructure to solve the given problem. Fitting together various approaches from the literature like puzzle-pieces and condensing the ideas into a more refined concept, concluding literature review and representing a first draft. Using this first draft, the development cycle can now be started which can also be represented by the spiral model displayed in figure **??**. The challenge of this third stage is now to implement the draft given in the previous stage, this gets accomplished by bouncing from an exploration step in which the theoretical constructs from the previous steps are implemented into a review step in which the systems are rated upon usability. This also represents the procedure of chapter 4 in which the theory from chapter 2 is implemented and developed. After having then developed a prototype of this work we can assess the results for our test cases (sensor failures) in chapter 5.

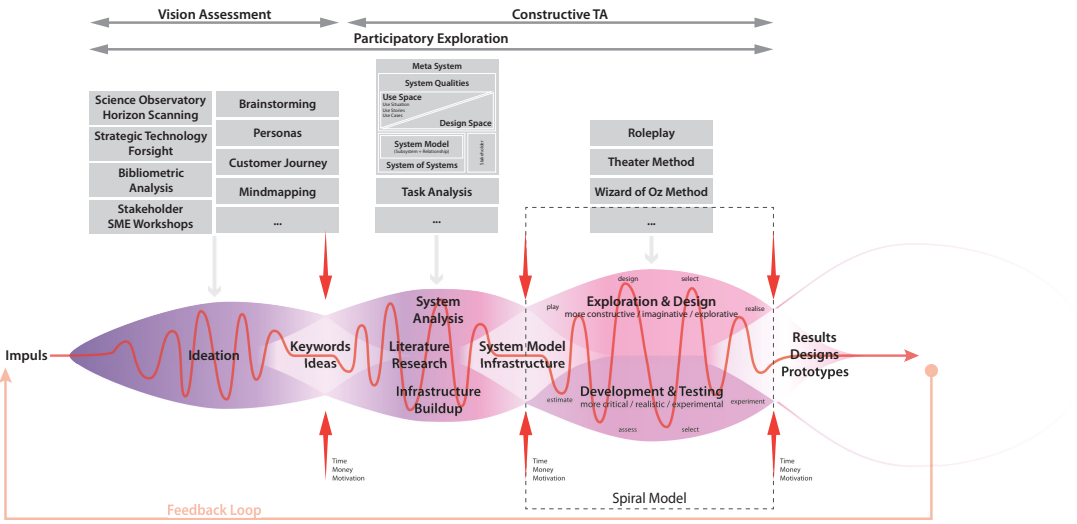


Figure 3.1: The innovation and exploration turbine in the context of this work as a two times nested feedback loop. [FLEM22]

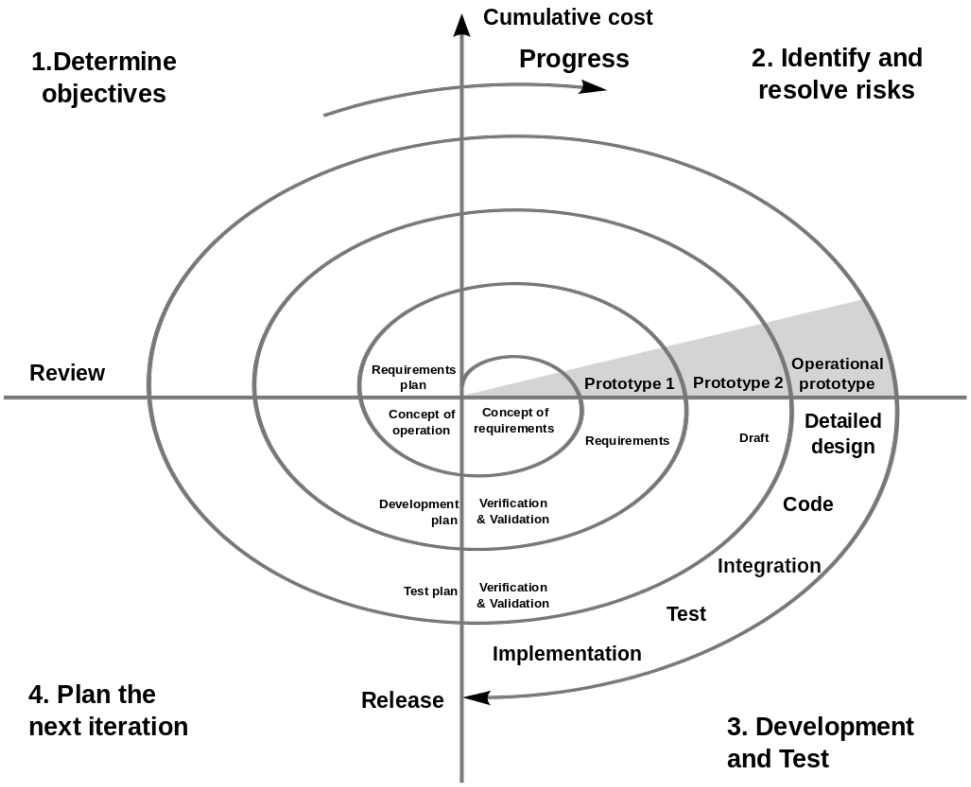


Figure 3.2: The spiral model featured within the innovation turbine [BOEH86]

4 Methods

The implementation efforts of this thesis are condensed into the following chapter. A holistic perspective on the data process is described in the following step following from the urgency that Sensor Health Monitoring needs to be perceived within its ecosystem. A SHM thus is directly reliant upon the data in the first place. It is then also reliant upon the configuration metadata since it will contain necessary information for processing the data further within the SHM. To then close the feedback loop it is also vital to generate a sensible data display that allows quick evaluation of the SHM since manual evaluation methods would slow the development process by orders of magnitude.

