INL/EXT-16-39851

Light Water Reactor Sustainability Program

Data Analysis Approaches for the Risk-Informed Safety Margins Characterization Toolkit

September 2016

DOE Office of Nuclear Energy

INL/EXT-16-39851

Light Water Reactor Sustainability Program

Data Analysis Approaches for the Risk-Informed Safety Margins Characterization Toolkit

**D. Mandelli, D. Maljovec, A. Alfonsi, C. Parisi,
P. Talbot, C. Picoco, J. Cogliati, C. Wang, C. Smith, C. Rabiti**

September 2016

**Idaho National Laboratory
Idaho Falls, Idaho 83415**

http://www.inl.gov/lwrs

**Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517**

ABSTRACT

The RISMC project aims to develop new advanced simulation-based tools to perform Computational Risk Analysis (CRA) for the existing fleet of U.S. nuclear power plants (NPPs). These tools numerically model not only the thermal-hydraulic behavior of a reactor primary and secondary systems but also external event temporal evolution and component/system ageing. Thus, this is not only a multi-physics problem but also a multi-scale problem (both spatial, μm-mm-m, and temporal, ms-s-minutes-years). As part of the RISMC CRA approach, a large amount of computationally expensive simulation runs may be required. In addition, these uncertainties and safety methods usually generate a large number of simulation runs (database storage may be on the order of gigabytes or higher). During the FY2016, we investigated, implemented and applied several methods and algorithms to analyze these large amounts of time-dependent data. The scope of this report is to present a broad overview of methods and algorithms that can be used to analyze and extract information from large data sets containing time dependent data. In this context, "extracting information" means constructing input-output correlations, finding commonalities, and identifying outliers. Some of the algorithms presented here have been developed or are under development within the RAVEN statistical framework.

.

CONTENTS

FIGURES

TABLES

ACRONYMS

CDF	Cumulative Distribution Function
CD	Core Damage
DOE	Department of Energy
DG	Diesel generator
EOP	Emergency Operating Procedures
ET	Event-Tree
FT	Fault-Tree
DTW	Dynamic Time Warping
GUI	Graphical User Interface
IE	Initiating Event
INL	Idaho National Laboratory
LHS	Latin Hypercube Sampling
LOOP	Loss Of Offsite Power
LWR	Light Water Reactor
LWRS	Light Water Reactor Sustainability
MC	Monte-Carlo
NPP	Nuclear Power Plant
PDF	Probability Distribution Function
PRA	Probabilistic Risk Assessment
PWR	Pressurized Water Reactor
R&D	Research and Development
RISMC	Risk-Informed Safety Margin Characterization
ROM	Reduced Order Model
RPV	Reactor Pressure Vessel
SAMG	Severe Accident Management Guideline
SSCs	Structures, Systems, and Components
T-H	Thermal-Hydraulics

Data Analysis Approaches for the Risk-Informed Safety Margins Characterization Toolkit

1. INTRODUCTION

In the Risk Informed Safety Margin Characterization (RISMC) [1] approach, what we want to understand is not just the frequency of an event like core damage, but how close we are (or not) to key safety-related events and how might we increase the safety margin. A safety margin can be characterized in one of two ways:

- A deterministic margin, typically defined by the ratio (or, alternatively, the difference) of a capacity (i.e., strength) over the load
- A probabilistic margin, defined by the probability that the load exceeds the capacity. A probabilistic safety margin is a numerical value quantifying the probability that a safety metric (e.g., for an important process observable such as clad temperature) will be exceeded under accident scenario conditions.

The RISMC Pathway uses the probabilistic methods to determine safety margins and quantify their impacts to reliability and safety for existing Nuclear Power Plants (NPPs), i.e., pressurized and boiling water reactors (PWRs and BWRs). As part of the quantification, we use both probabilistic (via risk simulation) and mechanistic (via system simulators) approaches, as represented in Figure 1. Probabilistic analysis is represented by the risk analysis while mechanistic analysis is represented by the plant physics calculations. In the plant simulation, all the deterministic aspects that characterize system dynamics (e.g., thermo-hydraulic, thermo-mechanics, neutronics) are coupled to each other.

The risk simulation contains all deterministic elements that impact accident evolution such as:

- Safety systems control logic
- Accident scenario initial and boundary conditions

In addition to stochastic ones such as:

- System/component failures
- Stochastic perturbation of internal elements of the physics simulation

The stochastic analysis [2] is performed in two steps:

1. Sampling the stochastic parameters, and
2. Evaluating the system response for the given set of sampled parameters

In the RISMC applications, system simulator codes model not only plant thermal-hydraulic, thermal-mechanic, neutronics and ageing behavior but also model external event and human interactions with the plant itself. This is not only a *multi-physics* problem (i.e., different sets of equations are solved) but also a *multi-scale* one (i.e., both temporal and spatial scales). The drawback is that a single plant accident analysis (e.g., prediction of the seismic response of a BWR that underwent to a 60 years life extension license) might require long computational time that grows exponentially if multiple runs (through the sampling process) are needed.

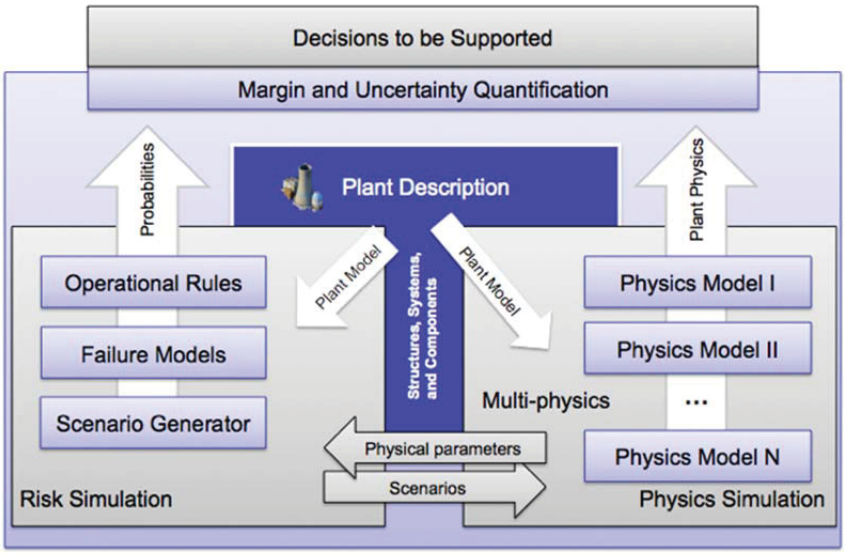

Figure 1. The approach used to support RISMC analysis.

This report completes another step in the development of the RISMC approach to measure risk associated to nuclear power plants using state of the art methods and algorithms. In contrast to state of practice approaches to measure risk which evaluate accident progression using static Boolean logic structures (e.g., Event-Trees, Fault-Trees), the RISMC approach heavily employs system simulators (to evaluate accident progression) coupled with stochastic analysis tools (to alter timing and sequencing of events).

In the past years, the development of the RISMC approach has covered the following topics

- Application of RISMC simulation based PRA methods on a BWR SBO test case [3]
- Comparisons of RISMC simulation based PRA against state of practice methods [4]
- Employment of reduced order model techniques [5] to reduce computation costs of RISMC simulation based PRA methods

During this fiscal year, we have developed several methods and algorithms to analyze the large amount of simulations generated. Classical statistical methods are in fact inadequate to analyze complex time-dependent data sets. This report summarizes the development and testing of these methods.

1.1 Structure of the report

This report is structured as follows:

- Section 2 provides an overview of the RISMC project, the RISMC toolkit and, in particular, of the RAVEN statistical framework
- Section 3 offers a general overview of data mining
- Section 4 provides a more detail overview of the algorithms and methods available from a data mining point of view
- Section 5 shows the challenges related to data mining associated to time-dependent data

- Section 6 gives a broad and detailed overview of the data analysis capabilities of RAVEN
- Section 7 presents several test cases in order to demonstrate the validity of RAVEN as a data analysis tools
- Section 8 shows how advanced data visualization techniques can be employed to analyze complex data sets
- Section 9 presents a set of conclusions and final considerations

2. RISMC OVERVIEW

The RISMC approach employs both deterministic and stochastic methods in a single analysis framework (see Figure 2). In the deterministic method set we include:

- Modeling of the thermal-hydraulic behavior of the plant [6]
- Modeling of external events such as flooding [7]
- Modeling of the operators responses to the accident scenario [8]

Note that deterministic modeling of the plant or external events can be performed by employing specific simulator codes but also surrogate models [5], known as reduced order models (ROM). ROMs would be employed in order to decrease the high computational costs of employed codes. In addition, *multi-fidelity codes* can be employed to model the same system; the idea is to switch from low-fidelity to high-fidelity code when higher accuracy is needed (e.g., use low-fidelity codes for steady-state conditions and high-fidelity code for transient conditions)

In the stochastic modeling we include all stochastic parameters that are of interest in the PRA analysis such as:

- Uncertain parameters
- Stochastic failure of system/components

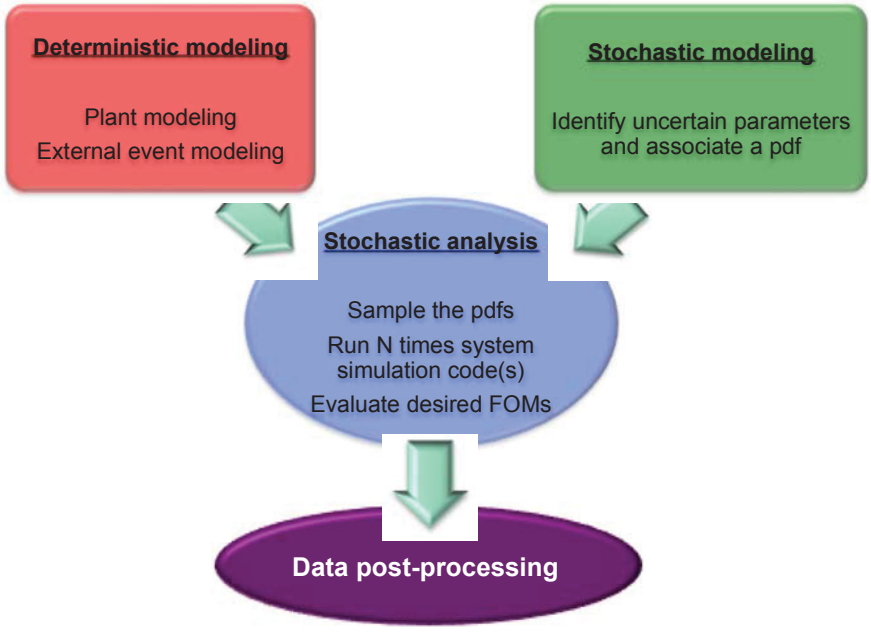

Figure 2. Overview of the RISMC modeling approach.

As mentioned earlier, the RISMC approach heavily relies on multi-physics system simulator codes (e.g., RELAP-7 [9]) coupled with stochastic analysis tools (e.g., RAVEN [10]). From a PRA point of view, this type of simulation can be described by using two sets of variables:

- $\boldsymbol{c} = \boldsymbol{c}(t)$ represents the status of components and systems of the simulator (e.g., status of emergency core cooling system, AC system)

- $\boldsymbol{\theta} = \boldsymbol{\theta}(t)$ represents the temporal evolution of a simulated accident scenario, i.e., $\boldsymbol{\theta}(t)$ represents a single simulation run. Each element of $\boldsymbol{\theta}$ can be for example the values of temperature or pressure in a specific node of the simulator nodalization.

From a mathematical point of view, a single simulator run can be represented as a single trajectory in the phase space. The evolution of such a trajectory in the phase space can be described as follows:

$$\begin{cases} \dfrac{\partial \boldsymbol{\theta}(t)}{\partial t} = \boldsymbol{\mathcal{H}}(\boldsymbol{\theta}, \boldsymbol{s}, \boldsymbol{c}, t) \\ \dfrac{\partial \boldsymbol{c}(t)}{\partial t} = \boldsymbol{\mathcal{C}}(\boldsymbol{\theta}, \boldsymbol{s}, \boldsymbol{c}, t) \end{cases} \tag{1}$$

where:

- $\boldsymbol{\mathcal{H}}$ is the actual simulator code that describes how $\boldsymbol{\theta}$ evolves in time
- $\boldsymbol{\mathcal{C}}$ is the operator which describes how $\boldsymbol{c}$ evolves in time , i.e., the status of components and systems at each time step
- $\boldsymbol{s}$ is the set of stochastic parameters.

Starting from the system located in an initial state, $\boldsymbol{\theta}(t = 0) = \boldsymbol{\theta}(0)$**,** and the set of stochastic parameters (which are generally generated through a stochastic sampling process), the simulator determine at each time step the temporal evolution of $\boldsymbol{\theta}(t)$. At the same time, the system control logic[1] determines the status of the system and components $\boldsymbol{c}(t)$. The coupling between these two sets of variables is shown in Figure 3.

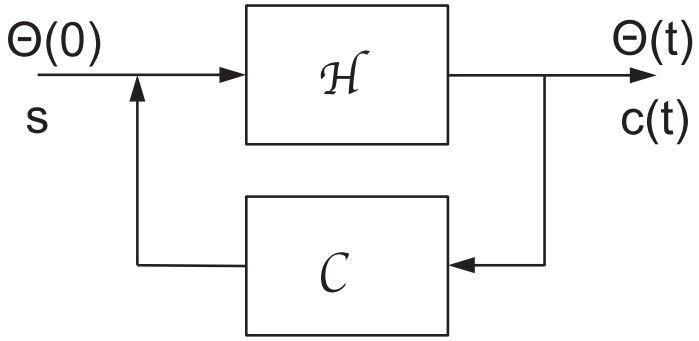

Figure 3. Relationship between simulator phisics code (*H*) and control logic (*C*).

By using the RISMC approach, the PRA analysis is performed by [3]:

1. Associating a probabilistic distribution function (pdf) to the set of parameters $\boldsymbol{s}$ (e.g., timing of events)
2. Performing stochastic sampling of the pdfs defined in Step 1
3. Performing a simulation run given $\boldsymbol{s}$ sampled in Step 2, i.e., solve the system of equations (1)
4. Repeating Steps 2 and 3 *M* times and evaluating user defined stochastic parameters such as core damage (CD) probability (P_{CD}).

[1] Which is usually integral part of the system simulator

2.1 RISMC toolkit

In order to perform advanced safety analysis, the RISMC Pathway has a toolkit that was developed at INL using MOOSE [11] as the underlying numerical solver framework. This toolkit consists of the following software tools (see Figure 4):

- RELAP-7 [9]: the code responsible for simulating the thermal-hydraulic dynamics of the plant.
- RAVEN [10] (see Section 2.2): it has two main functions: 1) act as a controller of the RELAP-7 simulation and 2) generate multiple scenarios (i.e., a sampler) by stochastically changing the order and/or timing of events.
- PEACOCK: the Graphical User Interface (GUI) that allows the user to create/modify input files of both RAVEN and RELAP-7. Also, it monitors the simulation in real time while it is running.
- GRIZZLY [12]: the code that simulates the thermal-mechanical behavior of components in order to model component aging and degradation. Note that for the analysis described in this report, aging was not considered.

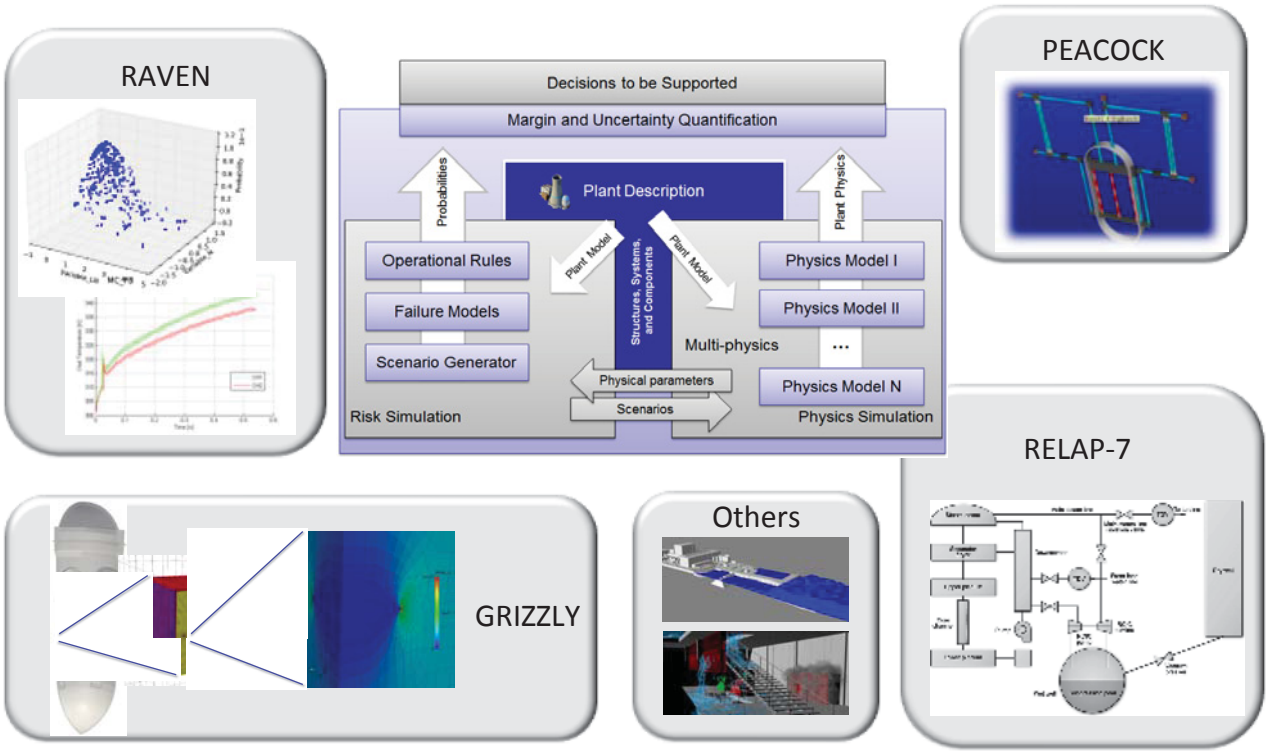

Figure 4. Overview of the RISMC toolkit.

2.2 RAVEN Statistical framework

RAVEN (Risk Analysis and Virtual control ENviroment) is a software framework that allows the user to perform generic statistical analysis. By statistical analysis we include:

- Sampling of codes: either stochastic (e.g., Monte-Carlo [13] and Latin Hypercube Sampling [14]) or deterministic (e.g., grid and Dynamic Event Tree [15])

- Generation of Reduced Order Models [16] also known as Surrogate models
- Post-processing of the sampled data and generation of statistical parameters (e.g., mean, variance, covariance matrix)

Figure 5 shows a general overview of the elements that comprise the RAVEN statistical framework:

- Model: it represents the pipeline between input and output space. It comprises both codes (e.g., RELAP-7) and also Reduced Order Models
- Sampler: it is the driver for any specific sampling strategy (e.g., Monte-Carlo, LHS, DET)
- Database: the data storing entity
- Post-processing module: module that performs statistical analyses and visualizes results

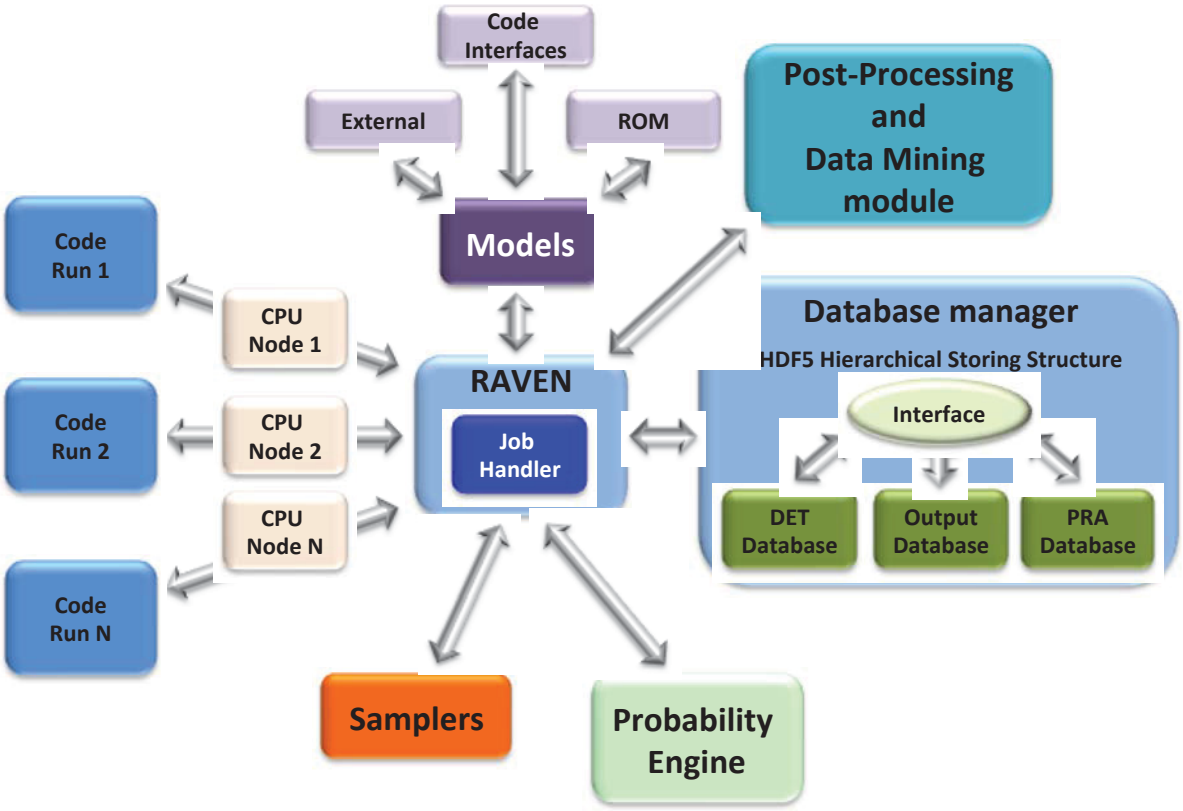

Figure 5. Scheme of RAVEN statistical framework components.

2.2.1 CROW Library

CROW is an INL developed C++ library which contains the full set of probabilistic functions which are used by RAVEN to perform any kind of statistic analysis. It contains the following modules:

- Interface with BOOST [17]. BOOST[2] is a set of libraries for the C++ programming language that provide support for tasks and structures such as linear algebra, pseudorandom number generation, multithreading, image processing, regular expressions, and unit testing. For our applications we are using the mathematical/statistical[3] library of BOOST which contains a wide selection of univariate statistical distributions and functions that operate on them (pdf and cdf calculation along with random number generation).

- Multi-variate distributions. This module contains the same functions mentioned above (pdf and cdf calculation along with random number generation) but applied to multi-variate distributions. Not only multi-variate normal distributions are modeled but also generic multi-variate distributions defined over a set of samples scattered or grid distributed.