4.1 Introduction

The implementation can be detailed by figure **??**. After having uploaded the Dataset the relevant information on the sensors as well as the aircraft properties is gathered and condensed into a single configuration metadata-set represented by a JSON file. This JSON file is then appended to the Dataset within the skystash architecture represented by the Online side that is accessed by the API. Processing the data now becomes a clean blackbox operation that is only fed by data received by the API returning its report containing notable occurences via the API and storing all relevant information online. Since this software is developed for flight operations and an interactive visualization of the report data facilitates evaluation of the generated report data a User Interface within a dashboard application is developed. Additional benefits contrary to the generation of PDF reports are dynamic updates as well as interactivity and an agile update cycle considering the spontaneous emergence of bugs.

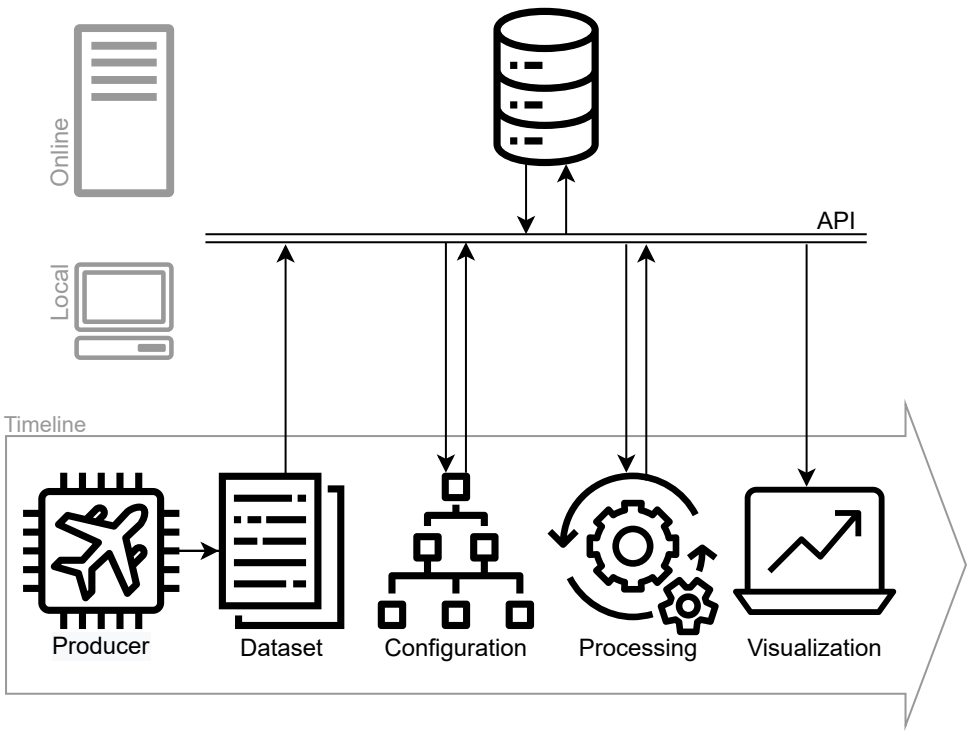


Figure 4.1: The data toolchain prospectively used for Sensor Health Monitoring

4.2 Data Parsing

Since the SHM has to run on something, the original data is briefly mentioned for completeness.

4.2.1 Parsing of ISTAR Data

Istar data is parsed for each flight for each sensor. Istar config data is present for each flight in the shape of an imcexp file.

ISTAR data originates from the ISTAR's DAQ system in the shape of .raw files for each parameter and is parsed and uploaded to the stash. It is then accessible via the stash api and can also be inspected in the stash webview. After the Dataset generation step the metadata available is only the one from the ISTAR DAQ system. The data is present in the shape of Projects->Collections->Series

The metadata is currently only the one from the ISTAR DAQ system. Since this metadata is not explicitly useful in itself it will be enriched further within the Configuration step (See figure **??**).

4.3 Configuration-Metadata

get imcexp data enrich excel data -aircraft data -shm data -limits -type -tags -reference COS get excel data

generate merged view for metadata.

The metadata uploaded to the stash in itself is not very expressive and needs to be refined further into a usable, concise format. This step details the considerations and implementations taken to be able to generate a useful metadata model that applies necessary information to the dataset within the skystash.

4.3.1 Basics: Metadata Collection, formatting into SOIL

The configuration of the ISTAR is of great importance for the data processing. Also of great importance is the knowledge database which is currently in the shape of an excel document. Attributes that are generally transmitted within the DAQ configuration are sampling rates, data origins and other miscellaneous information related to the system. Not contained are information about general sensor description, position of sensors, overall setup of the aircraft sensor architecture and check limits for data values.

4.3.2 Parsing and assignment of ISTAR data config

-Additional Metadata -physical limits -amplitude limits -tag (physical entity) -reference ()

4.4 Data Processing

4.4.1 Software architecture considerations

Careful consideration needs to be given to the workflow of the level 1 check to allow scalability and minimal manual interaction in later stages. To achieve this architecture, manual steps are reduced as far as possible. However, some level of configuration must be implemented. Otherwise only relational sensor behaviour could be detected. Meaning that sensor faults occuring temporarily within an experiment can be detected but permanent behaviour is not noticed by a relative algorithm

4.4.2 Check 1 Implementation

For implementing the level 1 checks, the measured data needs to be compared to the expected data. To gain insight into what the the expected data is, the configuration file of the data acquisitioning system needs to get parsed. Luckily, the DAQ's format is a .zip directory. The 7zip command line tool gets used for opening the configuration file since the configuration file's format is not fully complying to the zip standard making several python libraries fail during the process. This stems from an issue with the zip header and footer parts of the file that are not at expected places (i.e. the front the back). Once opened, the configuration file contains multiple files as well as an essential xml file that contains the needed sensor metadata.

Missing datasets can now easily be found by comparing the expected values from the configuration file to the actual generated data.

For the second step.

All Parameters present?

4.4.3 Check 2 Implementation

Since the IMC DAQ System resamples the 20Hz Sensors to a straight timeframe data gets lost in the process. This makes potentially noisy sensors output the same value twice in a row since the DAQ hasn't yet received the new sensor data and outputs the same value multiple times in a row. Even sensors with a high noise ratio such as the fine part of a gps latitude signal outputs the same value twice in a row which is highly unlikely to happen on a statistical basis. Since this occurs quite often the question arises if the actual sampling may be lower than the actual data.

No signal (value)

A check implementation counterchecking the sensors sampling that generates fault detections based on deviation from predefined sampling. This takes into account the DAQ config since it contains the current sampling rate. Minimizing false positives.

Out of range

Generally, predefined limits are checked within SHM Level 2. For the first step, values are checked whether they are within a predefined range. This means e.g. a range of -1000-40000 ft for barometric altitudes. or 1Pa-120000Pa for static pressure ports. Of course, this selection is biased and may not be accurate for most mission profiles like atmospheric research aircraft that cruise in altitudes of up to 45,000 ft MSL.

movement too high

The next category examining the sensor behaviour can be classified into sensor movement being too low or sensor movement being too high. Movement being defined within this context as the difference of a new sensor value to the previous one. Of course, this topic could be examined within a depth that may exceed this work such as estimating white noise using discrete fourier Transform Subspace Decompositions [HEND08] or statistical methods such as covariance operators.

The goal of this work however, is preservation of scalability, minimal user and configuration inputs. Hence, an approach using a STFT where the amplitude is averaged across all frequencies from its spectrum. This guarantees a generalized feature extraction, meaning standardization. Based on this spectral analysis, the logarithmic order of the variable can be estimated within its dataset. Previously, the frequency is assumed via time difference of start and end time divided by $n_{datapoints}$. The STFT also implements preprocessing steps that would otherwise be necessary such as translating the signal by its average value as well as scaling it by its standard deviation. Would one be interested in examining a signal using functions from a statistical toolbox, a similar result may be achieved by performing the previous steps of averaging and scaling the signal and then examining the variance within a moving window of the signal, similar to the STFT. The size of the moving window for such an analysis is defaulted to 256 data points for each signal. Certain issues with methods of moving windows are however, that they are not very meaningful for the beginning and ending of datasets within which the window is not fully occupied. This may lead to spikes for the STFT, which is why it is found that the STFT is generally more powerful to examine low sensor movement.

TODO: Comparison variance, standard deviation, mean noise. Scalability for other sensors.

Noise Tracking using DFT can be used to estimate noise combined with a strenuous effort Since this however would exceed the scope of this work, a simpler algorithm is implemented by comparing moving window variance as well as STFT overall amplitude estimations.

Limits of variance and stdev.

To better approximate the errors concerning white noise, a STFT is employed.

White noise approximation methods. Rolling window variance.

movement too low

4.4.4 Check 3 Implementation:

I lied about single source of config. Level 2 and 3 also have their own config.

avoid weighting by structuring checks into a tree format for independent aircraft state variables.

Aims of Level 3 Implementation are to model the aircraft's state in a reference system. This reference system shall be geodetic and fixed to the earth while the aircraft moves through it. Moving aircraft coordinate systems like the aerodynamic and the along track Coordinate system may be derived from its geodetic position using ?angles and velocities.

Interfacing: Clean interfaces are generated throughout the model. Enabling a standardized state vector x. Meaning that A remains standardized for any aircraft while B, U and L need to be adjusted for any changes to the aircraft or sensor data.

The integration step may be omitted in early design stages since the necessary equations and equilibriums of Forces and Moments could only be modelled linearly while neglecting various unknown factors like shifting CG due to fuel burn, actual Inertia of Aircraft and aicraft mass. Hence, this step may be implemented if time allows it.

Finding a reference state

Necessary to compare parameters is a common reference system. Hence, reference systems are examined upon suitability for usage of parameter transformations.

A starting point which shall be considered is the geodetic reference. it measures the aircraft's position by its displacement from the previously (ref?) discussed WGS84 system. Meaning that altitude gets measured as the orthometric height (see ellipsoid height-geoid height).

Latitude and Longitude form the x and y axis of the COS. True heading forms the reference heading within the geodetic system.

All aircraft parameters related to motion and position of the aircraft should be attempted to be condensed into this form.

Dynamic configuration and correlation of parameters

Motivation: a dynamic description for parameter correlations is searched in order to implement an efficient and quick way to assign parameter correlations. It is then assumed that a tree structure is fit for this task since parameters can be correlated to other upstream parameters in a single flow direction.