[2] http://www.boost.org/

[3] http://www.boost.org/doc/libs/1_58_0/libs/math/doc/html/dist.html

3. DATA MINING

Data mining is a fairly generic concept that entails the generation of information/knowledge from data sets. The process of generation of information/knowledge can be performed in various ways depending on the type of application but it possible to classify all data analysis approaches into four categories:

- Reduced Order Modeling (ROM): ROM algorithms aim to reduce to the complexity of the data by finding a mathematical objects that emulate the behavior of the data by learning its input/output relations and reconstructing such relations through a regression/interpolation based approach
- Dimensionality reduction: this category includes all methods than aim to reduce the dimensionality of the data set and project the original data into a reduced space
- Clustering: algorithms in this category partition the data based on a set of defined similarity measures
- Data searching: identify the closest data point in a database given a reference point. Once that point has been found it is possible to infer properties of the reference point

In [5] we focused the development directly on the first two categories using RAVEN and we showed several applications that are of interest in the RISMC project. This report focuses on the third step of the RISMC method (see Figure 6): data clustering post-processing (in particular time-dependent data mining) and data visualization.

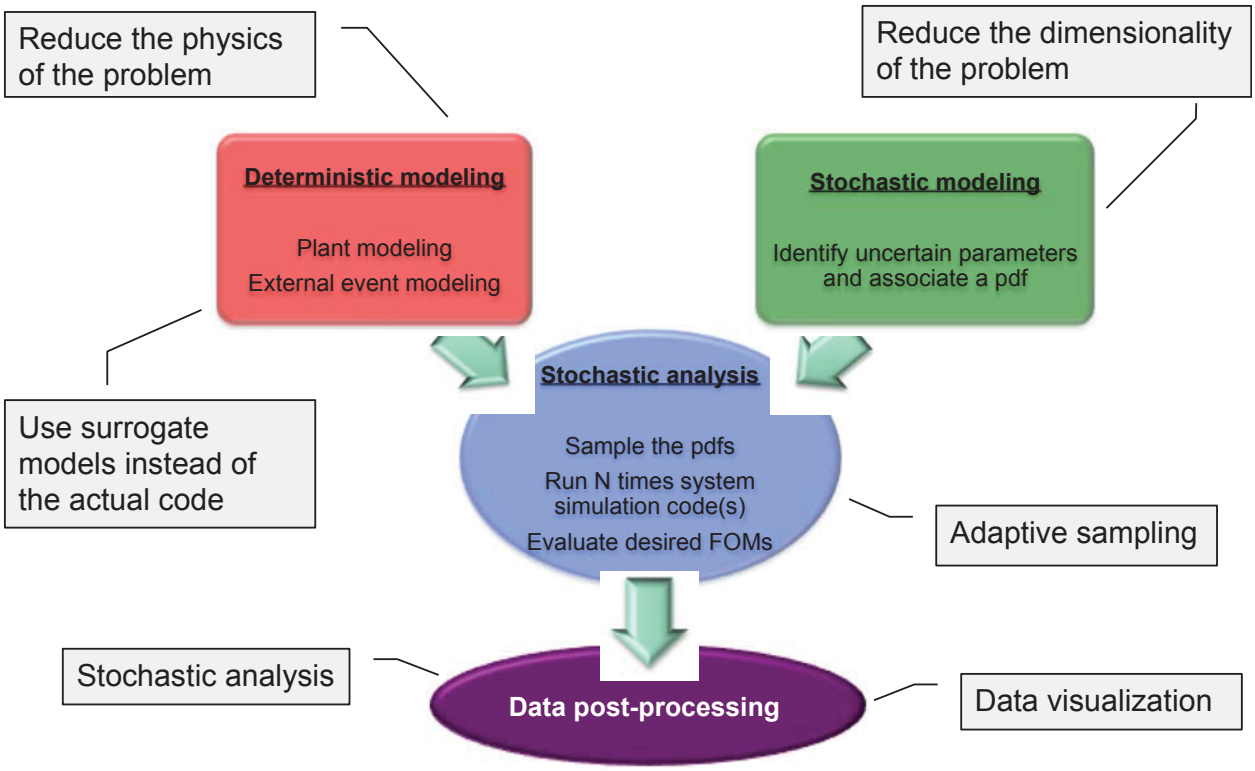

Figure 6. Overview of how data analysis techniques can be applied to the 4 main steps of a typical RISMC analysis.

Reference [18] shows the development of the basic data mining capabilities. These capabilities were introduced in RAVEN using the scikit-learn[4] library (often called sklearn). Scikit-learn is open source software machine-learning library for the Python. It includes classification, regression and clustering algorithms including support vector machines, Gaussian process models, PCA, K-Means and Mean-Shift, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Since [18] focused on static (i.e., non time-dependent data), during the FY16 the RAVEN development focused on the time-dependent data-mining. This development moved in two parallel directions:

1. Extension of existing data-analysis methods by performing them at specific time instants: the time series are sampled at specific time instants and data-mining algorithms are performed on the data set (which is static in nature) specific at each specific time instant

2. Development of new algorithms and extension of existing clustering methods to analyze time series as a whole: each time series is considered as a whole data point and the resulting data mining approach takes care of the full temporal profile of the time histories.

While the results obtained in the first direction are shown in [19], the second path is described in detail in this report.

The range of applications that are of interest from an engineering point of view (and, hence, not limited to a typical RISMC application) are the following:

- *Analysis of data input-output relationship*: this is the most relevant application within the RISMC project, i.e., the identification of the connection between input parameters (which are stochastically sampled) and the full temporal profile of the simulation. Using the system of equation (1), we want to create a correlation between $\boldsymbol{s}$ and $\boldsymbol{\theta}(t)$. While it is trivial to make a connection between input sampled parameters and static output parameters, such as max clad temperature or average fluid speed, it gets tougher when we want to consider the full temporal profile of the simulation

- *Outlier identification*: due to numerical limitations of system simulator codes (e.g., range of validity for correlations), the outcome of a subset of simulation runs may contain wrong results. When these limitations are reached, the temporal profile may contain anomalous behaviors. The objective of data mining would be to identify these anomalous behaviors, i.e. outliers. This would dramatically increase the quality of the data being generated/analyzed

- *Diagnosis/prognosis*: while this application is not referenced in detail in this work, it is worth highlighting that all data mining methods presented in this work can be used in an operating environment to assist reactor operators to assess plant conditions and possible plant evolutions during an accident scenario. The basic idea would be use a large set of simulations generated for different accident conditions as a database with the objective to constantly match the actual plant status. Once a match has been established (i.e., a set of simulated scenarios match closely enough[5] the actual and past plant status) the operators can now:

    - identify the possible causes that is generating the abnormal behavior in the plant (diagnosis)

    - predict possible future evolutions of the accident scenarios (prognosis)

    Thus at this point the operators have a tool that can provide risk-informed insights regarding task management and resources prioritization during accident scenarios.

In this respect the following major capabilities have been added to the actual RAVEN code (see Figure 7):

[4] http://scikit-learn.org/stable/

[5] This matching can be analytically performed using distance metrics that are described in Section 6.3

- *Time dependent metrics*: this block features a set of similarity measures (i.e., distance metrics) for time dependent data such as Euclidean distance and Dynamic Time Warping [20]. If needed, the user can uses RAVEN APIs to develop their own custom distance metrics

- *Interfaced post-processor*: this class of custom made post-processors allowed the user to develop its own multi-purpose generic processors (through a set of APIs) that can be directly employed internally in RAVEN. Several of these post-processors have been developed to manipulate time-dependent data

- *Pre-Processing methods*: with this feature the user can perform data pre-processing of the time-dependent data prior data mining

- *Agglomerative clustering*: agglomerative clustering is the basic hierarchical clustering available within sklearn

- *SciPy hierarchical*: in parallel to sklearn we have developed an interface with clustering algorithms available in SciPy. In particular we were interested into the hierarchical clustering algorithms here available. Compared to the sklearn version of the algorithm (see item above), the SciPy version allows the user to visualize and explore the hierarchical structure of the data: the dendrogram.

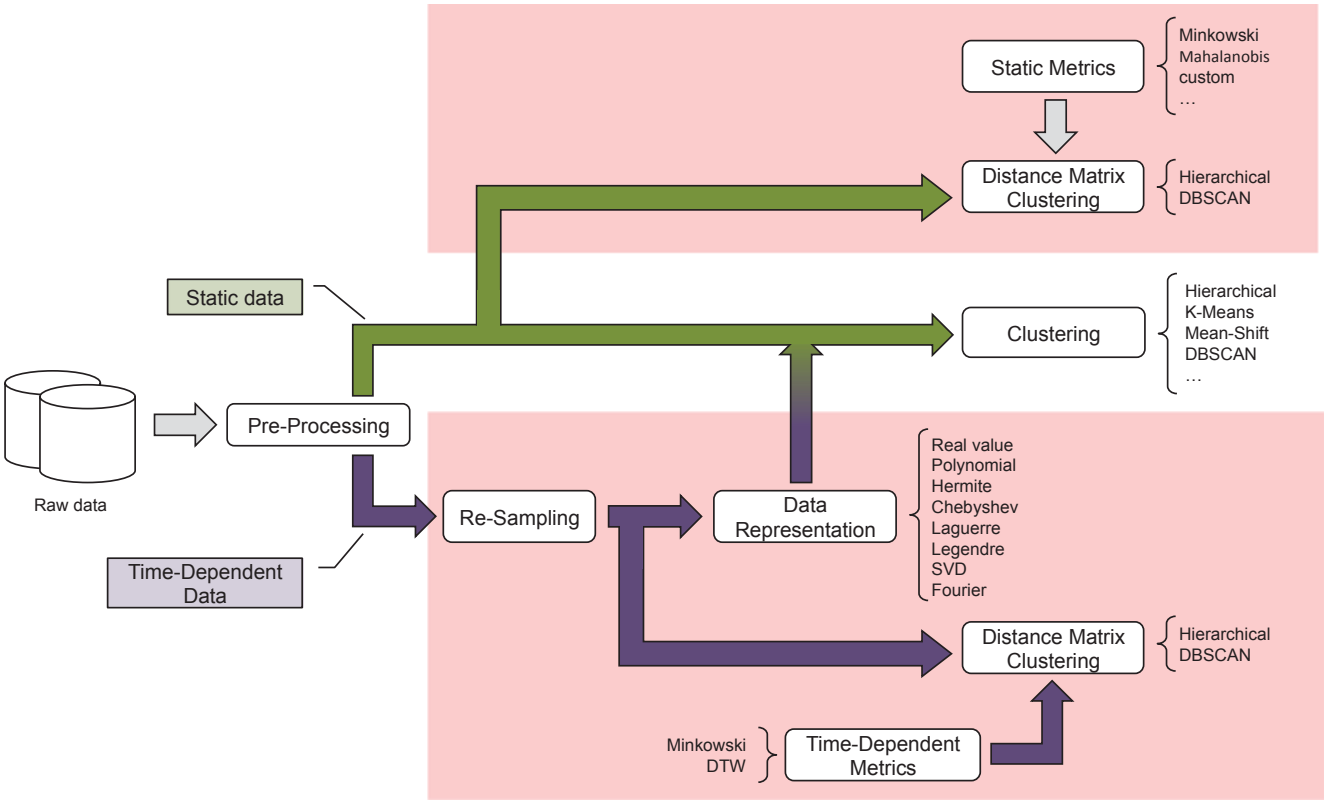

Figure 7. Data mining capabilities in RAVEN: portion added in FY16 and presented in this report is highlighted in red.

4. DATA MINING FRAMEWORK

In order to maintain the same standard throughout the report, we introduce here the set of mathematical symbols and terminology. The types of data that are considered in this report are exclusively numerical data[6], i.e. data which consists of numbers of any format (integer or double). Given this, we can represent a generic data set Ξ as a collection (i.e., a set) of N objects:

$$\Xi = \{H_1, H_2, \dots, H_N\} \tag{2}$$

Each object H_n $(n = 1, \dots, N)$ lies in a multi-dimensional space $\mathcal{H}$: $H_n \in \mathcal{H}$ and $\Xi \subset \mathcal{H}$. From now on we will consider $\mathcal{H}$ as a metric space [21]: $\mathcal{H}$is a space where it is possible to define an arbitrary distance function, i.e., a metric (see Section 4.1).

Since we are dealing with numerical data in a multi-dimensional space we can say that $\mathcal{H} \subset \mathbb{R}^D$ where D is the dimensionality of the space:

$$H_n = [H_n^1, \dots, H_n^d, \dots, H_n^D] \tag{3}$$

We will indicate with x^d $(d = 1, \dots, D)$ each coordinate d of $\mathcal{H} \subset \mathbb{R}^D$. Thus, each element H_n^d of H_n is the d coordinate, i.e. dimension x^d, $(d = 1, \dots, D)$ of the object H_n.

In the next sections we will describe in detail the basic concepts that are part of any type of data-mining activity.

4.1 Metrics

The similarity criterion is distance. Two or more objects belong to the same cluster if they are "close" according to a specified distance. The approach of using distance metrics to clustering is called distance-based clustering and it is used in this work. The notion of distance implies that the data points lay in a metric space [21]:

Definition, Metric Space: metric space is a space X provided with a function $\delta(.,.)$: $\mathcal{H} \times \mathcal{H} \to \mathbb{R}$ satisfying the following properties $\forall\ H_1, H_2 \in \mathcal{H}$:

- $\delta(H_1, H_2) \geq 0$
- $\delta(H_1, H_2) = d(H_2, H_1)$
- $\delta(H_1, H_2) \leq \delta(H_1, H_3) + \delta(H_3, H_2) \quad \forall\ H_3 \in \mathcal{H}$

The function $\delta(H_1, H_2) \geq$ is usually called the distance function. In a 2-dimensional Euclidean space ($\mathbb{R}^2$), the distance between points can be calculated using the Pythagorean theorem which is the direct application of the Euclidean distance $\delta_2(H_1, H_2)$ and it is a special case of the most general Minkowski distance:

$$\delta_2(H_1, H_2) = \sqrt{(H_1^1 - H_2^1)^2 + (H_1^2 - H_2^2)^2} \tag{4}$$

between two points $H_1 = [H_1^1, H_1^2]$ and $H_2 = [H_2^1, H_2^2]$ in $\mathbb{R}^2$.

In the literature [22], it is possible to find several types of distances other than the Euclidean and the Minkowski distance as shown in Table 1. The approach of using distance metrics is called distance-based

[6] As opposed to symbolic data, i.e., data which contains symbols (e.g., letters)

clustering and will be used in this work.

Table 1. Summary of the commonly used dissimilarity measures[7]

Measure	Form
Minkowski	$\delta_n(H_1, H_2) = \sqrt[n]{\sum_{k=1}^{K} \left\|H_1^k - H_2^k\right\|^n}$
Euclidean	$\delta_2(H_1, H_2) = \sqrt{\sum_{k=1}^{K} \left\|H_1^k - H_2^k\right\|^2}$
Taxicab	$\delta_1(H_1, H_2) = \sum_{k=1}^{K} \left\|H_1^k - H_2^k\right\|$
Supremum	$\delta_0(H_1, H_2) = \max_k \left\|H_1^k - H_2^k\right\|$
Mahalanobis[8]	$\delta_M(H_1, H_2) = (H_1 - H_2)^T S^{-1} (H_1 - H_2)$

4.2 Data Pre-Processing

The raw data simulated or gathered by any monitoring device often requires manipulation depending on the type of application. This type manipulation can involve the following operations:

- *Variable selection*. This step consists of the identification of the variables that are considered useful/relevant for the characterization of the data. For a transient in a nuclear plant, these variables may be both process variables such as temperature, pressure or level in specific points of the system and hardware/software/firmware states
- *Data representation*. In this step the user decides on how to structure the information content of each data point, i.e., it defines the "real-world" information content for each element $H_{n,d}$ (e.g., for transients of nuclear power plants an element $H_{n,d}$ could represent the maximum clad temperature of the hottest channel in the RELAP5 nodalization). This step and the previous one specify the dimensionality D of the data space X
- *Data normalization*. This steps address the issue that dimensions of the data space X are characterized by different scales. If this is the case, dimension with higher scales can bias the analysis if the chosen metric provide same importance to each dimension of X. Two classical methods are used to perform data normalization:
    - Feature Scaling Normalization: this normalization brings all values $H_{i,j}$ in the range $[0,1]$ interval

[7] The measures in this table refers to two D-dimensional vectors $H_1 = [H_{1,1}, \dots, H_{1,d}, \dots, H_{1,D}]$ and $H_2 = [H_{2,1}, \dots, H_{2,d}, \dots, H_{2,D}]$

[8] S refers to the covariance matrix. This implies that the covariance matrix needs to be computed prior determining $\delta_M(H_1, H_2)$. This can causes problems, from a computational efficiency point of view, if the data set changes (i.e., data points are added or removed)

$$H_{i,j}^{norm} = \frac{H_{i,j} - \min_j(H_{i,j})}{\max_j(H_{i,j}) - \min_j(H_{i,j})} \tag{5}$$

- Z-Normalization: this normalization (also known as standard score normalization) transforms the data to have zero mean and unit variance:

$$H_{i,j}^{norm} = \frac{H_{i,j} - \underset{j}{\text{mean}}(H_{i,j})}{\underset{j}{\text{stdDev}}(H_{i,j})} \tag{6}$$

where the operators $\underset{j}{\text{mean}}(.)$ and $\underset{j}{\text{stdDev}}(.)$ calculate the mean and the standard deviation along dimension j respectively.

4.3 Clustering

From a mathematical viewpoint, given a data set Ξ, clustering [23] aims to find a partition $\boldsymbol{C} = \{C_1, \dots, C_l, \dots, C_L\}$ of Ξ where each C_l $(l = 1, \dots, L)$ is called a cluster. The partition $\boldsymbol{C}$ is such that:

$$\begin{cases} C_l \neq \emptyset \ \ \forall l = 1, \dots, L \\ \bigcup_{l=1}^{L} C_l = \Xi \end{cases} \tag{7}$$

Even though the number of clustering algorithms available in the literature is large, usually the most used ones when applied to time series are the following: Hierarchical [24], K-Means [25] and Mean-shift [26].

Hierarchical algorithms build a hierarchical tree from the individual points (leaves) by progressively merging them into clusters until all points are inside a single cluster (root). Clustering algorithms such as K-Means and Mean-Shift, on the other hand, seek a single partition of the data sets instead of a nested sequence of partitions obtained by hierarchical methodologies.

4.4 Algorithms

In this section we present in more detail a portion of the clustering algorithms that have been here employed:

- Hierarchical (see Section 4.4.1)
- K-Means (see Section 4.4.2)
- Mean-Shift (see Section 4.4.3)
- DBSCAN (see Section 4.4.4)

4.4.1 Hierarchical

The hierarchical clustering is, along with the K-Means one (see Section 4.3.1), the most basic clustering algorithms available in literature. Hierarchical algorithms organize data into a structure according to a proximity matrix in which each element (j, k) is some measure of the similarity (or distance) between the items to which row j and column k correspond. Usually, the final result of these algorithms is a binary tree, also called *dendrogram*, in which the root of the tree represents the whole data set and each leaf is a data point.

To show how hierarchical clustering works we will employ the data set pictured in Figure 8, i.e., a data set containing about one hundred points in a 2-dimensional space.

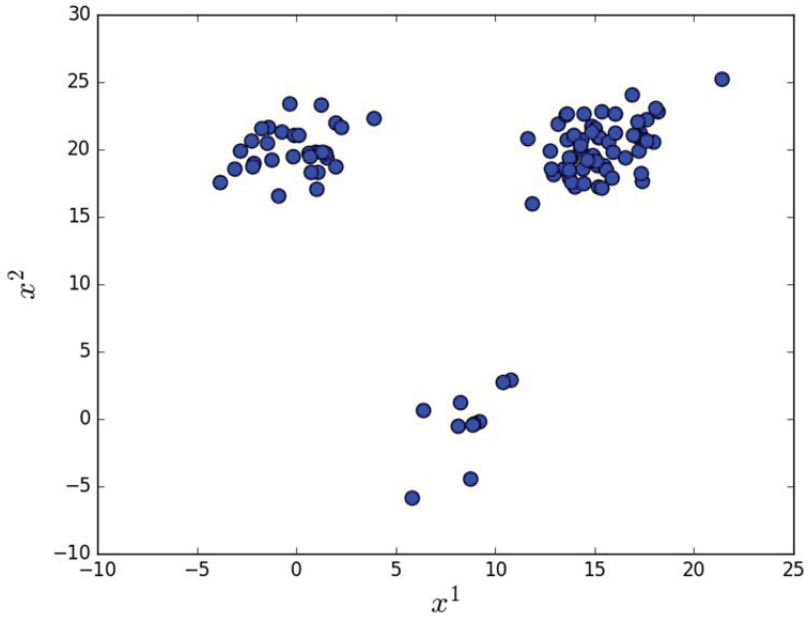

Figure 8. Example of static data scatter plot.

Given a set of N items to be clustered, and a $N \times N$ distance (or dissimilarity) matrix, the basic steps behind hierarchical clustering are the following:

1. Assign each data point to a cluster, i.e., from N data points, N clusters are initialized, each cluster contains just one data point. The distances among each cluster is the same as the distances between the data points they contain
2. Determine the closest pair of clusters and merge them into a single cluster
3. Determine the distances between the new cluster (obtained in Step 2) and the remaining clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N

Step 3 can be done in different ways, which is what distinguishes single-linkage from complete-linkage and average-linkage clustering.

- S*ingle-linkage clustering* (also called the connectedness or minimum method), we consider the distance between one cluster and another cluster to be equal to the shortest distance from any member of one cluster to any member of the other cluster. If the data consist of similarities, we consider the similarity between one cluster and another cluster to be equal to the greatest similarity from any member of one cluster to any member of the other cluster.
- *Complete-linkage clustering* (also called the diameter or maximum method), we consider the distance between one cluster and another cluster to be equal to the greatest distance from any member of one cluster to any member of the other cluster.
- *Average-linkage clustering*, we consider the distance between one cluster and another cluster to be equal to the average distance from any member of one cluster to any member of the other cluster.

The $N \times N$ distance matrix is $D = [d(i,j)]$. The clusterings are assigned sequence numbers $0,1,\ldots,n-1$ and $L(k)$ is the level of the kth clustering. A cluster with sequence number m is denoted (m) and the proximity between clusters (r) and (s) is denoted $d[(r),(s)]$.

The algorithm is composed of the following steps:

1. Begin with the disjoint clustering having level $L(0) = 0$ and sequence number $m = 0$.
2. Find the least dissimilar pair of clusters in the current clustering, say pair (r), (s), according to

$$d[(r),(s)] = min\, d[(i),(j)] \tag{8}$$

   where the minimum is over all pairs of clusters in the current clustering.

3. Increment the sequence number: $m = m + 1$. Merge clusters (r) and (s) into a single cluster to form the next clustering m. Set the level of this clustering to

$$L(m) = d[(r),(s)]$$

4. Update the proximity matrix, D, by deleting the rows and columns corresponding to clusters (r) and (s) and adding a row and column corresponding to the newly formed cluster. The proximity between the new cluster, denoted (r,s) and old cluster (k) is defined in this way:

$$d[(k),(r,s)] = min\, d[(k),(r)], d[(k),(s)] \tag{9}$$

5. If all objects are in one cluster, stop. Else, go to step 2.

Figure 9 shows the dendrogram of the data points of Figure 8 while Figure 10 is a plot identical to Figure 9 where data points are colored based on the label associated to the chosen threshold level (i.e., 22). Note that the dendrogram clearly indicates three clusters.

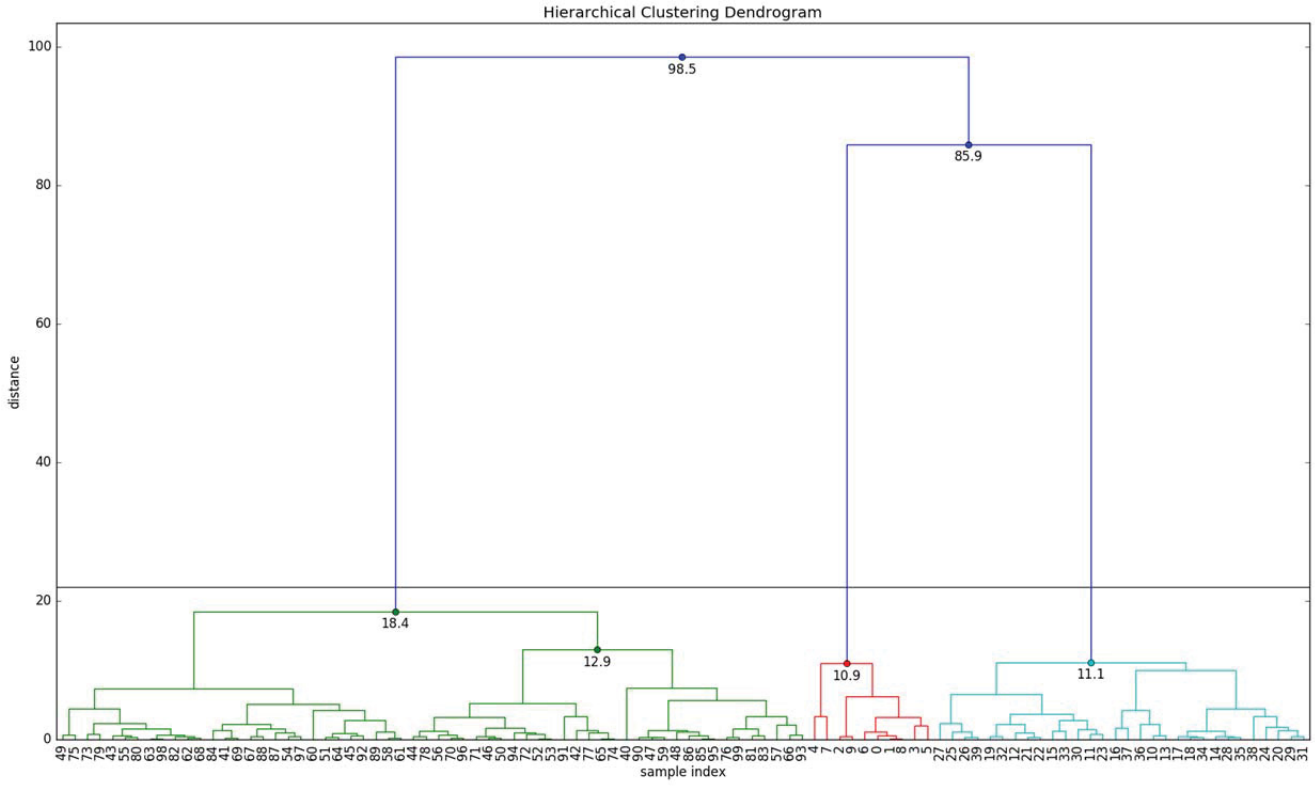

Figure 9. Dendrogram obtained by RAVEN for the data set shown in Figure 8.

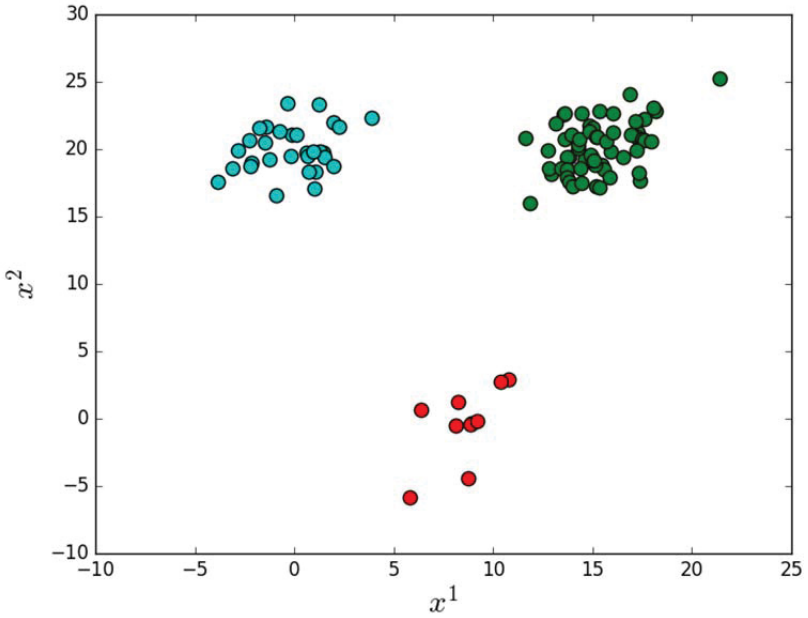

Figure 10. Scatter plot of the data set shown in Figure 8 colored by the label values obtained by the hierrachical algorithm (see Figure 10).

4.4.2 K-Means

The K-Means algorithms [25] is probably the most simple clustering algorithm. The basic idea is to determine K centers (K is provided as an input variable) $\varkappa_k$ $(k = 1, \dots, K)$: one center for each cluster. At the beginning these centers are placed randomly in the feature space. The next step is to take each point of the data set and associate it to the nearest center $\varkappa_k$. At this point the centers $\varkappa_k$ are re-calculated as average of the data points associated to $\varkappa_k$. This data points and centers average step is repeated. As a result of this loop, the K centers change their location step by step until no more changes are done or, in other words, centers do not move any more.

It can be shown that this algorithm aims to minimize the squared error function $\mathcal{L}$:

$$\mathcal{L} = \sum_{k=1}^{K} \sum_{n=1}^{N} \left(\delta_2(H_n, \varkappa_k)\right)^2 \tag{10}$$

where:

- N and K refers to the cardinality of the data set and of the cluster centers respectively
- $\delta_2(H_n, \varkappa_k)$ refers to the Euclidean distance between H_n and $\varkappa_k$

In mode details, given a dataset $\Xi = \{H_n\}$ $(n = 1, \dots, N)$ and the desired number of clusters K, the algorithm works as follows:

1. Randomly select K cluster centers
2. Determine $\delta_2(H_n, \varkappa_k)$ $\forall\, n = 1, \dots, N$ and $\forall\, k = 1, \dots, K$
3. Assign each data point H_n to the closest cluster center
4. Calculate the new set of cluster centers as:

$$\varkappa_k = \frac{1}{N_k}\sum_{n=1}^{N_k} H_n \tag{11}$$

where the sum is performed over all N_k data points H_n associated to the cluster K ($\sum_{k=1}^{K} N_k = N$).

5. Repeat Steps 2 through 4 until the data points to cluster association does not change

4.4.3 Mean-Shift

Mode-seeking approaches [27] look at the density distribution of data points lying in a metric space. Clusters are viewed as regions of the space with high point density separated by regions of low point density. Clusters can be identified by searching for regions of high density, called modes. For the comparison a Mode-seeking algorithm referred to as the Mean-Shift.

The Mean-Shift algorithm [26] is a non-parametric iterative procedure that can be used to assign each point to one cluster center through a set of local averaging operations. The local averaging operations provide empirical cluster centers within the locality and define the vector which denotes the direction of increase for the underlying unknown density function.

The underlying idea is to treat each point x_i $(i = 1, \dots, N)$ of the dataset as an empirical probability distribution function using kernel $K(x)$: $\mathbb{R}^D \to \mathbb{R}$. This multivariate kernel density resides in a multidimensional space where regions with high data density (i.e., modes) correspond to local maxima of the density estimate $f_I(x)$ defined by:

$$f_I(x) = \frac{1}{N\,BW^d}\sum_{i=1}^{N} K\left(\frac{x - x_i}{BW}\right) \tag{12}$$

where $x \in \mathbb{R}^D$ and BW is often referred as the bandwidth associated with the kernel. The kernel K in Eq. 12 serves as a weighting function associated with each data point and is expressed as:

$$K(x) = c_k\, k(\|x\|^2) \tag{13}$$

where $k(x)$: $[0, \infty] \to \mathbb{R}$ is referred as the kernel profile and c_k is a normalization constant. The profile satisfies the following properties:

- $k(x)$ is non negative
- $k(x)$ is non increasing
- $k(x)$ is piecewise continuous and $\int_0^\infty k(x)dx < \infty$

In order to estimate the data points with highest probability from an initial estimate (i.e., the modes of $f_I(x)$), consider the gradient of the density function $\nabla f_I(x) = 0$ where:

$$\nabla f_I(x) = \frac{2c_k}{NBW^{D+2}}\left(\sum_{i=1}^{N} g\left(\left\|\frac{x - x_i}{BW}\right\|^2\right)\right)\left(\frac{\sum_{i=1}^{N} x\, g\left(\left\|\frac{x - x_i}{BW}\right\|^2\right)}{\sum_{i=1}^{N} g\left(\left\|\frac{x - x_i}{BW}\right\|^2\right)} - x\right) \tag{14}$$

which points in the direction of the increase in kernel density estimate. The kernel $K(x)$ is also referred to as the shadow of $G(x) = c_g\, g(\|x\|^2)$ where c_g, similar to c_k, is a normalization constant and $g(x)$ is the derivative of $k(x)$ over x, i.e., $g(x) = k'(x)$.

In Equation 14 the first term denoted is a scalar proportional to the density estimate computed with the kernel $G(x)$ and does not provide information regarding where the mode resides. Unlike the first term, the second term in Equation 14, is difference between the weighted mean

$$m(x) = \frac{\sum_{i=1}^{N} x\, g\left(\left\|\frac{x - x_i}{BW}\right\|^2\right)}{\sum_{i=1}^{N} g\left(\left\|\frac{x - x_i}{BW}\right\|^2\right)} \tag{15}$$

and the initial estimate x. This term points in the direction of local increase in density using kernel $G(x)$, hence provides a means to find the mode of the density. Note that all points used to compute a particular mode are considered to reside in the same cluster.

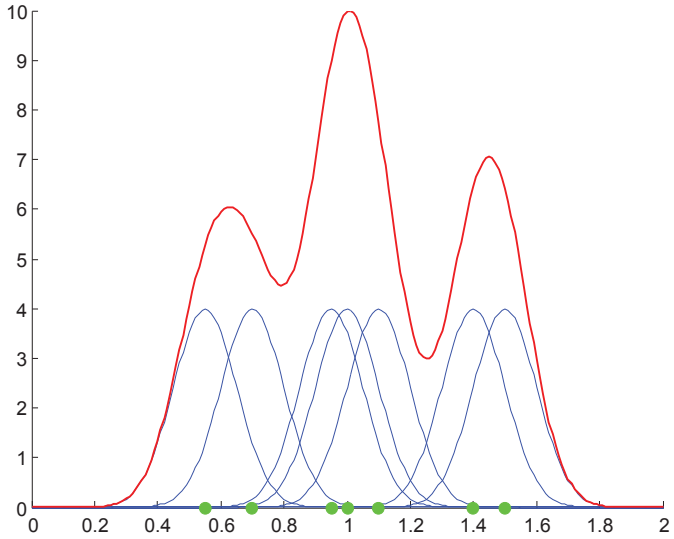

Figure 11. Density estimation (red line) for points distributed in a 1D space (green dots) given kernel functions associated to each point (blue lines).

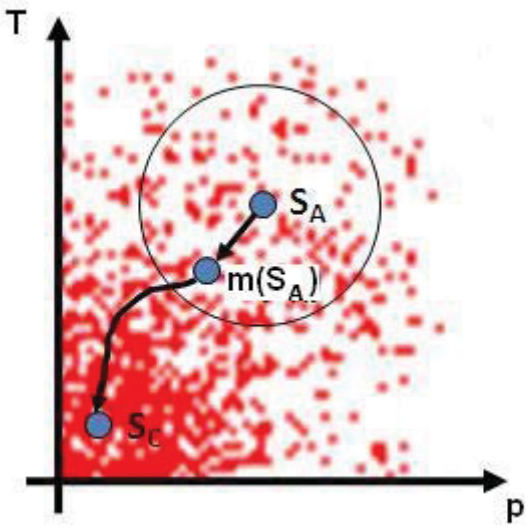

Figure 12. Cluster center (S_c) estimation using gradient based approach.

Since each data point x_i is considered as an empirical probability distribution function (see Figure 11), this consideration allows to include in the scenario clustering analysis also the possible uncertainty associated with each scenario.

Starting from an arbitrary data point (e.g., point S_A in Figure 12), the algorithm defines the locality using a circular region centered around this point (or a hyper-spherical region depending on the number of dimensions). The radius of this region is equal to the bandwidth BW of the chosen kernel. The objective is to consider points residing in this region for estimating their weighted average which corresponds to the center of mass of these points (point $m(S_A)$) in Figure 12) where:

$$m(S_A) = \frac{\sum_{i=1}^{N} x\, g\left(\left\|\frac{S_A - x_i}{BW}\right\|^2\right)}{\sum_{i=1}^{N} g\left(\left\|\frac{S_A - x_i}{BW}\right\|^2\right)} \tag{16}$$

The weighted average or the center of mass $m(S_A)$ is used as the new position for S_A for the next iteration, such that the density in the new center of mass is always higher than its previous position. Convergence is reached when the distance between the new center of mass and the old one is below a fixed threshold[9] (point SC in Figure 12). Upon reaching this stopping condition:

- point S_C is considered the center of a cluster, and,
- the original point S_A is uniquely associated to the cluster centered by point S_C.

During this process, several derivative kernels $G(x)$ can be used as indicated in [28], including:

- Uniform kernel: $G(x) = \begin{cases} 1 & \|x\| \leq BW \\ 0 & \|x\| > BW \end{cases}$
- Gaussian kernel: $G(x) = e^{-\frac{\|x\|^2}{BW^2}}$
- Epanechnikov kernel: $G(x) = \begin{cases} \frac{3}{4}(1 - x^2) & \|x\| \leq BW \\ 0 & \|x\| > BW \end{cases}$

As indicated earlier, the purpose of $G(x)$ is to assign different weights to different points during the estimation of the center of mass. Kernels such as the Gauss and the Cone kernel assign higher weights to the points located at the center of kernel which implies that the calculated center of mass is biased toward the center of the kernel. Thus, the distance between two consecutive calculated positions of the center of mass is smaller if Gauss or Cone kernels are used compared to the uniform kernel. Consequently, convergence can be reached faster using kernels such as the Gauss one.

When this procedure is repeated for all the points in the data set it is possible to obtain:

- The center of all the clusters and the list of all the points that belong to that specific cluster, and,
- The cluster to which each point belongs (as mentioned earlier, each point belongs to one cluster only).

[9] Usually the threshold is a fixed value chosen to be a small fraction of h (typically $h/100$ or $h/1000$)

4.4.4 DBSCAN

DBSCAN is a density-based clustering algorithm not dissimilar from the Mean-Shift one described in Section 4.4.3. Given a set of data points, the algorithm clusters together data points located in high density regions while marking as outliers points that lie alone in low density regions.

The DBSCAN algorithm require two parameters:

- BW: bandwidth
- MinPts: minimum number of points required to form a high density region

The algorithm starts with an arbitrary starting data point that has not been previously visited. Then the algorithm retrieves all data points within the bandwidth radius: if there are sufficiently many points (i.e., MinPts) a cluster is started otherwise the point is labeled as noise.

If a data point is found to be a dense part of a cluster, its BW-neighborhood is also part of that cluster. Hence, all points that are found within the BW-neighborhood are added, as is their own BW-neighborhood when they are also dense. This process continues until the density-connected cluster is completely found. Then, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise.

5. TIME DEPENDENT DATA ANALYSIS

So far we have briefly described the most relevant data mining techniques within the RISMC framework. At this point we can make step further: making the algorithms shown in Section 4 useful to deal with time dependent data. Using same analogy presented for static data we aim to group scenario with similar temporal patterns.

An example is shown in Figure 13 applied to a data set containing the time evolution of 1000 time series has been generated by randomly changing (through a Monte-Carlo sampling) three variables (i.e., x, y, z). We introduced a "discontinuity" in the temporal evolution of the time series depending if $x > 4$ or $x < 4$.

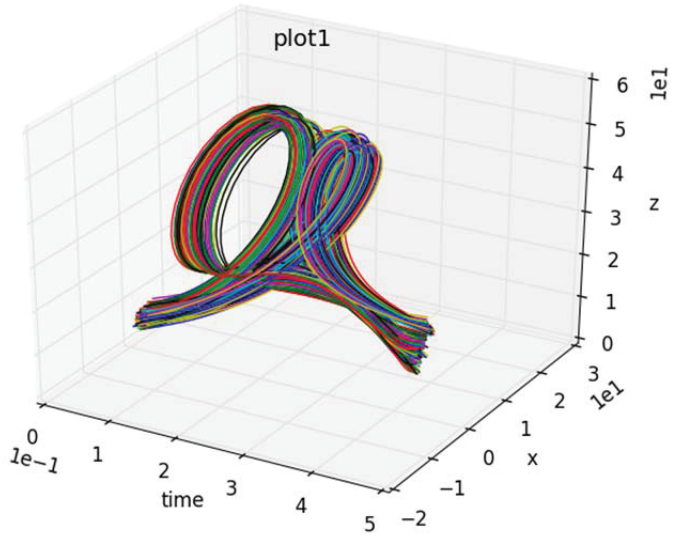

Figure 13. Plot of a 1000 time series data set in a 2-dimensional space (plus time).

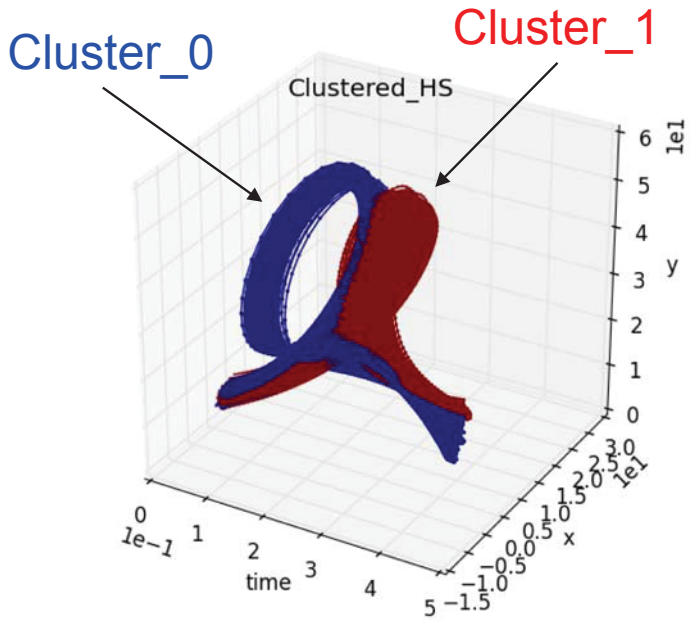

Figure 14. Plot of the clusters obtained from the data shown in Figure 13.

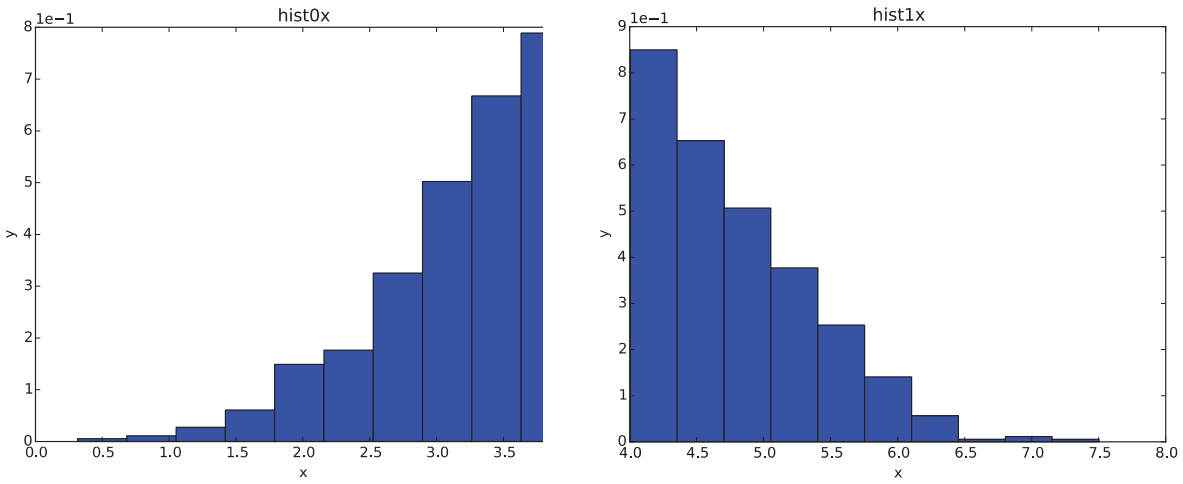

Figure 15. Histograms of the sampled values for Cluster_0 and Cluster_1 (shown in Figure 14) that created them and were captured by the clustering algorithm.

By using any clustering algorithm we want to partition the 1000 generated scenario into 2 clusters (see Figure 14). Note how the scenarios in each cluster have a very similar temporal behavior. Then, by looking at the histograms of the sampled variables x, y, z for the scenarios contained in each cluster we were able to verify that x was creating the splitting of the data set. Figure 15 shows the histograms of x for both clusters: for Cluster_0 $x < 4$ while $x > 4$ for Cluster_1. Note that we would not have been able to capture this "discontinuity" by considering only the end or max values of the time series.

5.1 Data Set format

Again, we will indicate with Λ the data set generate by any of the methods mentioned above which contain N time series[10] TS_{n} $(n = 1, \dots, N)$:

$$\Lambda = \{TS_1, \dots, TS_{\mathrm{n}}, \dots, TS_{\mathrm{N}}\} \tag{17}$$

To preserve generality, given the complexity of the data generated by simulation based PRA methods such as RISMC, we now assume that each scenario H_n contains three components:

$$TS_{\mathrm{n}} = \{\boldsymbol{\theta}_n, \boldsymbol{\Delta}_n, \boldsymbol{\Gamma}_n\} \tag{18}$$

These components are the following:

- Continuous data $\boldsymbol{\theta}_n$: this data contains the temporal evolution of each scenario, i.e., the time evolution of the M state variables x_m^n $(m = 1, \dots, M)$ (e.g., pressure and temperature at a specific computational node). All of these state variables change in time t (where t ranges[11] from 0 to t_n):

  $$\boldsymbol{\theta}_n = \{x_1^n, \dots, x_M^n\} \tag{19}$$

  where each x_m^n is a an array of values having length T_n. Hence, $\boldsymbol{\theta}_n$ can be viewed as a $M \times T_n$ matrix[12].

- Discrete data $\boldsymbol{\Delta}_n$: which contains timing of events. Note that a generic event E_i^n can occur:

[10] In this paper we will indicate time series as simulation runs or histories

[11] This allows us to maintain generality by having time series with different time lengths

[12] As an example, x_2^3 is a vector having length T_3 which represents the temporal profile of variable 2 for scenario 3.

- At a time instant τ_i: in this case the event can be defined as (E_i^n, τ_i), or,
- Over a time interval $[\tau_i^\alpha, \tau_i^\omega]$: in this case the event can be defined as $(E_i^n, [\tau_i^\alpha, \tau_i^\omega])$

- Set $\boldsymbol{\Gamma}_n$ of V boundary conditions BC_v^n $(v = 1, \dots, V)$ and U initial conditions IC_u^n $(u = 1, \dots, U)$.