To attach meaning to the parameters, they are specifically tagged with descriptions of their function. A simple example that is implemented is that of *static pressure*. The tag is defined in the excel configuration and

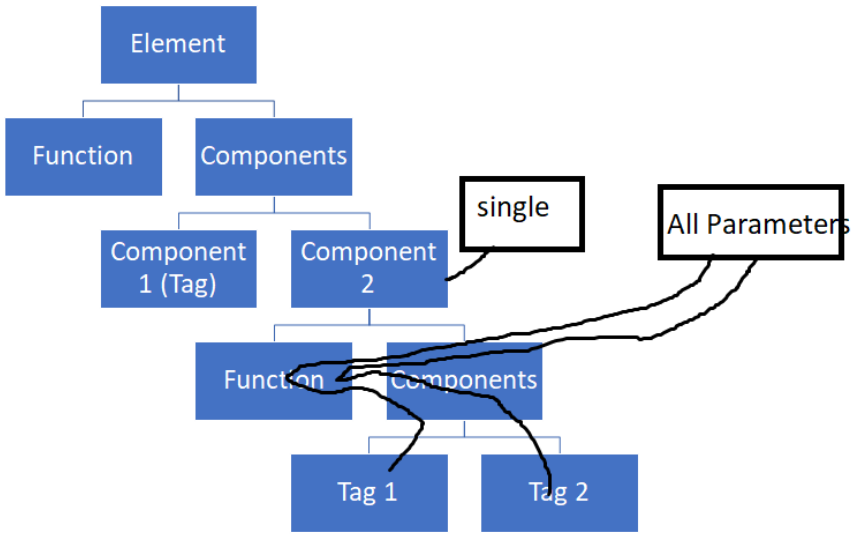


Figure 4.2: General structure of physical correlation setup

furthermore gets configured into a dynamic configuration JSON-file. This file allows specification of sensors into subclasses and divides the aircraft into its degrees of freedom. This tree structure is then parsed and calculates a value for each level of detail taking into account its lower lying neighbours. Some weighting also has to be considered since a number of redundant pressure sensor could skew results against a single GNSS parameter. Thus a condensed parameter is calculated for each degree of freedom that is based on predefined algorithms and tags that can be freely defined within the config file that are then recursively parsed within the tree structure.

In figure **??** the general structure of the dynamic configuration in JSON format is shown. The configuration parent is an element similar to the configuration in SOIL-format [BODE21]. In our case the ISTAR-aircraft. The istar aircraft contains top-level components that are the independent state variables of the aircraft. Generally a component can contain other components or parameter tags that describe parameters that are referenced. A component can also contain a function that transforms its parameters into the parent parameter.

Residual generation

Based upon this previously defined model that is based on the aircraft configuration as a tree structure, residuals are generated based upon the difference of the parameters to the top level parameters (Single, fused value) and the single transformed sensor values (All Parameters) as previously discussed in Table **??**.

An alternate method for fusion of redundant sensors is proposed that is similar to voting algorithms employed by Flight Control Systems of [TISC18]. An issue for voting algorithms is the abrupt exclusion of single sensors once their difference to the other sensors is too large. Hence, a weighting strategy for averaging is proposed that is based on the value difference.

An example is shown in figure **??**

	A	B	C
A	0	0.25	2
B	0.25	0	1.75
C	2	1.75	0
Sum	2.25	2	3.75

This method tries to account dynamically for sensor distance by scaling parameters to small values compared to their neighbours. This works for a minimum of three redundant values.

Based on the single sensor values a distance matrix for $d_{i,j}$ is set up.

The column sum is then further defined $\hat{d}_i$. To generate a ratio we define:

$$w_i = \frac{1}{\hat{d}_i} \tag{4.1}$$

Since we desire that $\sum w_i \stackrel{!}{=} 1$ we need to scale the ratios using:

$$\bar{w}_i = \frac{w_i}{\sum w_i} \tag{4.2}$$

With this ratio we can now calculate the new fused value:

$$\bar{x} = \sum_{i}^{n} x_i \cdot \bar{w}_i \tag{4.3}$$

This works out to a fused value that is shown in **??**.

Compare results with averaging here using figure.

To further increase sensitivity for distance we can replace the term in equation **??** with a quadratic term:

$$w_i = \frac{1}{\hat{d}_i^{\,2}} \tag{4.4}$$

Residual Interpretation

The residuals are then checked against the limits of the standard deviation of the normal signal for a highpass filtering generated by taking a moving window of 256 points. If any value falls outside these values of the standard deviation of the condensed signal it is noted within the report. A fixed residual limit as well as a probability density function implemented in works such as [SVÄR14] may be implemented at a later point.

Examining Altitude Sensors

As seen in **??** gps altitudes have a base mean value of difference of about 3-4 meters. During flight level changes the base level changes significantly. Further investigation is needed how these changes may arise.

One possible approach would be considering different placements of gps antennas within the airframe. since the IMAR's position is precisely known and lies around the center of gravity but the ASCB's gps position is not known exactly the process is not facilitated. However using the process of lever arms the position could be roughly estimated and compensated from the altitude difference in figure **??**.

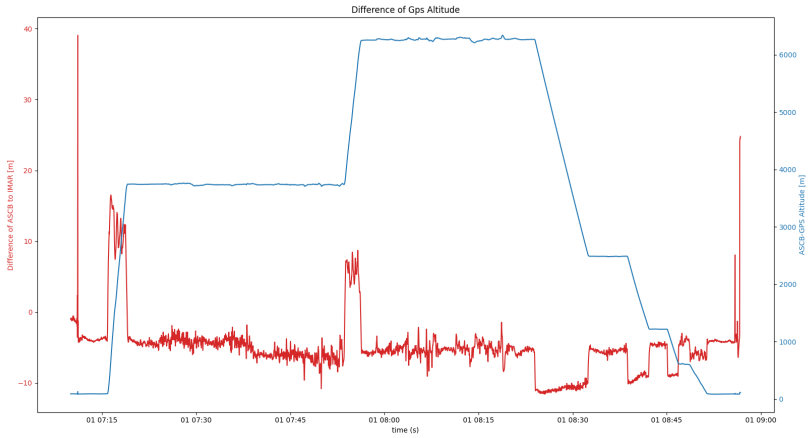


Figure 4.3: Comparing the GPS sensors from the internal aircraft GPS (ASCB) to an experimentally installed GPS system (IMAR)

Another incurring deviation is investigating the 40m/150ft offset for gps altitudes. Possible causes may be Uncorrected Ellipsoid gnss altitudes. However, this appears unlikely since generally the offset in the region would be added and not subtracted [further investigation needed].