This report focuses on the continuous part $\boldsymbol{\theta}_n$ and $\boldsymbol{\Gamma}_n$ of the data set Ξ.

5.2 Approaches

Two possible paths[13] that can be followed to analyze time dependent data:

1. *Path 1* (see Figure 16): Employ classical clustering algorithms by transforming each time series as a single multi-dimensional vector. Recall that algorithms such as K-Means and Mean-Shift can naturally deal with multi-dimensional vectors, i.e., each data point can be represented as a multi-dimensional vector. Following this, in this path each time series is converted into a multi-dimensional vector (as part of a pre-processing step). This can be done, for example, through a polynomial or Fourier transformation (see Section 6.2).
2. *Path 2* (see Figure 17): Maintain the original format of the time series end employ clustering algorithms that can receive in input a distance matrix (thus appropriate distance metrics needs to be defined). Few algorithms, such the hierarchical and the DBSCAN clustering algorithms can received in the input the solely distance matrix $\Delta = [\delta_{ij}]$ where each element δ_{ij} represent the distance between time series i and j.

The advantage of the first path is that it employs a large variety of clustering algorithms that can handle very complex data sets (i.e., data points clustered in complex shapes other than ellipsoidal). On the other side, the conversion of the time series prior the clustering may cause erroneous results if clustering parameters are not chosen properly (e.g., if the time series are very similar to each other and a coarse representation is chosen).

The second path, however, since it employs algorithms which can accent as input the distance matrix Δ, they do not require any data transformation. When dealing with time series data the most important parameter to be considered here is the distance metric chosen to determine each element δ_{ij} of Δ.

Both these methods are available in RAVEN and are described in more detail in Section 6.

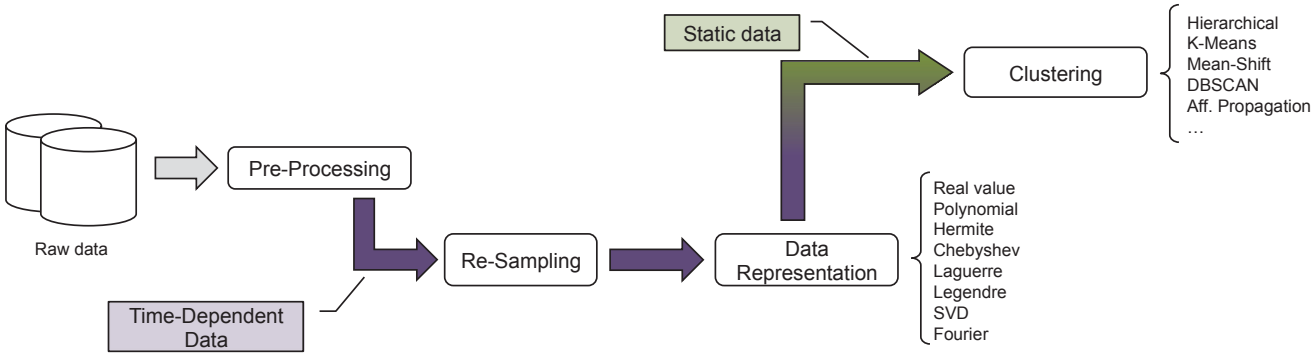

Figure 16. Analysis of time-dependent data using static representation conversion.

[13] A third path that is currently under investigation is to reconstruct the major clustering algorithms available in the literature (e.g., K-Means, Mean-Shift) so that they can natively perform data analysis on the time series data set (i.e., without prior transformation, see path 1). The major challenge in this approach is the need to define an operator that, given a subset of time series, it can generate a distance-based average time series. This average value can be challenging to obtain depending on the distance metrics employed.

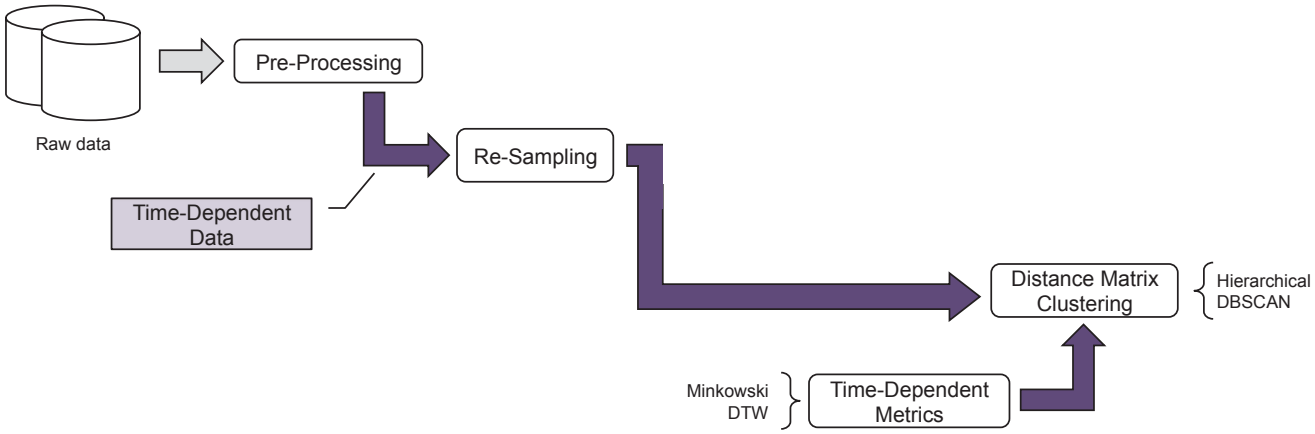

Figure 17. Analysis of time-dependent data using distance matrix based algorithms (e.g., Hierarchical).

6. RAVEN TIME DEPENDENT DATA ANALYSIS

6.1 Data Pre-processing

Depending on the application, the data set may need to be pre-processed. A common pre-processing method is the Z-normalization procedure: each variable x_m^n of $\boldsymbol{\theta}_n$ is transformed into $\hat{x}_m^n$:

$$\hat{x}_m^n = \frac{x_m^n - mean(x_m^n)}{stdDev(x_m^n)} \tag{20}$$

where $mean(x_m^n)$ and $stdDev(x_m^n)$ represent the mean and the standard deviation of x_m^n. This transformation provides an equal importance to every x_m^n and it compensates for amplitude offset and scaling effects when distance between time series is computed[14].

In case the time-series are affected by noise, it might be worthwhile to smooth the time series using classical filtering and regression techniques so that the noise is filtered out and the series information is maintained. A commonly used de-noising or filtering technique is the kernel-regression technique.

This simple technique starts from the raw data $\boldsymbol{\theta}_n$ which is time dependent (i.e., $\boldsymbol{\theta}_n(t)$) and generate the regressed term $\widetilde{\boldsymbol{\theta}}_n(t')$ as follows:

$$\widetilde{\boldsymbol{\theta}}_n(t') = \frac{\sum_{t=0}^{T_n} K(t-t')\,\boldsymbol{\theta}_n(t)}{K(t-t')} \tag{21}$$

where $K(t-t')$ is the kernel used to smooth $\boldsymbol{\theta}_n$.

Another operation that can be performed in the pre-processing is the re-sampling of $\boldsymbol{\theta}_n$. Recall that $\boldsymbol{\theta}_n$ contains the values of the time dependent data variables $\{x_1^n, \dots, x_M^n\}$ sampled at specific time instants. The re-sampling operation aim to reduce those time instants by choosing a new set of time instants (typically a smaller set) that preserves the information content of the $\boldsymbol{\theta}_n$.

The motivations behind the choice of this step are the following:

1. Less memory intensive
2. Faster computations

In RAVEN several re-sampling strategies have been implemented (see Figure 18):

- *Uniform*: N sample points (N is provided as input) are uniformly located along the time axis
- *First derivative*: N sample points (N is provided as input) concentrated in regions with higher values of the first derivative
- *Second derivative*: N sample points (N is provided as input) concentrated in regions with higher values of the second derivative
- *Filtered first derivative*: sample points are located when the first derivative is greater than a value provided as input; thus the obtained number of samples cannot be determined a priori
- *Filtered second derivative*: sample points are located when the second derivative is greater than a value provided as input; as indicated above, the obtained number of samples cannot be determined a priori

[14] This is in particular relevant when x_m have different scales (e.g., temperatures in the [500,2200] F interval while pressures are in the [0,16 10^6] Pa interval)

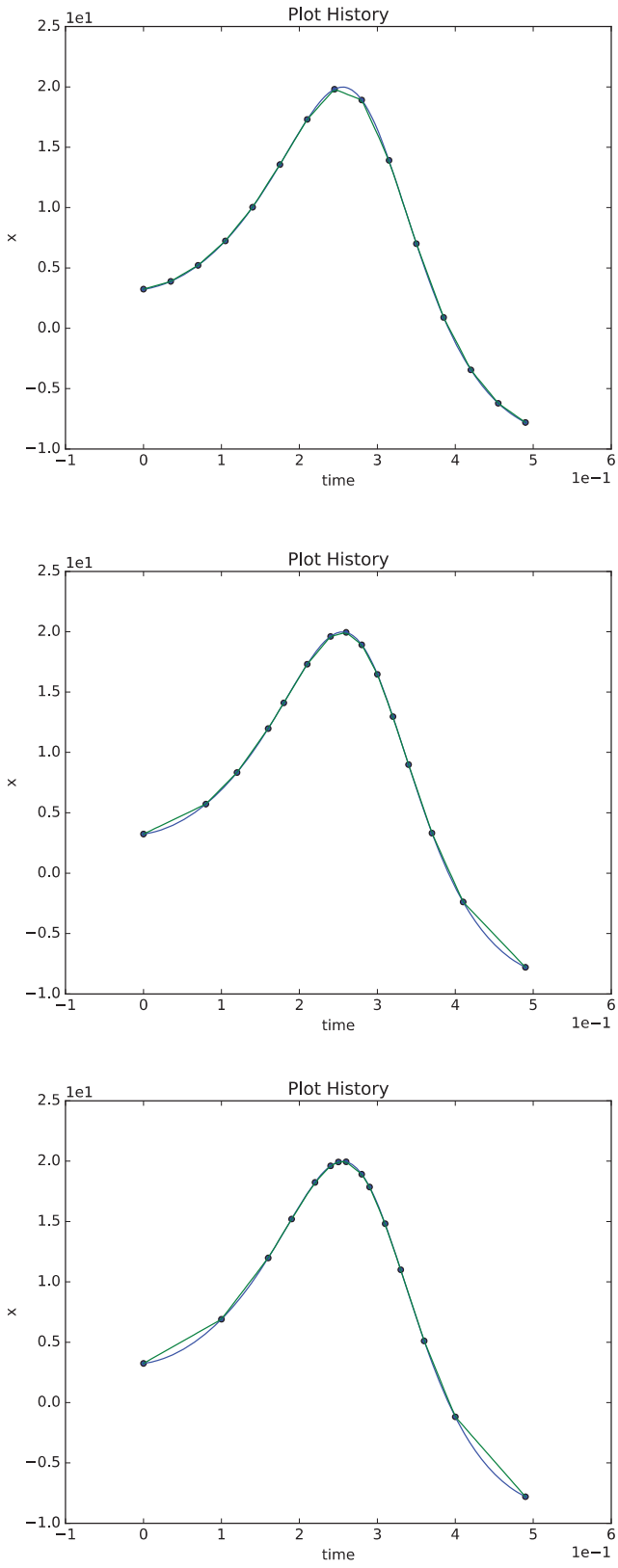

Figure 18. Plot of three of the time-series re-sampling strategies avaibale in RAVEN: uniform (top), first-derivative (middle) and second-derivative (bottom).

6.2 Data Representation

One of the most fundamental modeling choices regarding time dependent data is how each time series is actually represented in the data mining process. Reference [29] provides a broad analysis of the many representation methods. Some of these methods have been implemented in RAVEN; the choice of these implemented methods was based on their effectiveness on nuclear engineering applications. In the following sub-sections we will present these methods in mode detail along with some preliminary results obtained by RAVEN.

We also include a method (see Section 6.2.5) that have been investigated but not directly implemented in RAVEN. This method employs a symbolic conversion of the raw data (i.e., not numbers but characters) which has the advantage that many text mining algorithms can be exploited (e.g., Markov Models). It has been chosen to post-pone the implementation of this methods into future developments since:

- RAVEN internal data objects format does not allow to include symbolic data
- Such extension would require major coding effort
- Symbolic data mining is still an R&D activity even the promising capabilities

6.2.1 Real-valued

The original format of the time series is maintained. This approach does not require any prior knowledge from the user so it can be considered a fail-safe approach. On the other side this method (depending on the data set) can be memory and computationally intensive.

6.2.2 Polynomial

The time series is approximated by a Taylor polynomial function (see Figure 19) up to a fixed degree and the vector of coefficients are retained as representatives for the time series. Recall that $\boldsymbol{\theta}_n = \{x_1^n, \dots, x_M^n\}$ contains the temporal evolution of a set of M variables (i.e., $x_1^n = x_1^n(t)$), for the Taylor case for example, the approximation is performed as follows for each $x_1^n(t)$:

$$x_1^n(t) \cong \sum_{\varsigma=0}^{C} c_\varsigma \, t^\varsigma \tag{22}$$

The representation process using Taylor expansion replace $x_1^n(t)$ with a vector having dimensionality $C+1$ containing all coefficients c_ς $(\varsigma = 0, \dots, C)$.

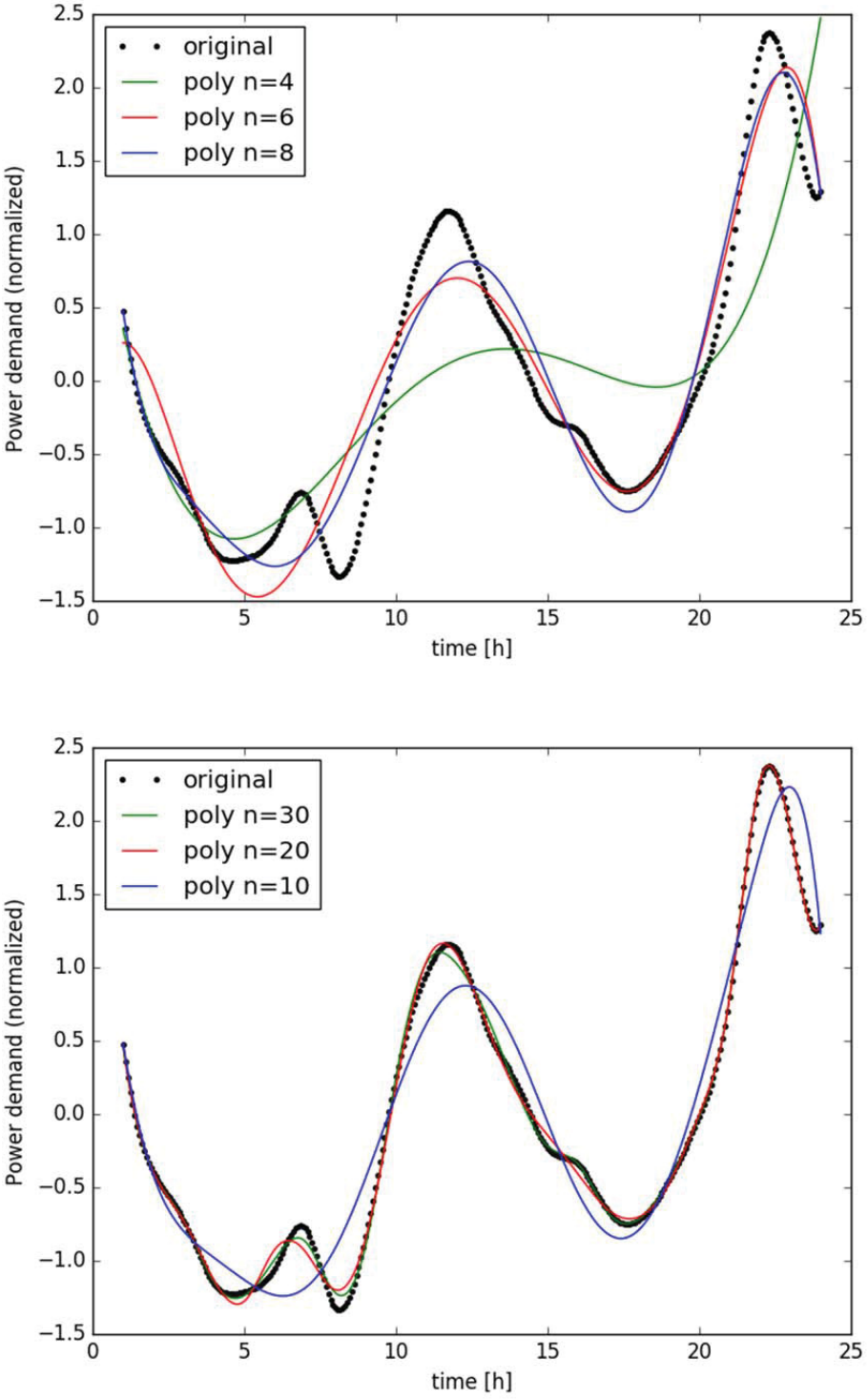

Figure 19. Polynomial approximation of a time series for several polynomial degrees.

6.2.3 Chebyshev

The Chebyshev representation follows exact principle presented above for the Taylor case (see Figure 20):

$$x_1^n(t) \cong \left[\sum_{\varsigma=1}^{C-1} c_\varsigma \, T_\varsigma(t)\right] - \frac{1}{2} c_0 \tag{23}$$

where $T_\varsigma(t)$ is the Chebyshev polynomial of order ς:

$$\begin{aligned} &T_0(t) = 1 \\ &T_1(t) = t \\ &T_2(t) = 2\,t^2 - 1 \\ &T_3(t) = 4\,t^3 - 3\,t \\ &T_4(t) = 8\,t^4 - 8\,t^2 + 1 \\ &\cdots \\ &T_{\varsigma+1}(t) = 2\,t\,T_\varsigma(t) - T_{\varsigma-1}(t) \end{aligned} \tag{24}$$

The representation process using Chebyshev expansion replace $x_1^n(t)$ with a vector having dimensionality $C + 1$ containing all coefficients c_ς $(\varsigma = 0, \ldots, C)$.

6.2.4 Legendre

The Legendre polynomials are polynomials (see Figure 21) of the following form:

$$\begin{aligned} &P_0(t) = 1 \\ &P_1(t) = t \\ &P_2(t) = (2\,t^2 - 1)/2 \\ &P_3(t) = (5\,t^3 - 3\,t)/2 \\ &P_4(t) = (35\,t^4 - 30\,t^2 + 3)/8 \\ &\cdots \\ &T_{\varsigma+1}(t) = \frac{(2\,\varsigma - 1)\,t\,T_{\varsigma-1}(t) - (\varsigma - 1)T_{\varsigma-2}(t)}{\varsigma} \end{aligned} \tag{25}$$

The representation process using Chebyshev expansion replace $x_1^n(t)$ with a vector having dimensionality $C + 1$ containing all coefficients c_ς $(\varsigma = 0, \ldots, C)$.

6.2.5 Laguerre

The Laguerre polynomials $L_n(t)$ are polynomials (see Figure 23) of the following form::

$$\begin{aligned} &L_0(t) = 1 \\ &L_1(t) = 1 - t \\ &\ldots \\ &(n+1)L_n(t) - (2n + 1 - t)L_n(t) + n\,L_{n-1}(t) = 0 \end{aligned} \tag{26}$$

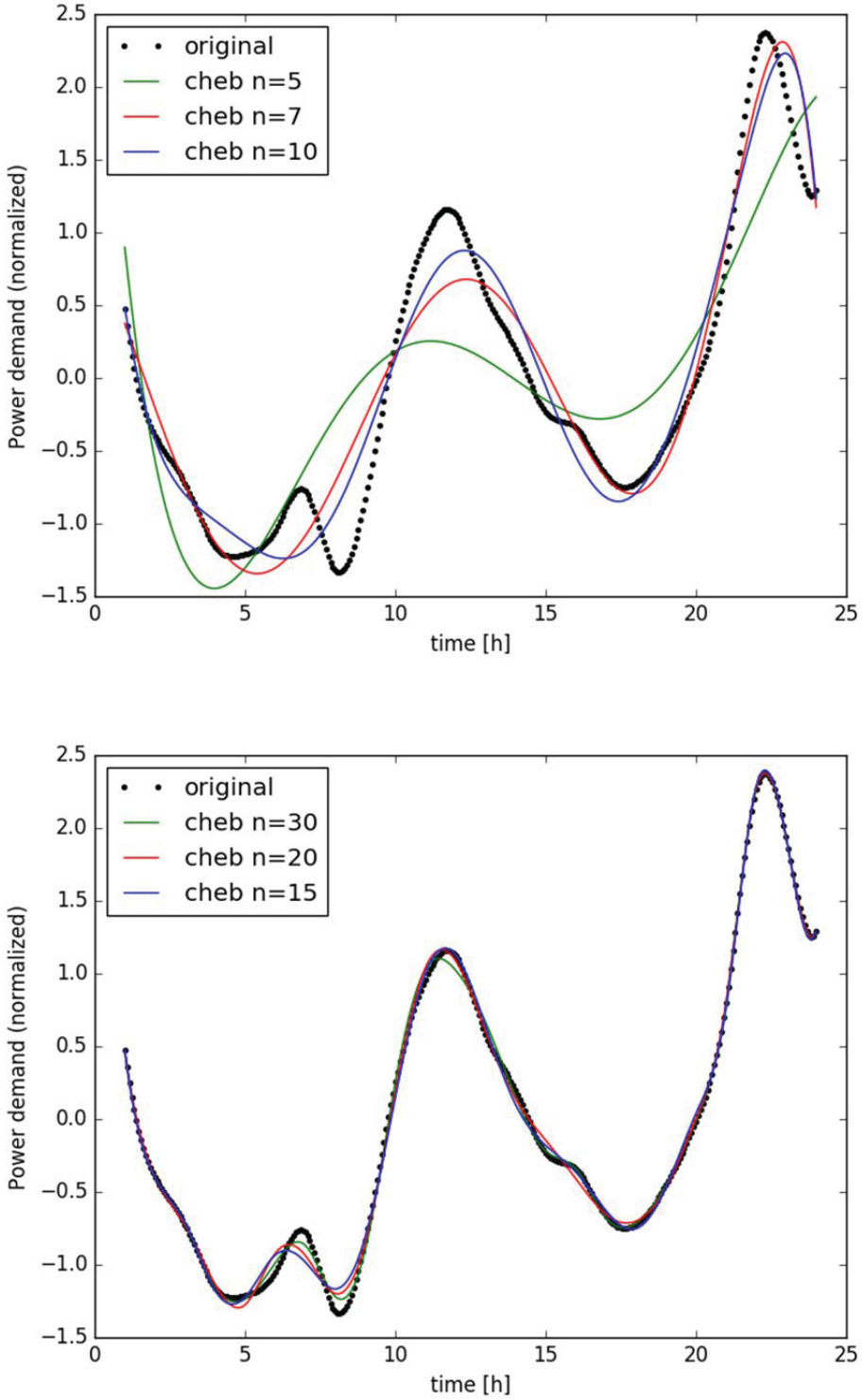

Figure 20. Chebyshev approximation of a time series for several polynomial degrees.

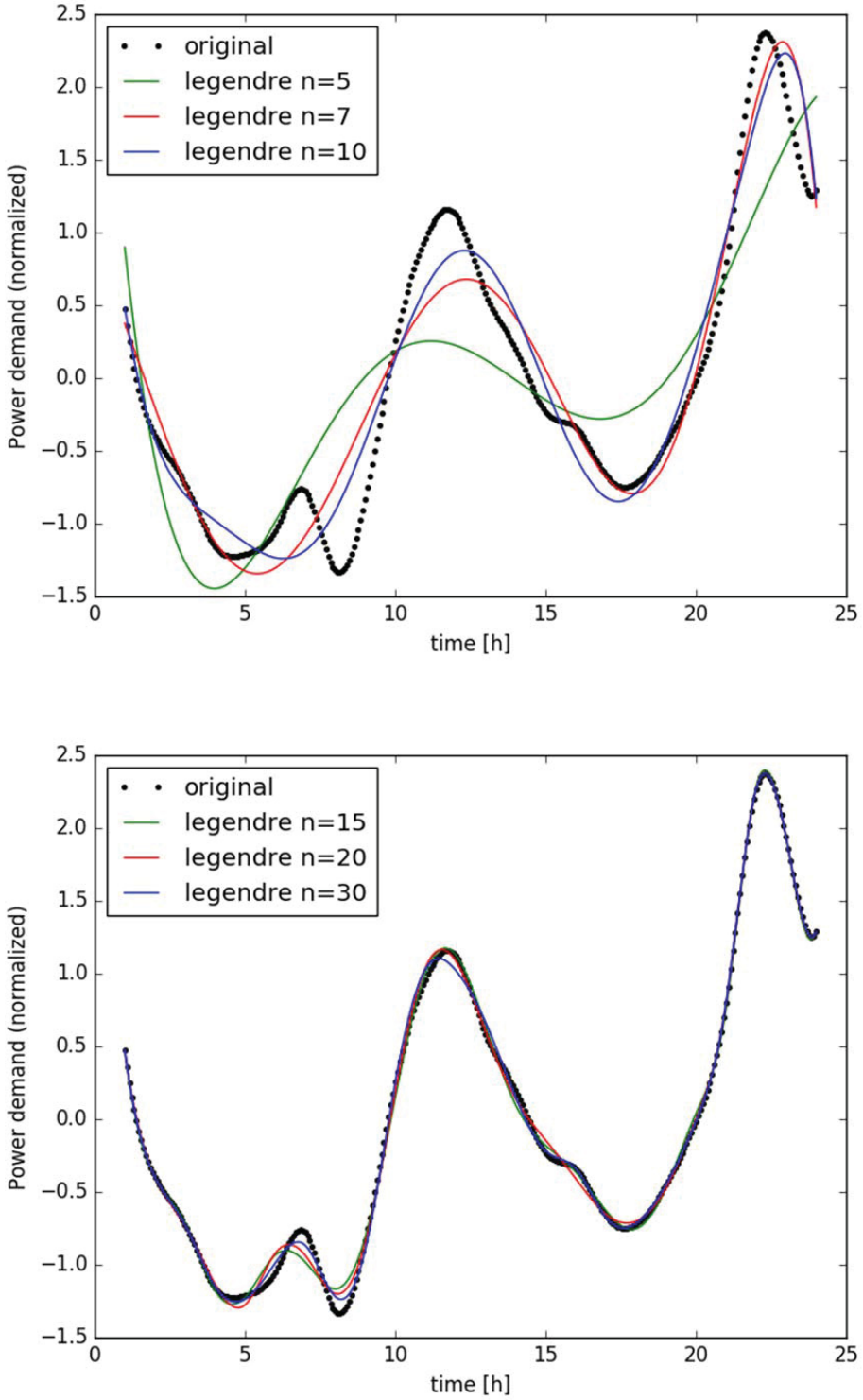

Figure 21. Legendre approximation of a time series for several polynomial degrees.

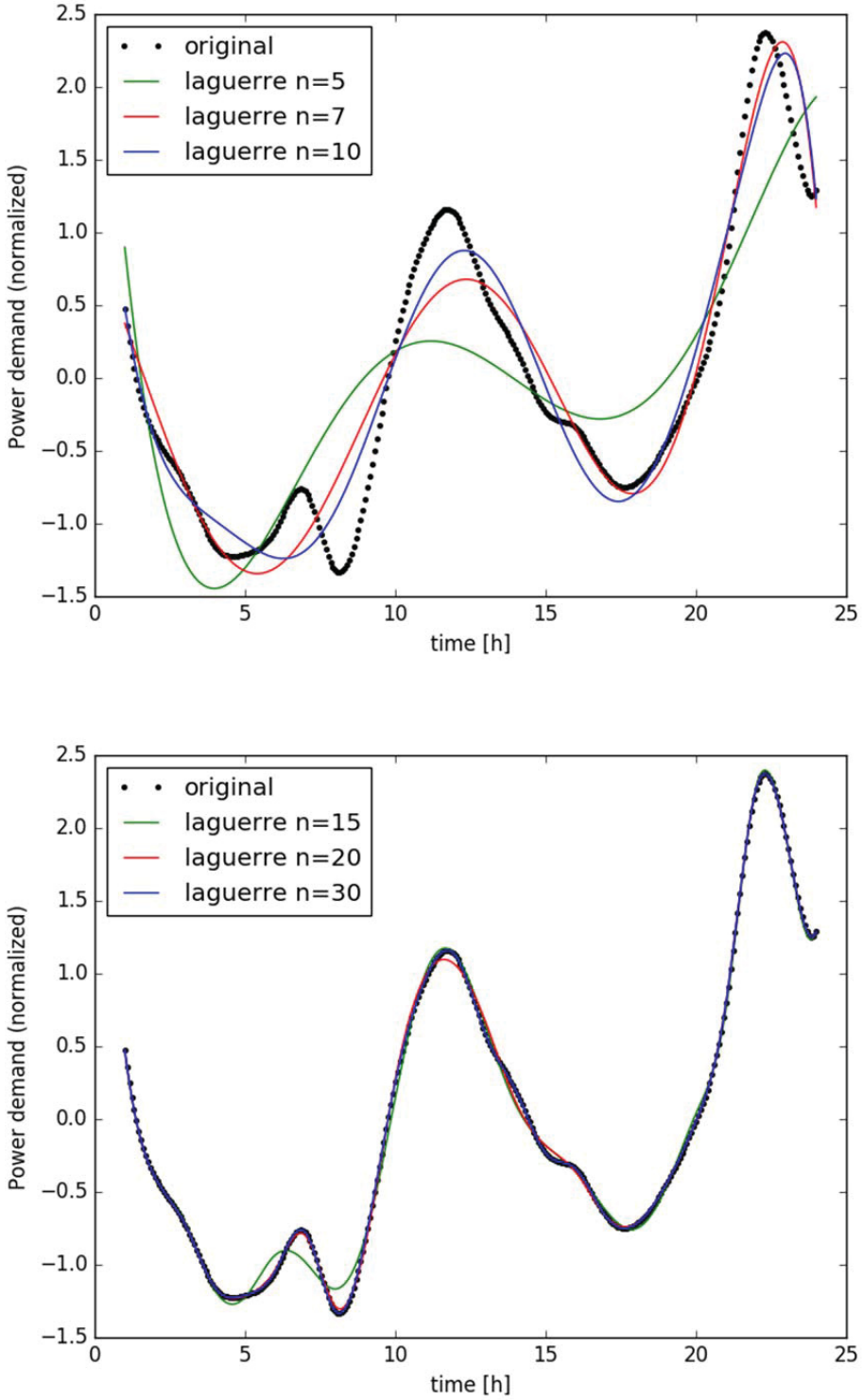

Figure 22. Laguerre approximation of a time series for several polynomial degrees.

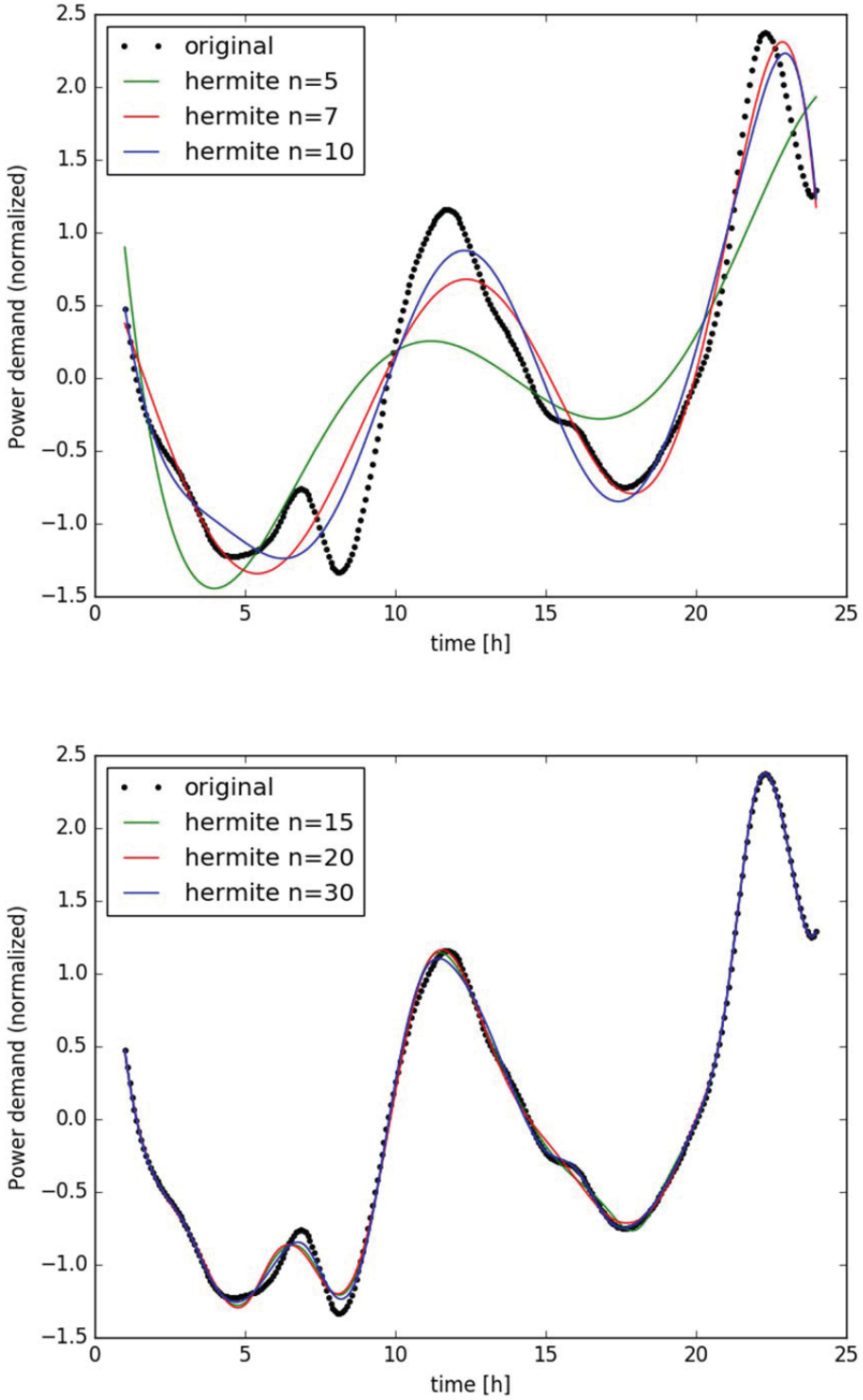

Figure 23. Hermite approximation of a time series for several polynomial degrees.

6.2.6 Hermite

The Hermite polynomials $H_n(t)$ are polynomials of the following form (see Figure 23):

$$\begin{aligned} &H_0(t) = 1 \\ &H_1(t) = t \\ &\dots \\ &H_{n+1}(t) - 2\,t\,H_n(t) + 2\,n\,H_{n-1}(t) = 0 \end{aligned} \tag{27}$$

6.2.7 Discrete Fourier Transform

Similar to the polynomial representation, the time series is approximated by a Fourier series and the series coefficients are retained as representatives for the time series. The Fourier series is as follows (see Figure 24):

$$x_1^n(t) \cong \frac{a_0}{2} \sum_{\varsigma=1}^{C} \left(a_\varsigma \cos(\varsigma\, t) + b_\varsigma \cos(\varsigma\, t) \right) \tag{28}$$

6.2.8 Singular Value Decomposition (SVD)

This method performs an eigenvalues and eigenvectors decomposition of $\boldsymbol{\theta}_n$ and selects a reduced set of eigenvectors. Each time series H_n is represented by the coefficients associated to each eigenvector. Note that this decomposition must be performed on all time-series as a whole since SVD decomposition is performed on the covariance matrix which is calculated by considering all set of time series and not one time series separately. This is performed for each x_m $(m = 1, \dots, M)$ independently by:

1. Normalizing the data: zero mean and unit variance (see Figure 25)
2. Resampling the data set so that all time series have been sampled on the exact same time instants
3. Determining the covariance matrix
4. Performing SVD of the covariance matrix, i.e., eigenvalues and eigenvectors decomposition (see Figure 26). Note that each eigenvector is a time series sampled at the same time instants of the original time series. The eigenvectors can be ranked based on the associated eigenvalues: the space reduction can be performed by considering the eigenvector with higher eigenvalues (see Figure 27)
5. Projecting the original time series into the eigenvectors space (either reduced or not) and using the projection coefficients as time series representation format.

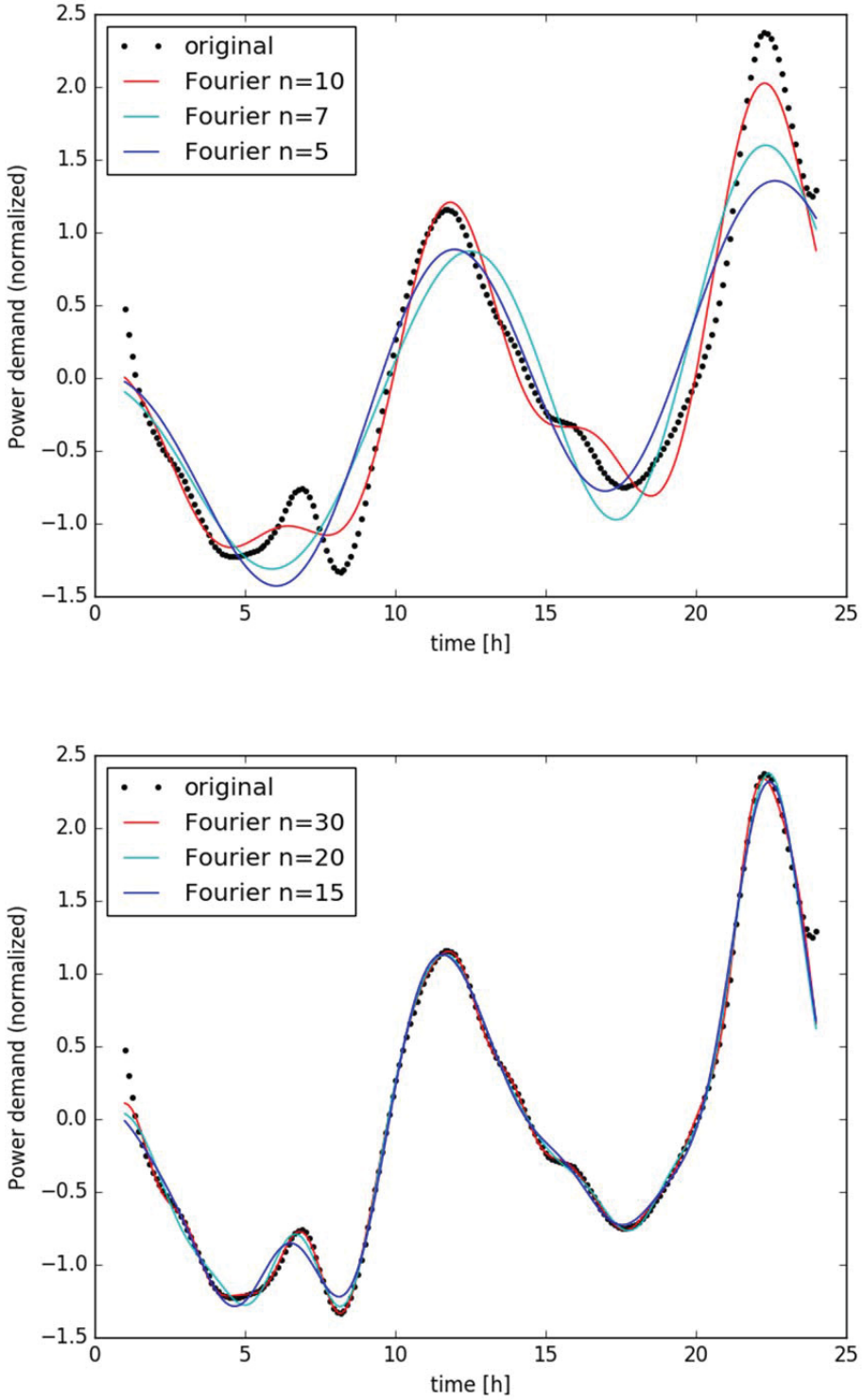

Figure 24. Fourier approximation of a time series for several polynomial degrees.

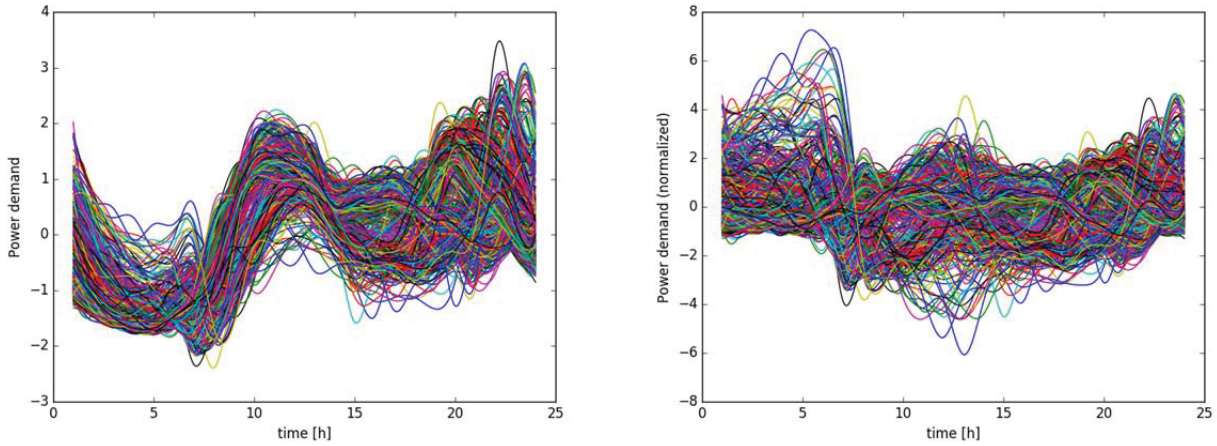

Figure 25. Plot of the original (left) and normalized (right) data sets used to to test the SVD representation.

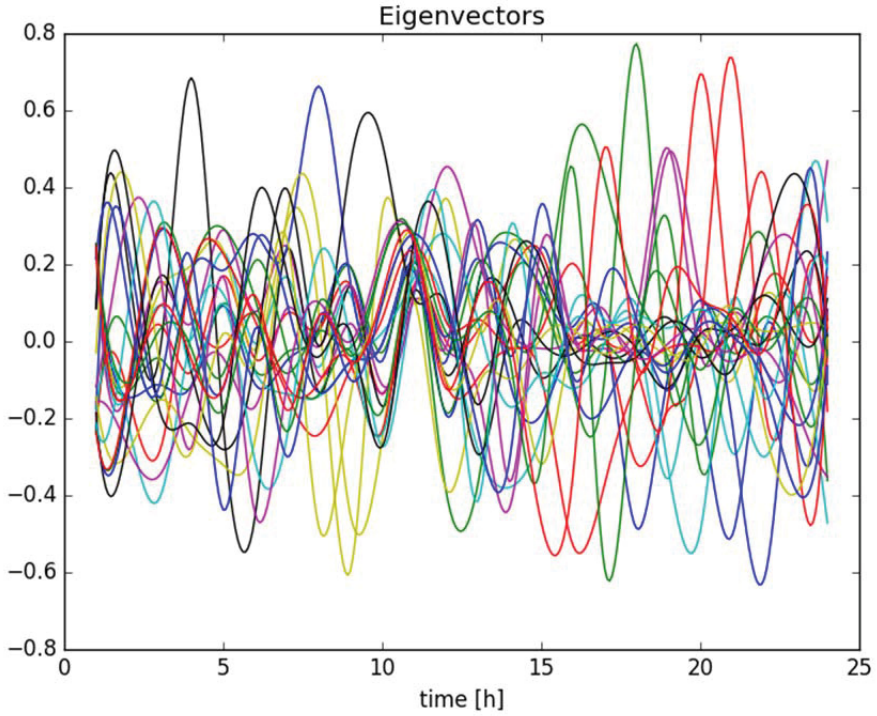

Figure 26. Plot of the eigenvectors of the data set shwon in Figure 25.

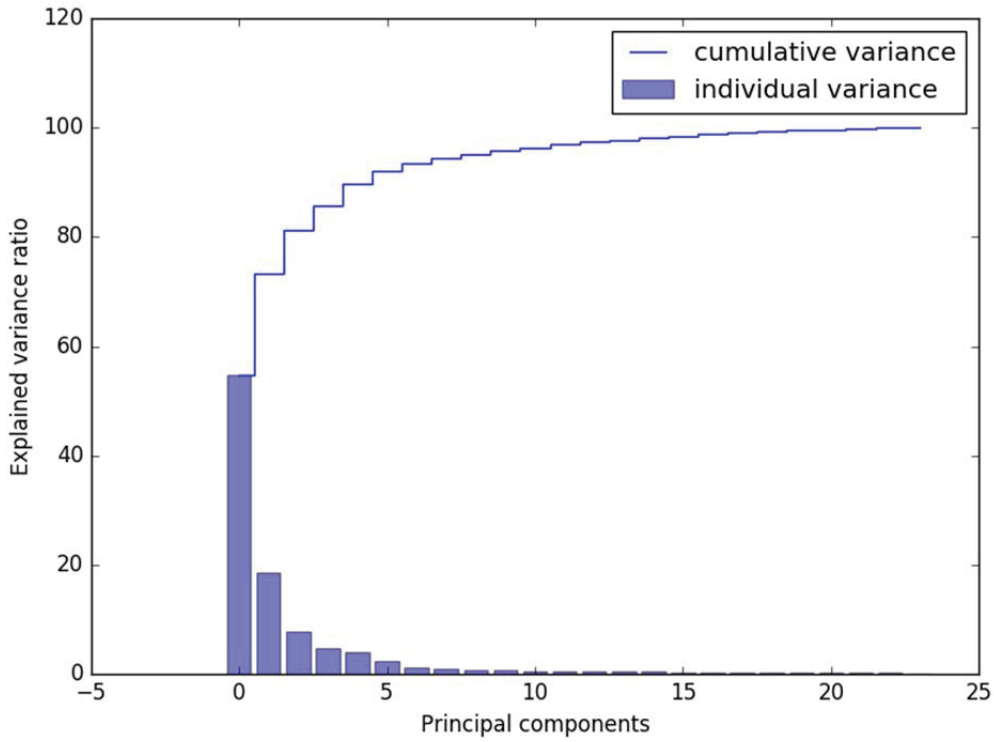

Figure 27. Plot of the individual and cumulative variance for each of the 24 eigenvectors.