Difficulty diagnosing sensors without previous knowledge. limitations within sensor behaviour. Checks include: range, too much sensor movement, too few sensor movement.

Relative examinations possible for errors occuring for finite time within experiment but not if all of the experiment data is corrupt.

altitude comparison

plot here: -gps to gps -gps to baro. Beta overlayed as well as Ma Number

some mean ground has to be found to crossreference the various altitudes.

For standardization. The SI-unit meters is used for altitudes.

In the first draft. The altitude is sampled with 1Hz for computational speed and since gnss update rates are updated in a similar frequency.

Physical

GNSS

The following briefly glosses over GNSS and does not delve into the technical details. it rather tries to present GNSS Systems as a black box and examines inputs and outputs.

Generally speaking, GNSS systems calculate position based on run times to satellites. For output GNSS systems return position values for the ellipsoid model of the earth. Since the difference between MSL and ellipsoid varies for up to 100m vertically based on the receivers positions on the earth. Mostly, the difference is in the 40-50m range in germany.

Explain here why orthometric and ellipsoid value differs (vector difference, cg)

GNSS-units work with geoid models to deliver an orthometric value. Different Geoid models have different drawbacks and resolutions. For aviation use, the World Geodetic System (WGS84) generally proposes a geoid model that is used. Within this works scope it is omitted to implement an existing geoid model into the software-loop to reserve this workload for a later time.

4.5 Report Visualization

Motivation:

An interactive data report is chosen for data analytics since it works more efficiently and rather follows the paradigma of a single source of truth since it allows for dynamic updates and thus does not represent a data source in itself but rather view on the source.

Requirements: needs to be able to quickly gain an overview over state of sensors. Details can be looked up later. UX-centric design. Level1: Quickly see if many sensors are missing Level2: Identify outliers

4.5.1 Visualization

of missing parameter list

After monitoring, a list of missing parameters is generated. To show and detect this info quickly, a speed gauge style display is chosen to display this scalar value.

Displaying notable occurences of single sensor examination

To quickly get an overview of missing parameters, a timeline graph is implemented, showing cumulative errors over the flight. This allows a quick overview over all parameters.

Possible features: implement altitude graph later to contextualize sensor behaviour.

Examining sensor interactions

whole flight: occurences Single sensor: mean value, residual, stdev

It shows that while conservatively chosen ranges do not indicate many occurences, tighter limits enable more detailed monitoring.

4.5.2 Adapt sensitivity and algorithms (balance false positives)

The validation loop is driven by previously known error types (See OneNote Sensor Errors)

4.6 Conclusion

This parameter showed the methods used and the considerations made for the implementation of this SHM. Various methods and approaches were implemented to check for sensor faults. Next to the straight implementation, focus was put into developing solid interfaces to ease further implementation of future methods of fault detection. This applies for Levels 1, 2 as well as 3 which in this work was only minimally implemented to allow this prototype to see the light of day.

5 Results and Discussion

This chapter presents the findings of this work as well as the work that has been done and then discusses them in a wider context.

5.1 Metadata structure

Is this a sensible result representing reality and fulfilling demands in respect to working SHM and FAIR principles?

Thoughts on data structure flow.

A common ground has been found for the upload so that alterations to further processing parts of the cycle don't necessitate a full reupload of the original data. This common ground asserts that original is uploaded once with the downside that the DAQ configuration may not be fully interpretable at this stage due not being self describing metadata.

The following step of configuration solves this problem then by collecting and merging available metadata into a common scheme. This step works upon the DAQ metadata that is already uploaded to the skystash and then fuses it with other metadata sources that are locally available to generate a holistically valid metadataset. Notable is that this step will certainly be part of rather constant modification due to SHM configurations that are also provided within this step. Also the metadata schema is far from complete and will be in need of constant expansion to compensate the complex dimensions of flight test data as well as relevant test configuration data that also may be taken into consideration within this step **??**. Since this step is far less computationally intensive than the parsing process this is considered as an acceptable tradeoff within this process.

This work's approach tries to implement the FAIR principles by using configuration files for the check algorithms that enable systemic control and extendability as well as by using a single source of truth, cleanly encapsulating every single step of the SHM process. However at the current point the metadata scheme is not clearly defined and work for a later point. Possible implementations may include the SOIL-scheme or other schemes that are possibly already suited for describing data quality and correlations. By also including the processing metadata configuration within the metadata, a reusability and traceability of the report logic is guaranteed.

An interactive frontend for the data report enables Accessibility of the SHM. Enabling a data quality overview for an entire flight dataset at a glance. Searchable metadata tags enable a dynamic search within the MongoDB Database.

5.2 Check testing 1,2,3 (test with real data)

Show plots of

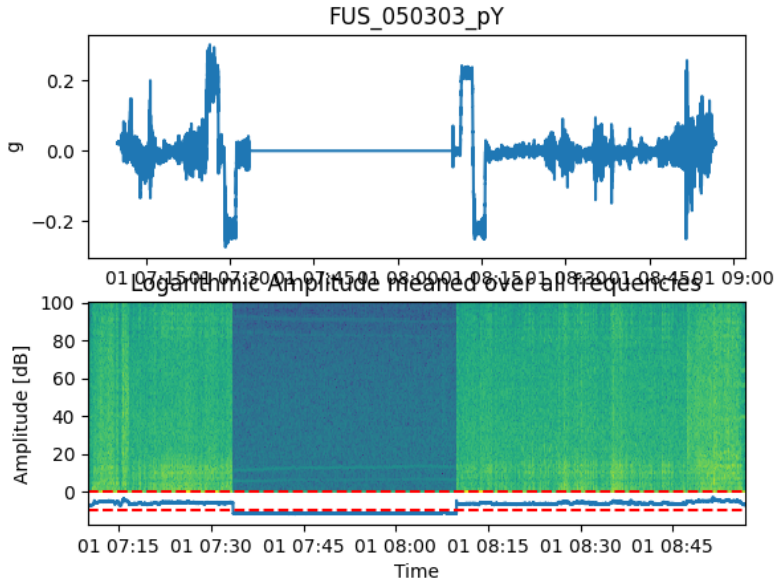


Figure 5.1: Capturing Errors

Check with faulty data (Plots and comparisons)

see onenote sensor errors

AE-errors F89: $\text{FUS}_0 50303$

-stft from onenote. show progression from spectrogram to averaged amplitude over time

20210923$_A VIA2terFlug_{(1)} : FUS_0 50303_p Y, FUS_0 50209_p Z(offset), DFUS_2 30306$

5.2.1 Comparison with previously developed FMEAs

PTF,Kalman,PCA

not nearly as good enough. However, focus of this work is to start off error detection within a FAIR context to generate interfaces for the skystash.

What could be reached with other FMEAs? Correlation Applications as well as Neural Networks.

5.2.2 Rate check quality with parameters

The data quality parameter fine tuning will still have to go a long way to fulfill all necessary and required conditions.

To summarize, the algorithms employed generally work by inspecting limits of parameter Series. These limits can also be set for modified parameter series that have e.g. been differentiated or fourier transformed. The interfaces have been created to allow for further implementation and optimization of parameters. The limits are defined within the configuration file (currently in excel format) and the Series transformation may also be customly defined.

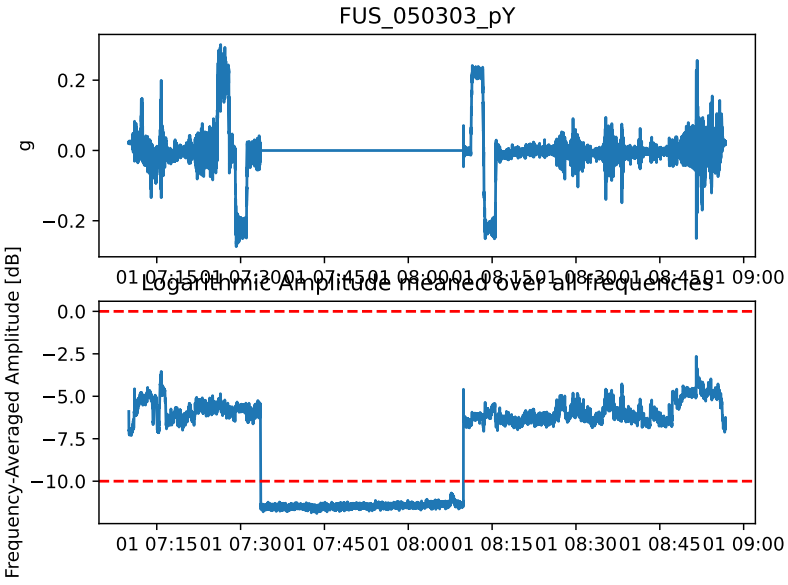


Figure 5.2: Capturing Errors

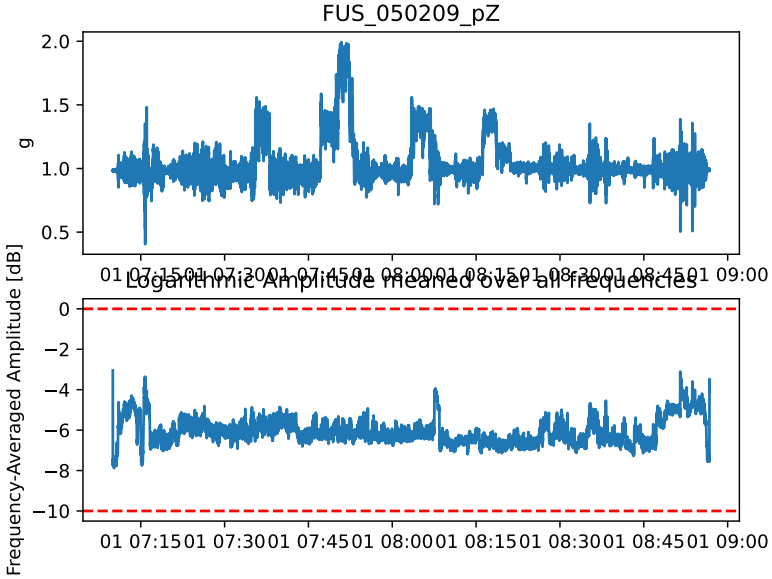


Figure 5.3: Capturing Errors

As for general formats it has been found to be difficult to generalize for all parameter formats since many employ strictly discrete values. For error detection using single sensors only, it is possible to divide into the following categories: Value is out of bounds, the movement of the value is to high or the movement of the value is too low.

In level 3, fault tolerant sensors methods are employed that use hardware as well as analytical redundancy. [ISER06, p.355-365]

5.3 Implementation ecosystem

How well is the implementation in the ecosystem? Modularity of components and extensibility with new functions.

-Embedding within the stash ecosystem. Clean Interfaces?

5.3.1 Toolchain FTI-tool

The following toolchain emerges for use of fti microservices.

Modular design allows encapsulation of algorithms as well as quick reaction towards faults and independence of fault modes. I.e. wrong config does not require the data to be reuploaded.