6.2.9 Symbolic

In short, this method performs a symbolic conversion of $\boldsymbol{\theta}_n$: i.e., the original numerical data is converted as a an ordered list of symbols (e.g., letters). This capability has not been implement in RAVEN due to the fact that RAVEN can handles only numerical data and the handling of symbolic data would require large development of the code itself. Given the fact that this representation method is still in a very early stage of evaluation, it has been decided to leave the development on hold. From an R&D point of view an initial testing has been performed and it was shown in [30]; this section summarizes such testing.

The algorithm we evaluated is based on SAX [31] which is an algorithm that allows the user to represent continuous time-varying data S as a series of n symbols $\bar{\bar{S}} = \bar{\bar{s}}_1, \bar{\bar{s}}_1, \dots, \bar{\bar{s}}_n$ where $\bar{\bar{s}}_i$ is a symbol.

While temporal discretization is performed by partitioning the time axis into n intervals having the same length, the discretization of $\theta(t)$ is typically performed by dividing the range of θ into α equi-probable regions. Each region has a character $\bar{\bar{s}}$ associated to it andthe alphabet size has cardinality15 α. The resulting conversion generates a time series of length n and an alphabet size equal to α. The SAX algorithm consists of the steps (also see Figure 28) described in Algorithm 1.

[15] Cardinality of the alphabet refers to the number of characters used.

Algorithm 1: SAX Algorithm
1: Input: n and α
2: Normalize the data (mean equal to 0 and standard deviation equal to 1)
3: Partitionthe temporal interval into n equal sized intervals
4: Divide the distribution of $\theta(t)$ into a equi-probable regions and assign a symbol to each region (alphabet has cardinality equals to α)
5: Consider the average value $\bar{s}$ of $\theta(t)$ in each interval
6: For each $\bar{s}$ assign its own $\bar{\bar{s}}$ according to the discretization performed in Step 4
7: Generate a phrase $\bar{\bar{S}}$ as a timely ordered sequence of symbols

The end result is a phrase $\bar{\bar{S}}$: a timely ordered sequence of symbols. An example of discretization for the temporal profile of a scenario taken from [28] is shown in Figure 28 and Figure 29.

Typically, data generated by simulations contain the temporal profile of multiple variables; moreover, as also shown in Figure 28, a fixed number (i.e., *n*) of time intervals having equal length is not optimal to capture rapid changes of *S*.

The issue of dealing with multiple variables can be solved by:

- Performing Steps 2 and 4 in Algorithm 1 independently for each variable, and,
- Maintaining the order of symbols for every variable in each time interval

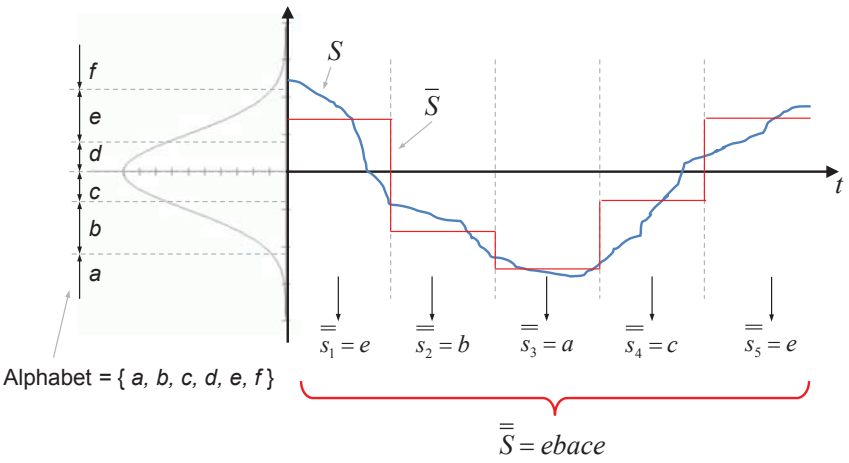

Figure 28. Example of symbolic conversion with a = 6 and n = 5.

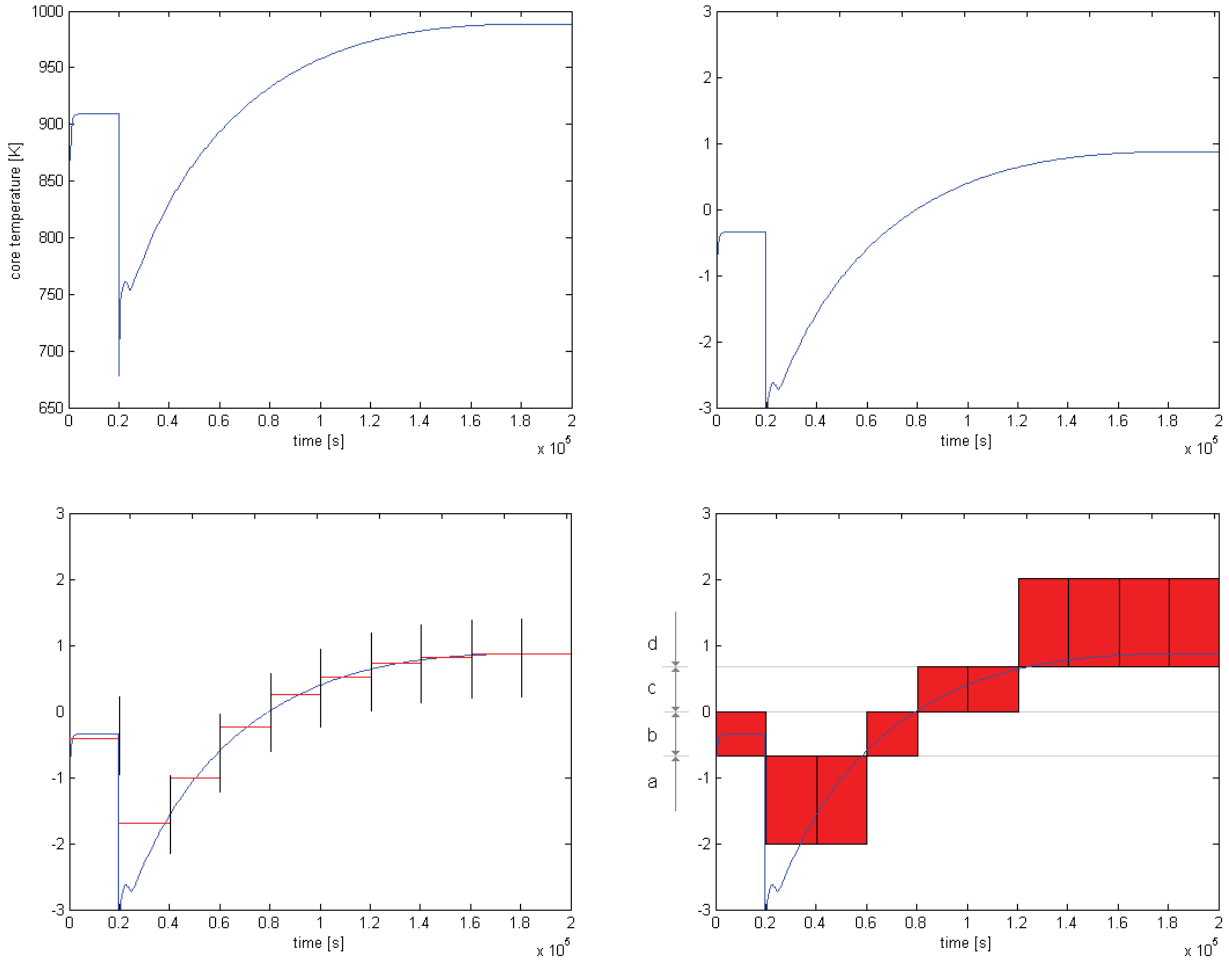

Figure 29. Application of the SAX algorithm [31] for a nuclear transient: raw data (top left), data normalized (top right), temporal discretization (bottom left) and symbol sequence generation (bottom right).

Regarding the issue of identifying rapid changes of state variables, a solution is to recursively analyze the rate of change of the covariance matrix computed in that interval (as shown in [31]). The rationale is to choose time intervals such that the rate of change of the covariance matrix eigenvalues is below a fixed threshold (see Algorithm 2). As input, the minimum and a maximum length of the time interval are required in order to preserve accuracy and avoid extremely long phrases.

Algorithm 2: Adaptive Time Discretization
1: Input: maximum value for the rate of change of the eigenvalues of the covariance matrix, $\dot{\lambda}_{Max}$; minimum and maximum length of the time discretization, i.e., t_{Min} and t_{Max}
2: Divide the time scale in intervals having length t_{Max}
3: Evaluate the covariance matrix of the data points contained in that interval and determine its eigenvaluesλ_i
4: Determine the highest eigenvalue relative variation $\dot{\bar{\lambda}}$
5: If $\dot{\bar{\lambda}}>\dot{\lambda}_{Max}$ then split the interval into 2 intervals of equal length
6: Repeat Steps 3 and 4 for all intervals

From our experience, the choice of the optimal values of n and α is case dependent. In our application we aim to choose values of n andα such that no two scenarios are represented by the same phrase. Figure 30 shows an example of time adaptive discretization of a scenario characterized by 2 variables.

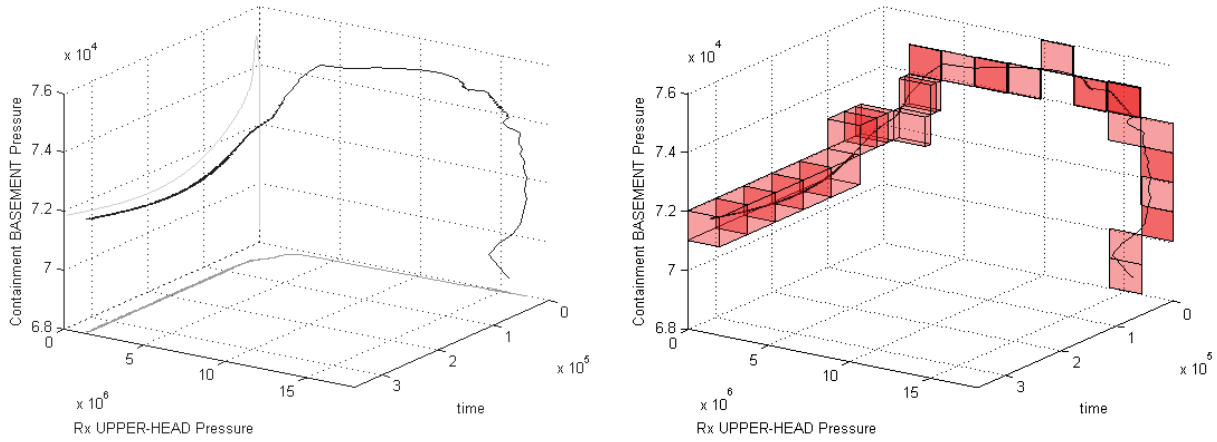

Figure 30. Adaptive time discretization of multi-dimensional scenarios; a 2-variable case (containment and reactor pressure vs. time) is shown: original data (left) and corresponding discretization (right).

6.3 Measuring Similarities

The second important modeling choice when dealing with time series regards the type of similarity metric also known as distance. Similar to the theory behind distances in Euclidean space, a distance metric $d(S,Q)$ measures the "similarity" between two time series S and Q. Recall that $d(S,T)$ has to obey the following rules:

$$\begin{cases} d(S,S) = 0 \\ d(S,Q) = d(Q,S) \\ d(S,Q) = 0 \text{ iff } S = Q \\ d(S,Q) \leq d(S,K) + d(K,Q) \end{cases} \tag{29}$$

When dealing with time series, the following two metrics are the most commonly use: Euclidean and Dynamic Time Warping (DTW) [20] distance. These distances are described in the next two subsections for the univariate case, i.e., two time series Q and S where their continuous part has $M = 1$. The more generic case, i.e., multivariate case, can be easily expanded from what is shown below.

6.3.1 Euclidean distance

Given two univariate time series S and Q having identical length (i.e., $T_S = T_Q$) the Euclidean distance $d_2(S,Q)$ is defined as:

$$d_2(S,Q) = \sqrt{\sum_{t=0}^{T_S} \left(x_1^S(t) - x_1^T(t)\right)^2} \tag{30}$$

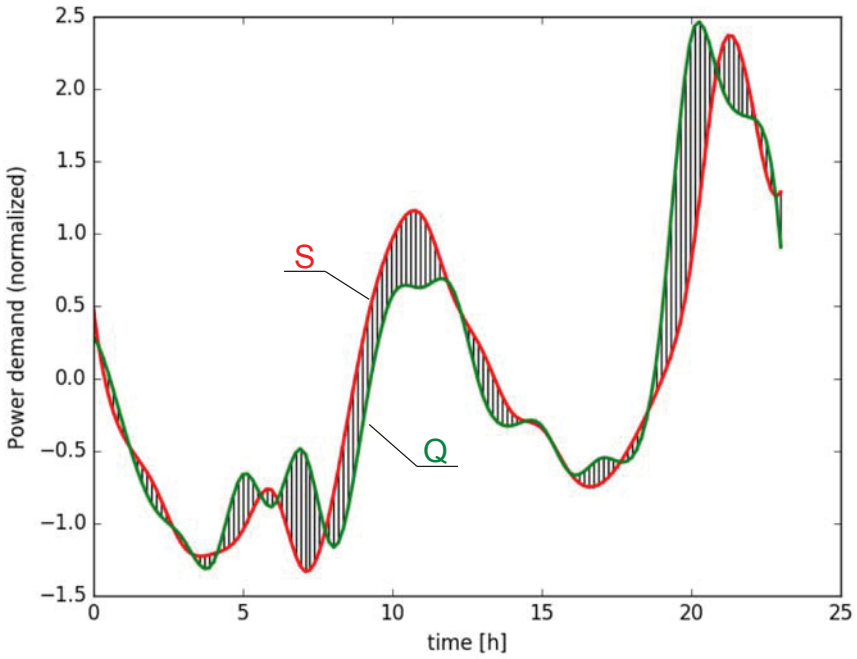

Figure 31. Euclidean distance metric for two time series S and Q. Each black segment represents: $x_1^S(t) - x_1^T(t)$.

6.3.2 DTW Distance

This distance can be viewed as a natural extension of the Euclidean distance applied to time series [20]. Given two univariate time series S and Q having length T_S and T_Q respectively[16]. The distance value $d_{DTW}(S,Q)$ is calculated by following these two steps:

1. Create a matrix $D = [d_{i,j}]$ having dimensionality $T_S \times T_Q$ where each element of D (see Fig. 3 for the time series shown in Fig. 4) is calculated as $d_{i,j} = (x_1^S[i] - x_1^Q[j])^2$ for $i = 1, \dots, T_S$ and $j = 1, \dots, T_Q$.
2. Search a continuous path $w_k|_1^K$ in the matrix D that, starting from $(i,j) = (0,0)$, it ends at $(i,j) = (T_S, T_Q)$ and it minimizes the sum of all the K elements $w_k = \left(d_{i,j}\right)_k$ of this path (see blue line in Fig. 3):

$$d_{DTW}(S,Q) = \min\left(\sum_{k=1}^{K} w_k\right) \tag{31}$$

Each element of the path corresponds to a specific black segment in Fig. 4. This metric can capture similarities between time series that are shifted in time.

[16] Note that here he we have relaxed the requirement: $T_S = T_Q$

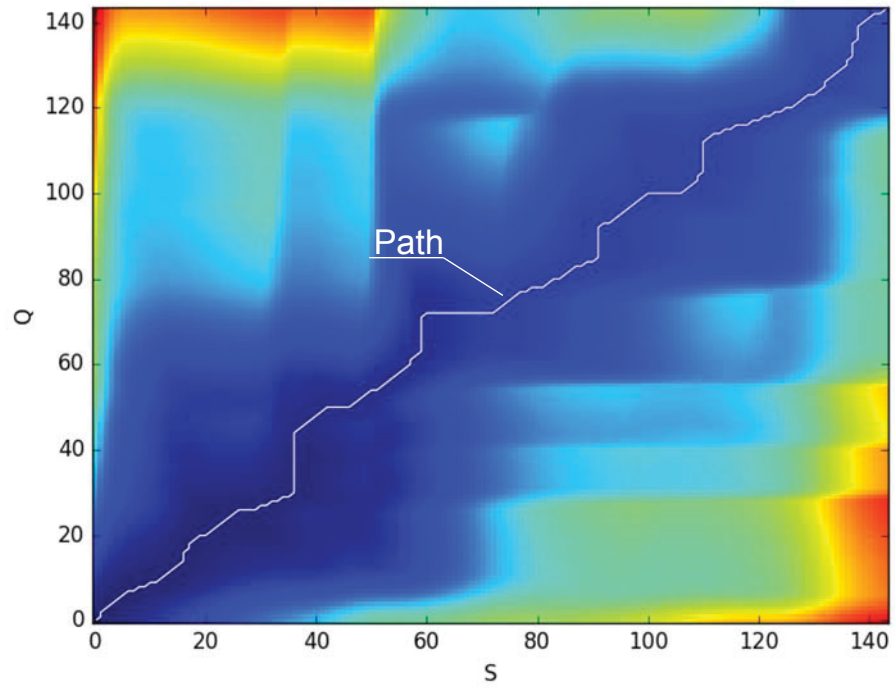

Figure 32. Colored plot of the distance matrix D for the time series S and Q plotted in Figure 31. White line represents the warp path w_k $(k = 1, \dots K)$.

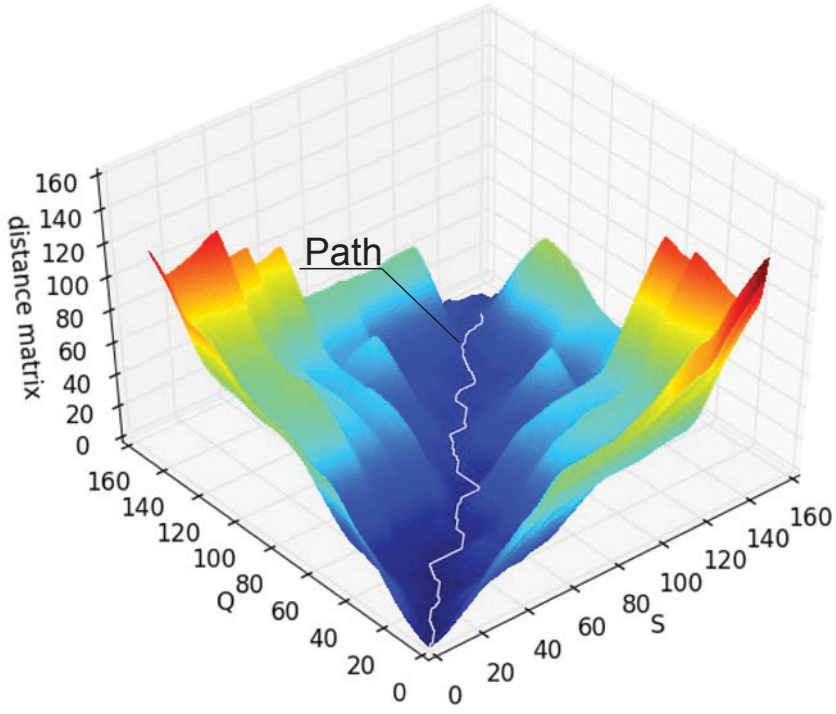

Figure 33. 3D plot of the of the distance matrix D for the time series S and Q plotted in Figure 31. White line represents the warp path w_k $(k = 1, \dots K)$.

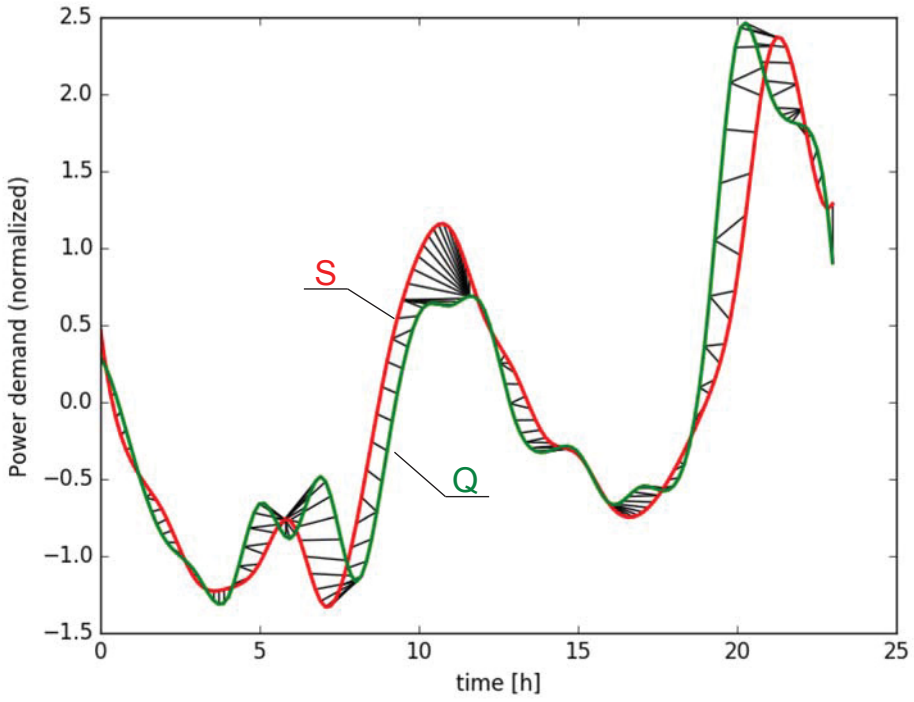

Figure 34. DTW distance metric for two time series S and Q. Each black segment represents an elements $w_k = (d_{i,j})_k$ of the warp path shown in Figure 33.

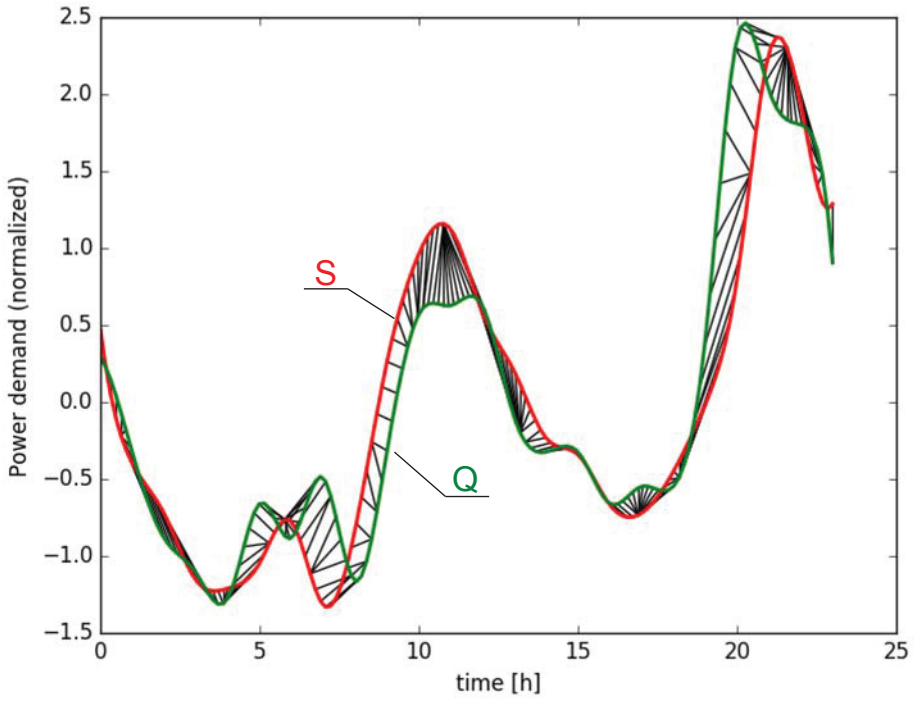

Figure 35. Derivative DTW distance metric for two time series S and Q.

6.4 Path 1: From Time-Dependent To Static Data

In Section 5.2 we have indicated two possible paths are available in RAVEN to analyze time-dependent data. As a reminder, the first path consists of transforming the time time-dependent data into static data. This is possible by performing the following steps in RAVEN:

1. Load data either from file or from a database (as a third option, the data can be generated in the same RAVEN input file through a Monte-Carlo sampling process for example)
2. Re-sample data: perform a temporal re-sampling of the data (see Section 6.1). This step might include also time-series synchronization: all time-series are sampled at the same time instants.
3. Cluster data: perform the actual clustering of the data (e.g., through Mean-Shift algorithm) on the pre-processed data. This step includes the conversion from time-dependent to static data where the conversion step is embedded in the definition of the Mean-Shift algorithm through the PreProcessor keyword which reference the conversion post-processor HSPS. The HSPS post-processor perform the actual conversion prior running Mean-Shift using a real-value conversion (see Sections 6.2.1 and 4.4.3)

```
<PostProcessor name="dataPreProc" subType="InterfacedPostProcessor">
   <method>HS2PS</method>
   <timeID>time</timeID>
 </PostProcessor>

 <PostProcessor name="MeanShift" subType="DataMining">
   <PreProcessor  class="Models" type="PostProcessor">dataPreProc</PreProcessor>
   <KDD lib="SciKitLearn">
     <SKLtype>cluster|MeanShift</SKLtype>
     <Features>output</Features>
     <bandwidth>12</bandwidth>
   </KDD>
 </PostProcessor>
```

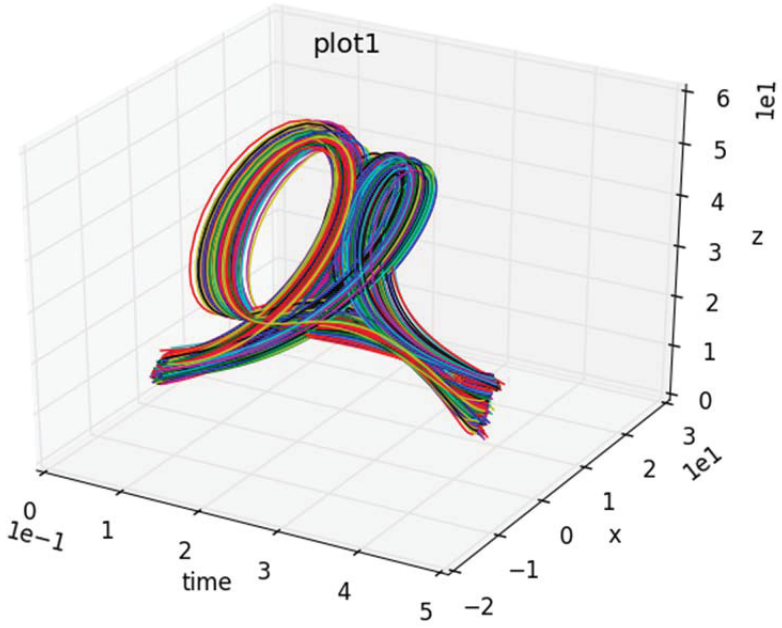

Figure 36. Plot of the time-dependent data set.

An example is shown in Figure 36 for a set of 1000 time series in a 4-dimensional space. Using the Mean-Shift algorithm we were able to detect 2 clusters by varying the bandwidth. The plot of the scenarios belonging to each cluster is shown in Figure 37 while Figure 38 plot the obtained cluster centers for each of the two clusters.

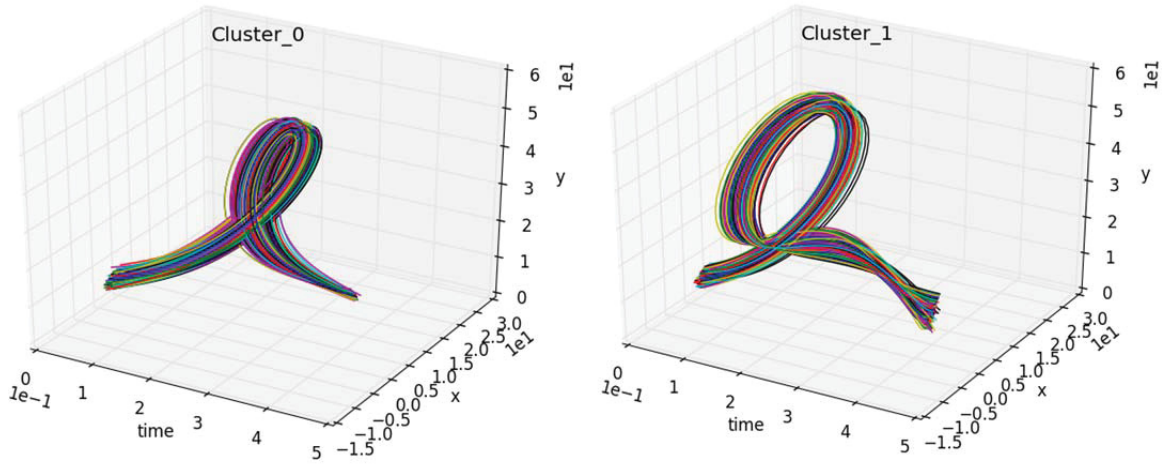

Figure 37. Plot of the time series belonging to each of the two clusters (cluster_0 and cluster_1) using Mean-Shift.

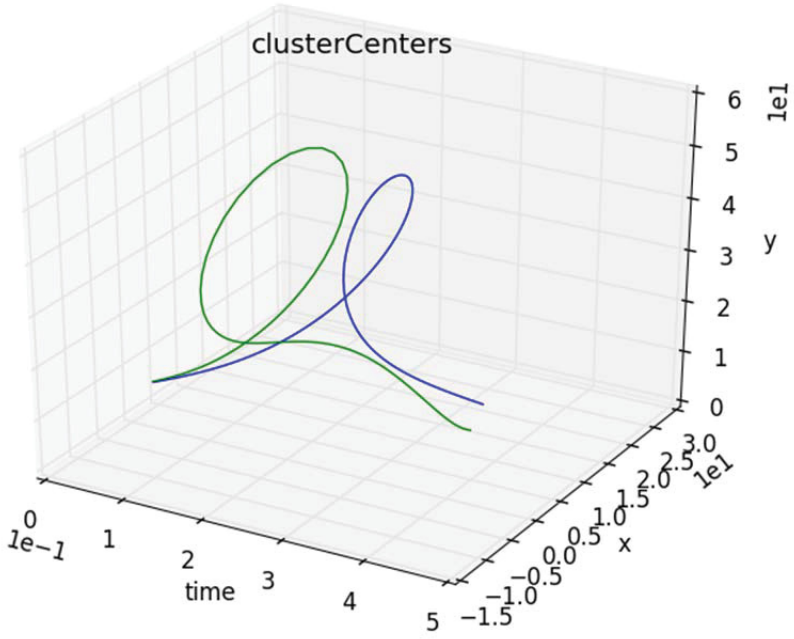

Figure 38. Plot of the cluster centers for each of the two clusters.

6.5 Path 2: Clustering Through Similarity Matrix

In Section 5.2 we have indicated also the second path available in RAVEN to analyze time-dependent data. This path leverages the clustering algorithms that can receive in input the similarity matrix. In RAVEN this is possible by performing the following steps:

1. Load data either from file or from a database (as a third option the data can be generated in the same RAVEN input file through a Monte-Carlo sampling process for example)
2. Re-sample data: perform a temporal re-sampling of the data (see Section 6.1). This step might include also time-series synchronization: all time-series are sampled at the same time instants.

3. Cluster data: perform the actual clustering of the data (e.g., through Hierarchical algorithm) on the pre-processed data. In this step two RAVEN entities must be defined: the metric (i.e., Euclidean or DTW) and the hierarchical post-processor as shown below. Note that the metric to be employed in the hierarchical post-processor is linked in the post-processor itself. If a time-dependent data metric is not linked then the hierarchical algorithm expect as input static data.

```
  <Metrics>
    <DTW name="example" subType="">
      <order>0</order>
      <pivotParameter>time</pivotParameter>
      <localDistance>euclidean</localDistance>
    </DTW>
    <Minkowski name="exampleMink" subType="">
      <p>2</p>
      <timeID>time</timeID>
    </Minkowski>
  </Metrics>

</Models>
    <PostProcessor name="hierarchical" subType="DataMining" verbosity="quiet">
      <Metric class="Metrics" type="DTW">example</Metric>
      <KDD lib="Scipy" labelFeature='labels'>
        <Features>output</Features>
        <SCIPYtype>cluster|Hierarchical</SCIPYtype>
        <method>single</method>
        <metric>euclidean</metric>
        <level>75</level> <!-- 42 -->
        <criterion>distance</criterion>
        <dendrogram>true</dendrogram>
        <truncationMode>lastp</truncationMode>
        <p>20</p>
        <leafCounts>True</leafCounts>
        <showContracted>True</showContracted>
        <annotatedAbove>10</annotatedAbove>
      </KDD>
    </PostProcessor>
  </Models>
```

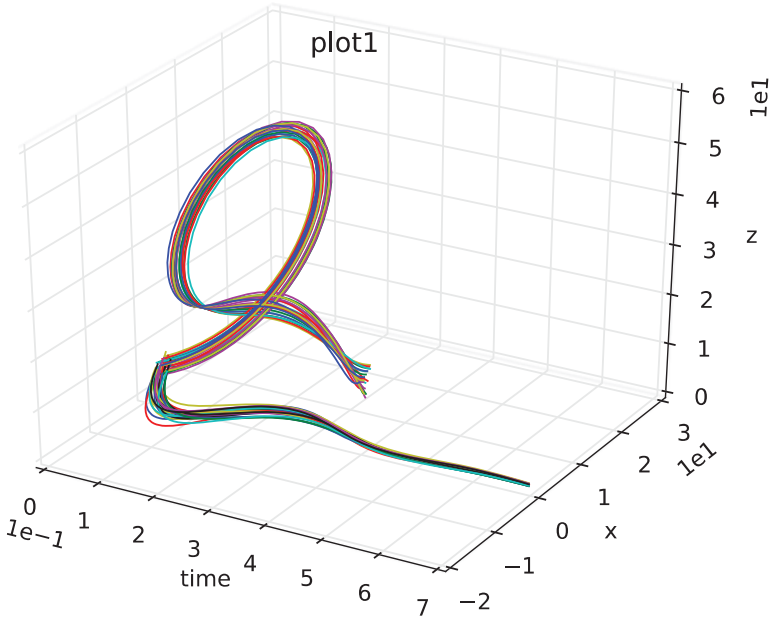

Figure 39. Plot of the time-dependent data set.

An example is shown in Figure 39 for a set of 1000 time series in a 4-dimensional space. Using the Hierarchical algorithm we were able to detect 2 clusters as indicated in the dendrogram shown in Figure 40. The plot of the scenarios belonging to each cluster is shown in Figure 41.

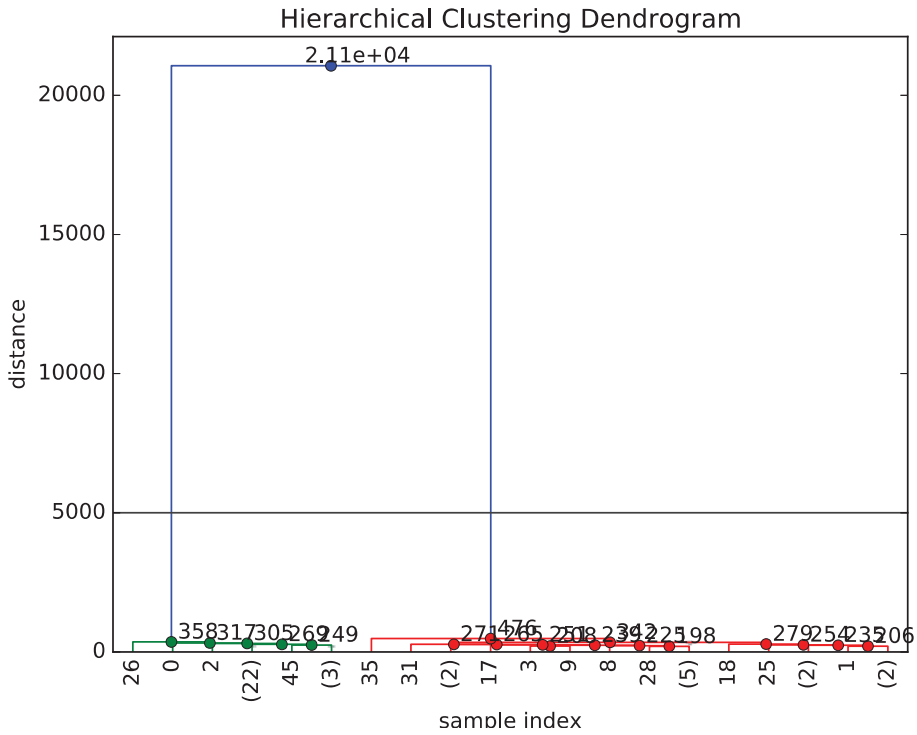

Figure 40. Dendrogram obtained from the data set shown in Figure 39.

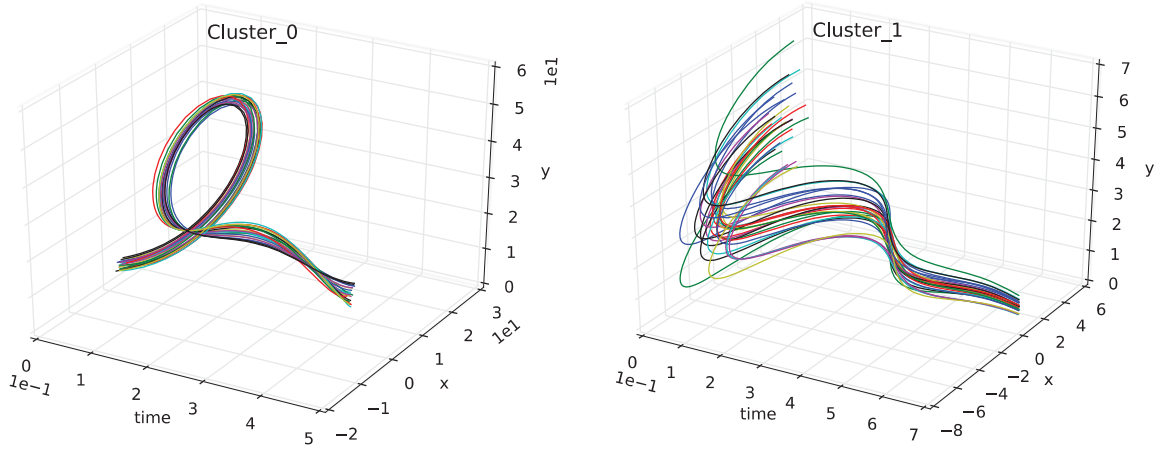

Figure 41. Plot of the time series belonging to each of the two clusters (Cluster_0 and Cluster_1) using Hierarchical algorithm.

7. RESULTS

7.1 Benchmark data sets

In the past decade the interest in time dependent data has grown exponentially. This is due to the fact that the great portion of the collected data actually consists of time series or can be represented as time series. The computer science community has responded to this interest by developing a wide variety of algorithms that can be used to perform data mining on time series.

In order to quantitatively measure algorithm performances and limitations several data sets have been produced. The objective is to define a common-ground where classification and clustering algorithms can be compared in terms of computational time, memory requirements, classification/clustering performances etc.

The UCR Time series classification archive[17] [32] contains a large variety of data sets which are freely available and usable. A drawback of this data set is that it contains only univariate data

One issue is related the fact that the original format of each data set is only Matlab compatible. For the scope of this report we have translated the original format into a RAVEN compatible one by:

1. Translating the original raw data on file (.csv format)
2. Converting the .csv file into a RAVEN historyset file set

Once completed the file set obtained in Step 2 can be loaded easily loaded and analyzed into RAVEN.

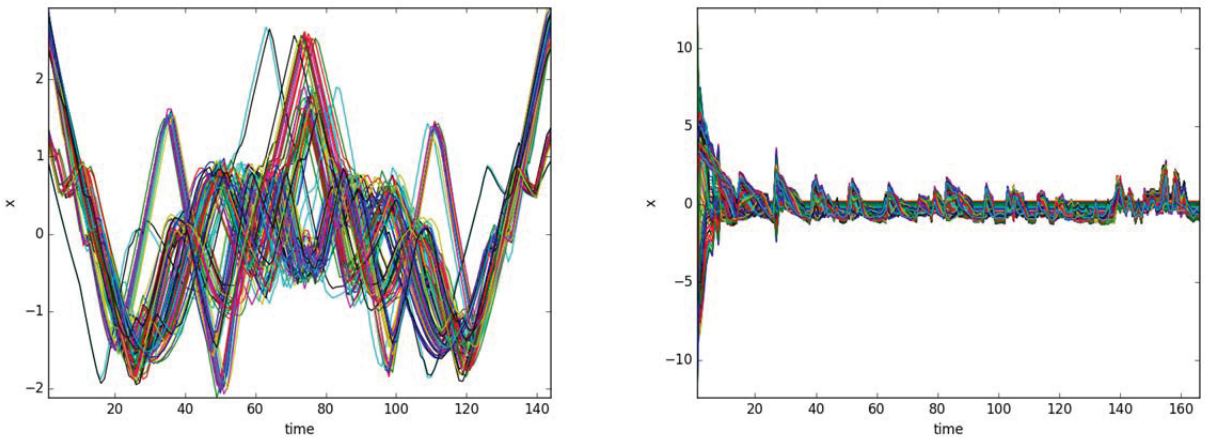

Figure 42. Plot of the *Plane* and *Chlorine concentration* data sets [32].

Only a small subset of the UCR archive data sets have been used in this fiscal year and partially reported in this document:

- Plane test (see Figure 42)
- Chlorine concentration (see Figure 42)
- ECG (see Figure 43)
- ECG_5days (see Figure 43)
- Ford (see Figure 43)

[17] - Web-site: http://www.cs.ucr.edu/%7Eeamonn/time_series_data/

- Gun_point (see Figure 43)
- Italy_power_demand (see Figure 43)
- Medical_images (see Figure 43)

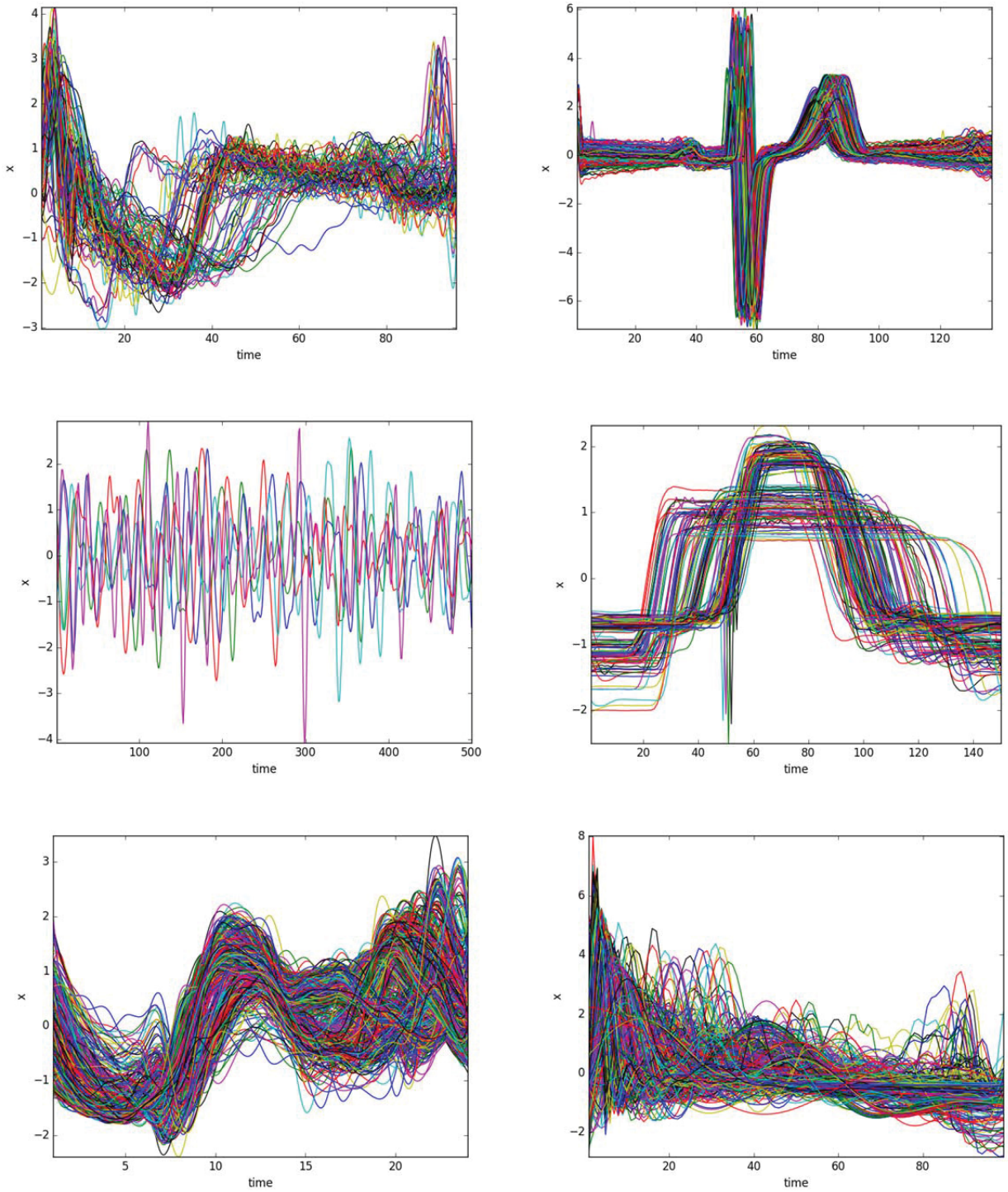

Figure 43. Plot of the *ECG*, *ECG_5days*, *ford*, *gun_point*, *power_demand* and *medical_images* data sets [32].

Even though these data sets are not strictly nuclear related, they provide a very good test bed for the algorithms developed in RAVEN.