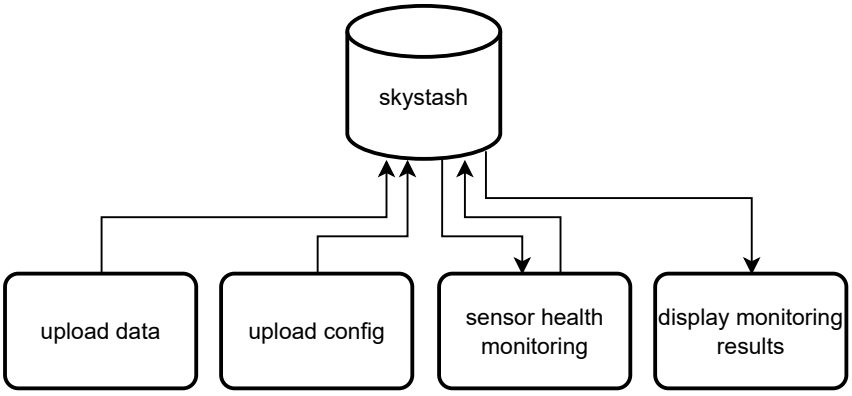


Figure 5.4: The emerging FTI-Toolchain

5.4 Discussion/Potential

Implementation list:tryouts -lvl3 rigid body simulation for flight dynamics -lvl3 implementation of multiple COS and their respective conversions -lvl3 residual interpretation (better than highpass filtering and checking for std)

-Real time implementation -Pseudo transfer functions (Generate a real time correlation matrix (MArkov parameters) and generate residuals based on that) -PCA -better parameter fusion algorithms (voting, or others see GNSS-RAIM) -aaim, -external factors integrity monitoring - GANomaly approach (neural network) since an aircraft dataset is similar to images in that it is a multidimensional dataset

-more and better evaluation of the report display to satisfy innovation turbine requirements.

5.5 Conclusion

These results show a promising prototype for a Sensor Health Monitoring Infrastructure. It establishes Interfaces and then

6 Conclusion

References

[ALJA15] K. F. Aljanaideh and D. S. Bernstein. "Aircraft Sensor Health Monitoring Based on Transmissibility Operators". en. In: *Journal of Guidance, Control, and Dynamics* 38.8 (Aug. 2015), pp. 1492–1495. ISSN: 0731-5090, 1533-3884. DOI: 10.2514/1.G001125.

[ARTS] E. Arts, M. Bäßler, S. Haufe, A. Kamtsiuris, H. Meyer, C. Pätzold, R. Schültzky, and M. Tchorzewski. "DIGITAL TWIN FOR RESEARCH AIRCRAFT". en. In: ().

[BODE21] M. Bodenbenner, M. P. Sanders, B. Montavon, and R. H. Schmitt. "Domain-Specific Language for Sensors in the Internet of Production". en. In: *Production at the leading edge of technology*. Ed. by B.-A. Behrens, A. Brosius, W. Hintze, S. Ihlenfeldt, and J. P. Wulfsberg. Series Title: Lecture Notes in Production Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2021, pp. 448–456. ISBN: 978-3-662-62137-0 978-3-662-62138-7. DOI: 10.1007/978-3-662-62138-7_45.

[BOEH86] B. Boehm. "A spiral model of software development and enhancement". en. In: *ACM SIGSOFT Software Engineering Notes* 11.4 (Aug. 1986), pp. 14–24. ISSN: 0163-5948. DOI: 10.1145/12944.12948.

[DIN77] DIN. *DIN25424_Fehlerbaumanalyse.pdf*. 1977.

[DIN95] DIN. *DIN 1319-1 Meßtechnik Grundbegriffe.pdf*. 1995.

[DLR18] DLR. *DLR-fleet*. 2018.

[DLR20] DLR. *DLR-Forschungsflugzeug ISTAR in 4K – Ein neuer Stern am Himmel der Luftfahrtforschung*. Feb. 2020.

[FAA08] FAA. *Federal_Radionavigation_Plan*. 2008.

[FLEM22] F. O. Flemisch, M. Preutenborbeck, M. Baltzer, J. Wasser, C. Kehl, R. Grünwald, H.-M. Pastuszka, and A. Dahlmann. "Human Systems Exploration for Ideation and Innovation in Potentially Disruptive Defense and Security Systems". en. In: *Disruption, Ideation and Innovation for Defence and Security*. Ed. by G. Adlakha-Hutcheon and A. Masys. Series Title: Advanced Sciences and Technologies for Security Applications. Cham: Springer International Publishing, 2022, pp. 79–117. ISBN: 978-3-031-06635-1 978-3-031-06636-8. DOI: 10.1007/978-3-031-06636-8_5.

[HAND17] A. Handl and T. Kuhlenkasper. *Multivariate Analysemethoden*. de. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. ISBN: 978-3-662-54753-3 978-3-662-54754-0. DOI: 10.1007/978-3-662-54754-0.

[HART22] P. Hartmann and S. Seitz. *Navigation-Sensordatenfusion*. 2022.

[HEND08] R. Hendriks, J. Jensen, and R. Heusdens. "Noise Tracking Using DFT Domain Subspace Decompositions". en. In: *IEEE Transactions on Audio, Speech, and Language Processing* 16.3 (Mar. 2008), pp. 541–553. ISSN: 1558-7916. DOI: 10.1109/TASL.2007.914977.

[ISER] R. Isermann and P. Ball. "TRENDS IN THE APPLICATION OF MODEL-BASED FAULT DETECTION AND DIAGNOSIS OF TECHNICAL PROCESSES". en. In: ().

[ISER06] R. Isermann. *Fault-Diagnosis Systems*. en. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. ISBN: 978-3-540-24112-6 978-3-540-30368-8. DOI: 10.1007/3-540-30368-5.

[ISER11] R. Isermann. *Fault-diagnosis applications: model-based condition monitoring: actuators, drives, machinery, plants, sensors, and fault-tolerant systems*. eng. Berlin Heidelberg: Springer, 2011. ISBN: 978-3-642-12766-3.

[ISO15] ISO. *Data-quality_Information and data quality_Concepts and Measuring.pdf*. Nov. 2015.

[ISO75] ISO. *Standard Atmosphere ISO 2533.pdf*. 1975.

[ISO97] ISO. *ISO5725-1_Accuracy of measurement methods and results.pdf*. 1997.

[KHAL22a] A. Khalil, M. Al Janaideh, K. F. Aljanaideh, and D. Kundur. "Transmissibility-Based Health Monitoring of the Future Connected Autonomous Vehicles Networks". en. In: *IEEE Transactions on Vehicular Technology* 71.4 (Apr. 2022), pp. 3633–3647. ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2022.3151326.

[KHAL22b] A. Khalil, K. F. Aljanaideh, and M. Al Janaideh. "Transmissibility-based Fault Detection in Systems with Unknown Time-Varying Parameters". en. In: *2022 American Control Conference (ACC)*. Atlanta, GA, USA: IEEE, June 2022, pp. 1947–1951. ISBN: 978-1-66545-196-3. DOI: 10.23919/ACC53348.2022.9867203.

[KUTS75] F. v. Kutschera. *Sprachphilosophie*. 2., völlig neu bearb. u. erw. Aufl. Uni-Taschenbücher ; 80. München: Fink, 1975. ISBN: 978-3-7705-1182-2.

[LI15] P. Li and Q. D. Vu. "A simple method for identifying parameter correlations in partially observed linear dynamic models". en. In: *BMC Systems Biology* 9.1 (Dec. 2015), p. 92. ISSN: 1752-0509. DOI: 10.1186/s12918-015-0234-3.

[LIE13] F. A. P. Lie and D. Gebre-Egziabher. "Synthetic Air Data System". en. In: *Journal of Aircraft* 50.4 (July 2013), pp. 1234–1249. ISSN: 0021-8669, 1533-3868. DOI: 10.2514/1.C032177.

[MEYE20] H. Meyer, J. Zimdahl, A. Kamtsiuris, R. Meissner, F. Raddatz, S. Haufe, and M. Bäßler. "Development of a Digital Twin for Aviation Research". en. In: (2020). Artwork Size: 8 pages Medium: application/pdf Publisher: Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V. Version Number: 1.0, 8 pages. DOI: 10.25967/530329.

[PEAR01] K. Pearson. "LIII. *On lines and planes of closest fit to systems of points in space*". en. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (Nov. 1901), pp. 559–572. ISSN: 1941-5982, 1941-5990. DOI: 10.1080/14786440109462720.

[PEÑA23] J. Peña-Consuegra, M. R. Pagnola, J. Useche, P. Madhukar, F. D. Saccone, and A. G. Marrugo. "Manufacturing and Measuring Techniques for Graphene-Silicone-Based Strain Sensors". en. In: *JOM* 75.3 (Mar. 2023), pp. 631–645. ISSN: 1047-4838, 1543-1851. DOI: 10.1007/s11837-022-05550-3.

[SHOE87] S. Shoemaker and S. Blackburn. "Spreading the Word." In: *Noûs* 21.3 (Sept. 1987), p. 438. ISSN: 00294624. DOI: 10.2307/2215195.

[SLAT98] J. A. Slater and S. Malys. "WGS 84 — Past, Present and Future". In: *Advances in Positioning and Reference Frames*. Ed. by K.-P. Schwarz and F. K. Brunner. Vol. 118. Series Title: International Association of Geodesy Symposia. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 1–7. ISBN: 978-3-642-08425-6 978-3-662-03714-0. DOI: 10.1007/978-3-662-03714-0_1.

[SMIT] S. W. Smith. "The Scientist and Engineer's Guide to Digital Signal Processing". en. In: ().

[SVÄR14] C. Svärd, M. Nyberg, E. Frisk, and M. Krysander. "Data-driven and adaptive statistical residual evaluation for fault detection with an automotive application". en. In: *Mechanical Systems and Signal Processing* 45.1 (Mar. 2014), pp. 170–192. ISSN: 08883270. DOI: 10.1016/j.ymssp.2013.11.002.

[TEUN17] P. J. Teunissen and O. Montenbruck, eds. *Springer Handbook of Global Navigation Satellite Systems*. en. Cham: Springer International Publishing, 2017. ISBN: 978-3-319-42926-7 978-3-319-42928-1. DOI: 10.1007/978-3-319-42928-1.

[TISC18] M. Tischler. *Advances in Aircraft Flight Control*. en. Ed. by M. B. Tischler. 1st ed. Routledge, Apr. 2018. ISBN: 978-1-315-13682-0. DOI: 10.1201/9781315136820.

[WILK16] M. D. Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". en. In: *Scientific Data* 3.1 (Mar. 2016), p. 160018. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.18.

[XIAO06] F. Xiao, S. Wang, and J. Zhang. "A diagnostic tool for online sensor health monitoring in air-conditioning systems". en. In: *Automation in Construction* 15.4 (July 2006), pp. 489–503. ISSN: 09265805. DOI: 10.1016/j.autcon.2005.06.001.

[ZHAN08] Y. Zhang and J. Jiang. "Bibliographical review on reconfigurable fault-tolerant control systems". en. In: *Annual Reviews in Control* 32.2 (Dec. 2008), pp. 229–252. ISSN: 13675788. DOI: 10.1016/j.arcontrol.2008.03.008.