7.2 Analytical Model

Prior to testing the algorithms on nuclear applications we tested them on a set of analytical tests. In this report we present one of them: the Lorentz attractor. This model describes the temporal evolution of the system in a 4-dimensional space (x, y, z and time t) and it is characterized by the following set of ordinary differential equations:

$$\begin{cases} \frac{dx}{dt} = \sigma(y - x) \\ \frac{dy}{dt} = x(\rho - z) - y \\ \frac{dz}{dt} = xy - \beta z \end{cases} \tag{32}$$

A RAVEN external model has been created. This model contains the set equations which are numerically solved (using finite difference scheme) given:

- Value of the parameters (static, i.e., not time-dependent) ρ, σ and β
- Initial conditions, i.e., $x(t = 0) = x(0)$, $y(t = 0) = y(0)$ and $z(t = 0) = z(0)$

Figure 44 and Figure 45 show a plot of several time series by generated by RAVEN when initial conditions (i.e., $x(0)$, $y(0)$ and $z(0)$) are stochastic variables (all normally distributed with mean $\mu = -4.0$ and sigma $\sigma = 1.0$)

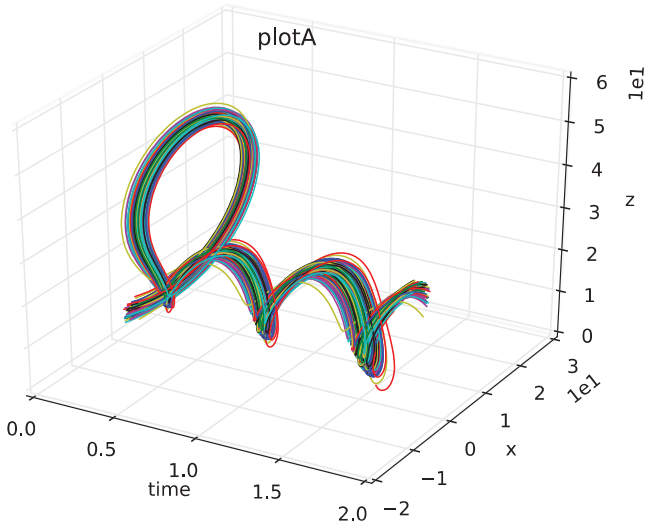

Figure 44. Time series generated by RAVEN.

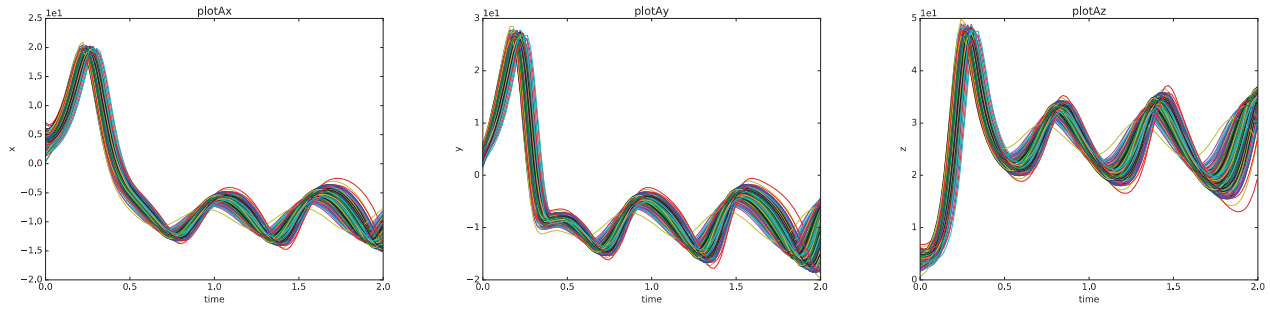

Figure 45. Time series generated by RAVEN projected in each of the three axis ($x(t)$, $y(t)$ and $z(t)$).

In order to create a more challenging problem we have deliberately introduced discontinuities in the model depending on the values of $x(0)$, $y(0)$ and $z(0)$. Such discontinuities cause trajectories with different time behavior. The discontinuities were the following:

- If $x(0) < 4.0$ then $\rho = 28.0$, else $\rho = -28.0$
- If $y(0) < 4.0$ then $\sigma = -1.0$, else $\sigma = 10.0$
- If $y(0) < 4.0$ then $\beta = \frac{35}{3}$ and simulation time is equal to 0.5, else $\beta = \frac{10}{3}$ and simulation time is equal to 0.65

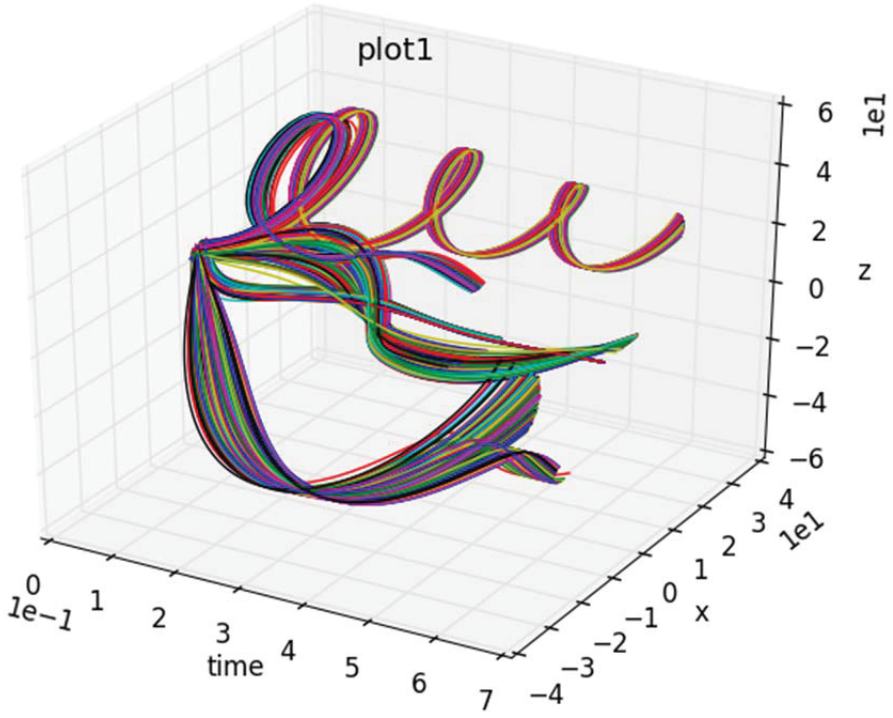

Figure 46. Data set generated by RAVEN containing multiple discountinuites and having variable time length.

Using RAVEN we have generated 1500 simulation runs using Monte-Carlo sampling (see Figure 46 and Figure 47) where $x(0)$, $y(0)$ and $z(0)$ are normally distributed with mean and standard deviation equal to 4.0 and 1.0 respectively (i.e., $x(0), y(0), z(0) = \mathrm{N}(4.0,1.0)$). Objective of the data mining algorithms described in

this report is to identify and reconstruct such discontinuities. Given the description above we expect to generate a data set that can be partitioned into 8 clusters: one for each combination of the values of $x(0)$, $y(0)$ and $z(0)$.

By employing hierarchical clustering with DTW distance metrics (see Figure 48) we were able to obtain 7 clusters (see Figure 49 and Figure 50). Figure 51, Figure 52, Figure 53, Figure 54, Figure 55, Figure 56 and Figure 57 show the temporal profile and the histograms of the three stochastic variables for the scenarios contained in each of the 7 clusters.

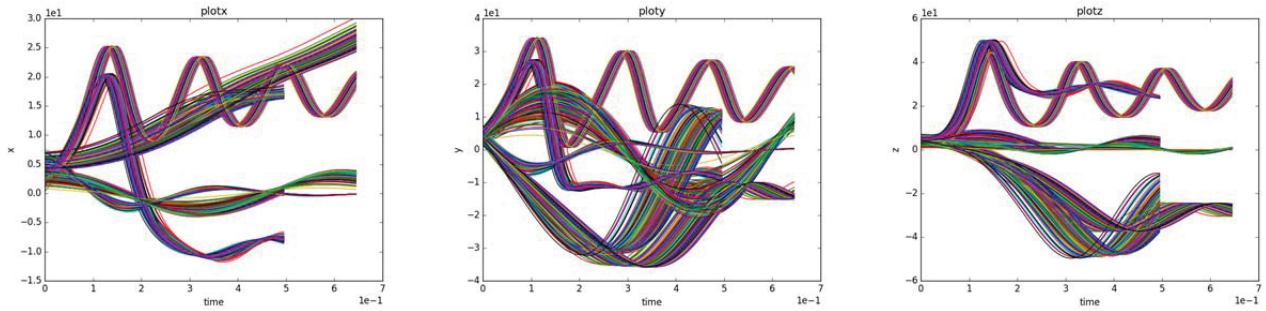

Figure 47. Data set generated by RAVEN containing multiple discountinuites and having variable time length projected in each of the three axis ($x(t)$, $y(t)$ and $z(t)$).

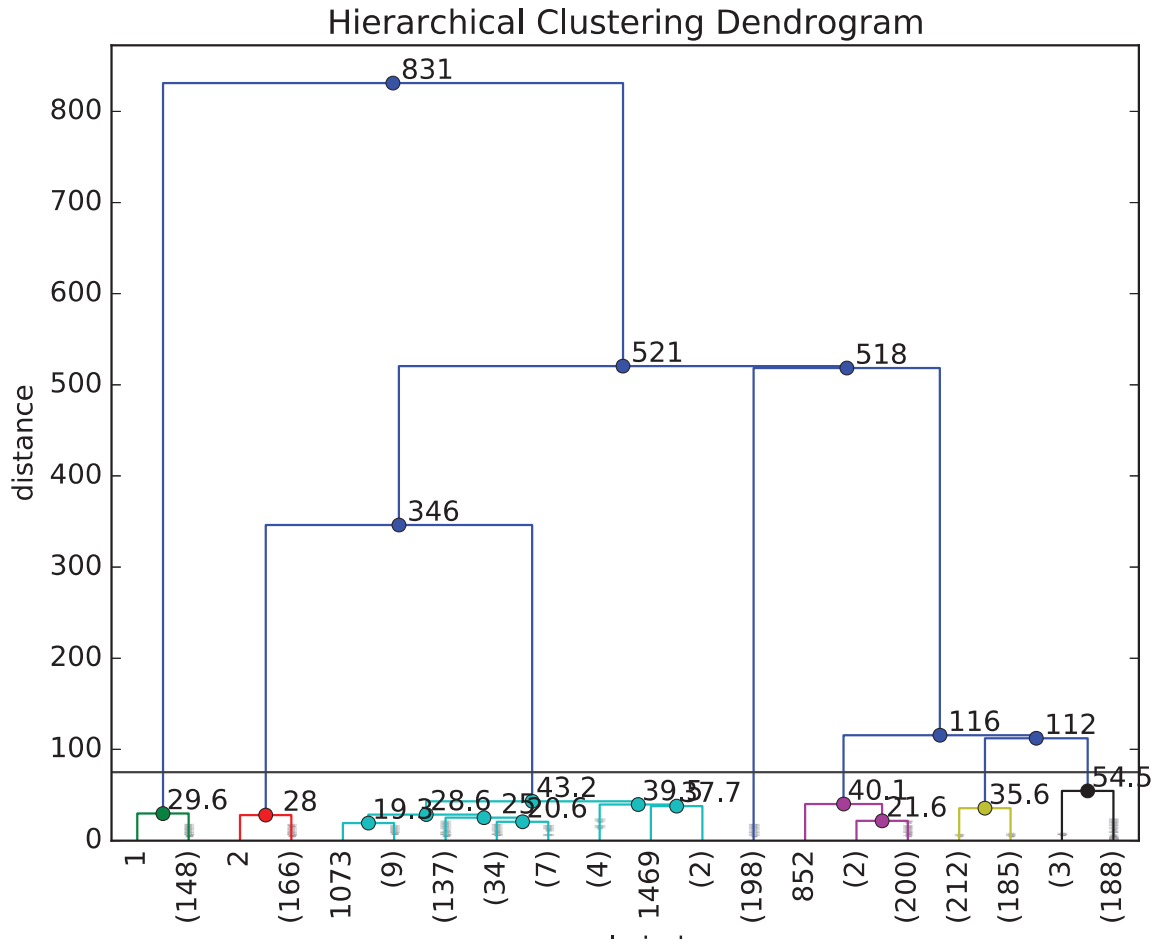

Figure 48. Dendrogram obtained using the RAVEN hierarchical algorithm for the data set shown in Figure 46.

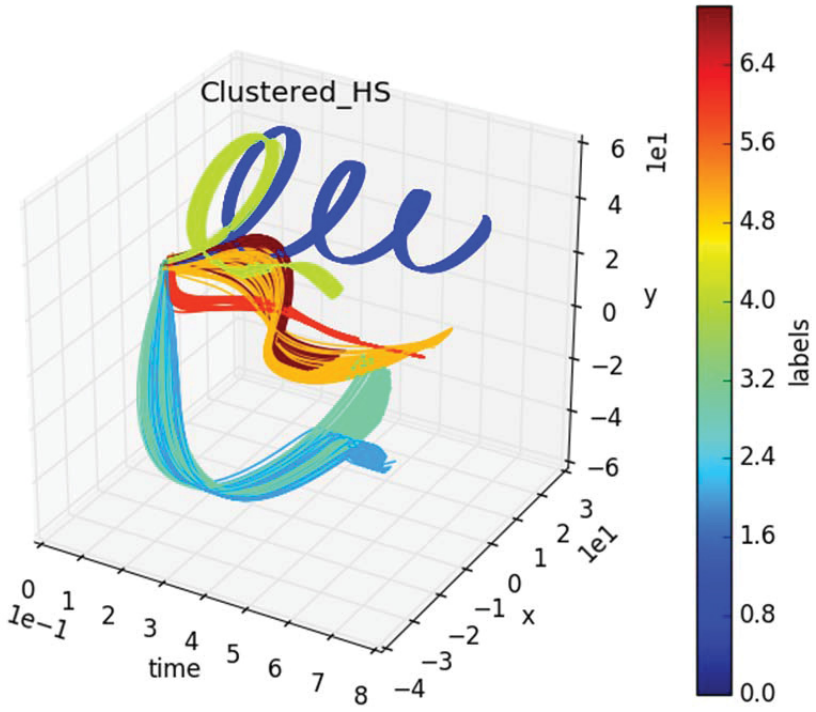

Figure 49. Plot of the clustered data set shown in Figure 46 colored by the cluster labels; each of the 9 clusters (from 1 through 9) correspond a color using Hierarchical algorithm (see Figure 48).

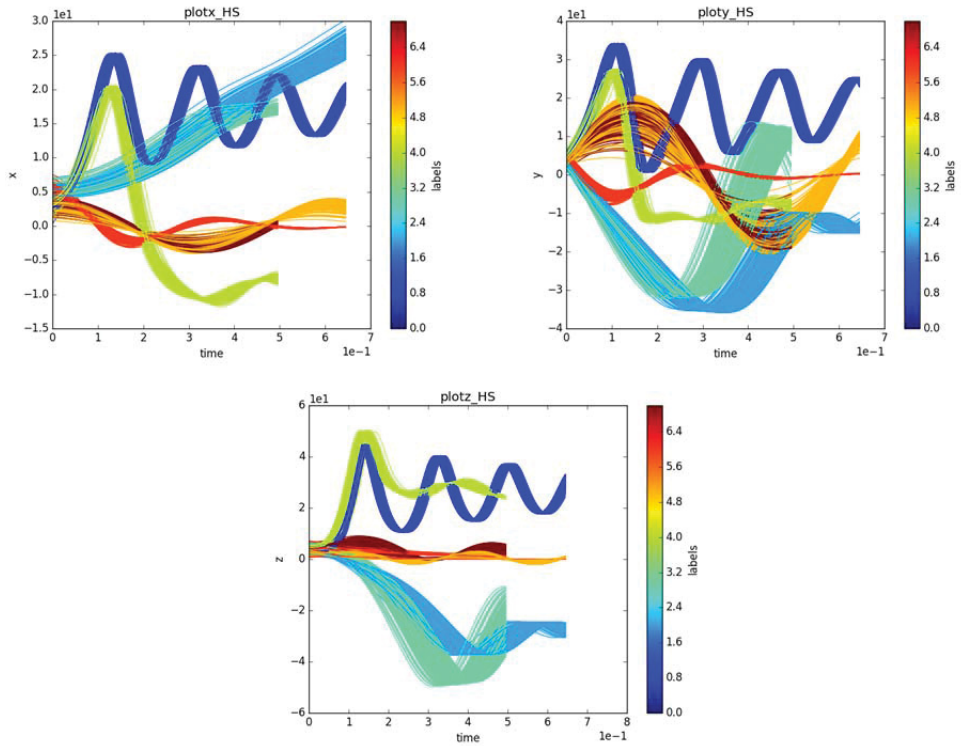

Figure 50. Plot of the clustered data set shown in Figure 46 colored by the cluster labels; each of the 9 clusters (from 1 through 9) correspond a color using Hierarchical algorithm (see Figure 48) projected in each of the three axis ($x(t)$, $y(t)$ and $z(t)$); compare with Figure 47.

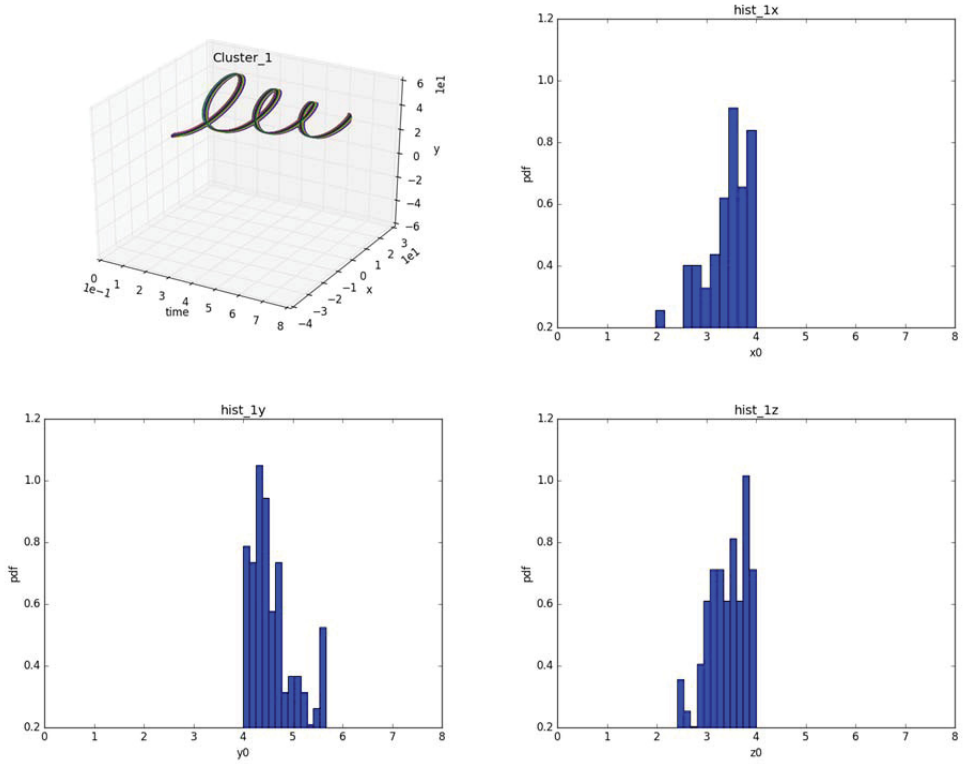

Figure 51. Analytical model analysis - Cluster 1 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

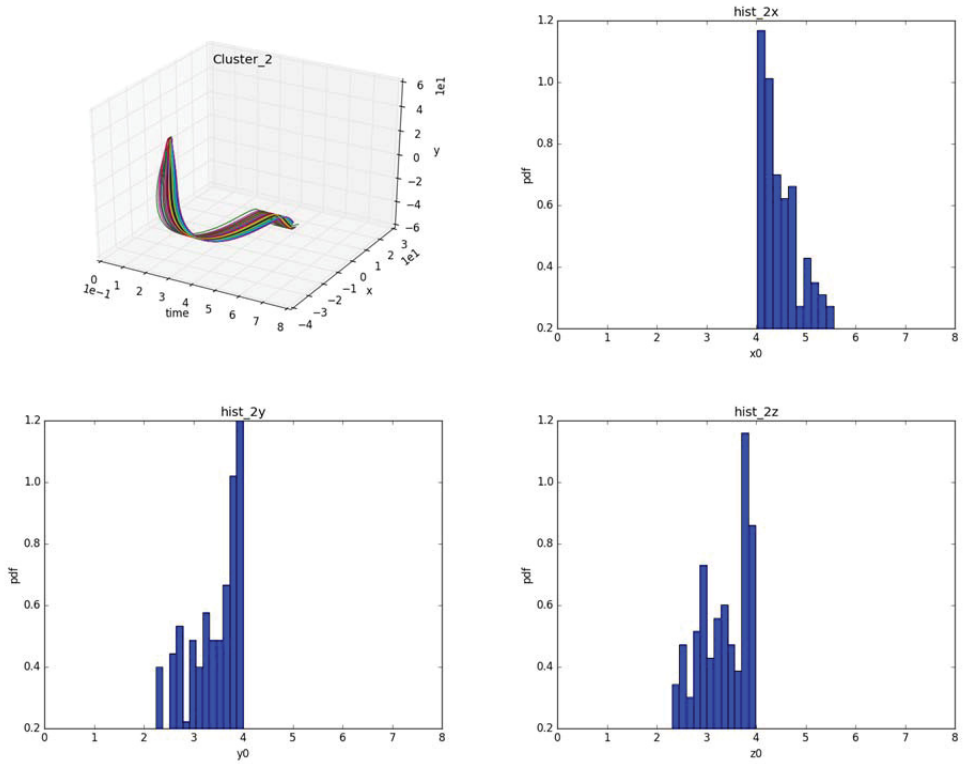

Figure 52. Analytical model analysis - Cluster 2 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

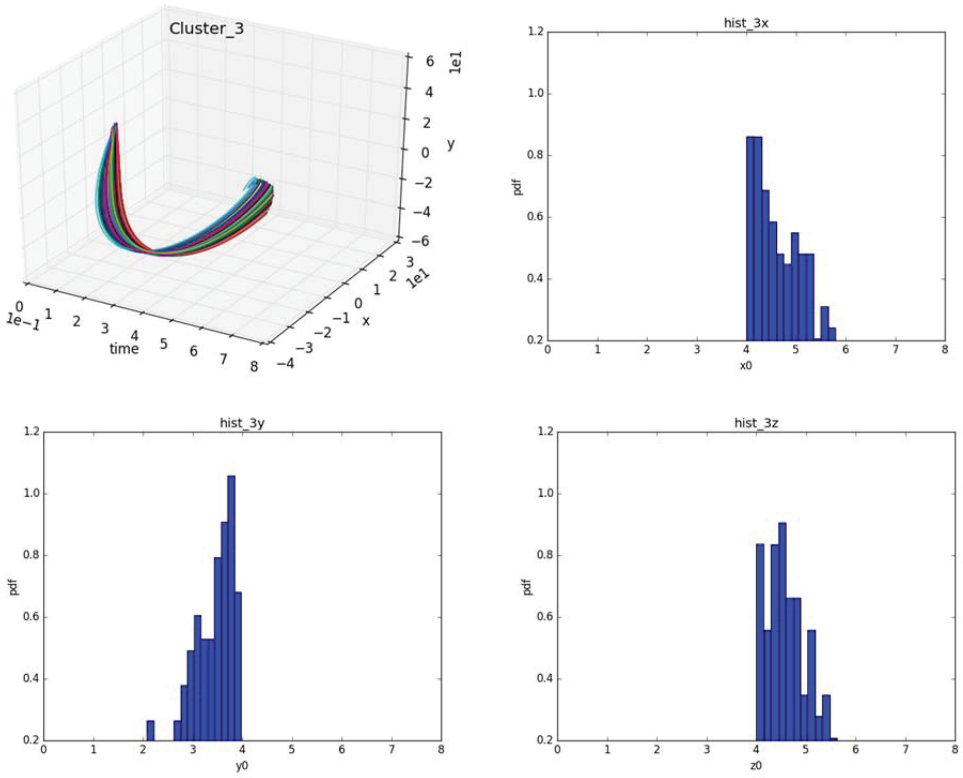

Figure 53. Analytical model analysis - Cluster 3 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

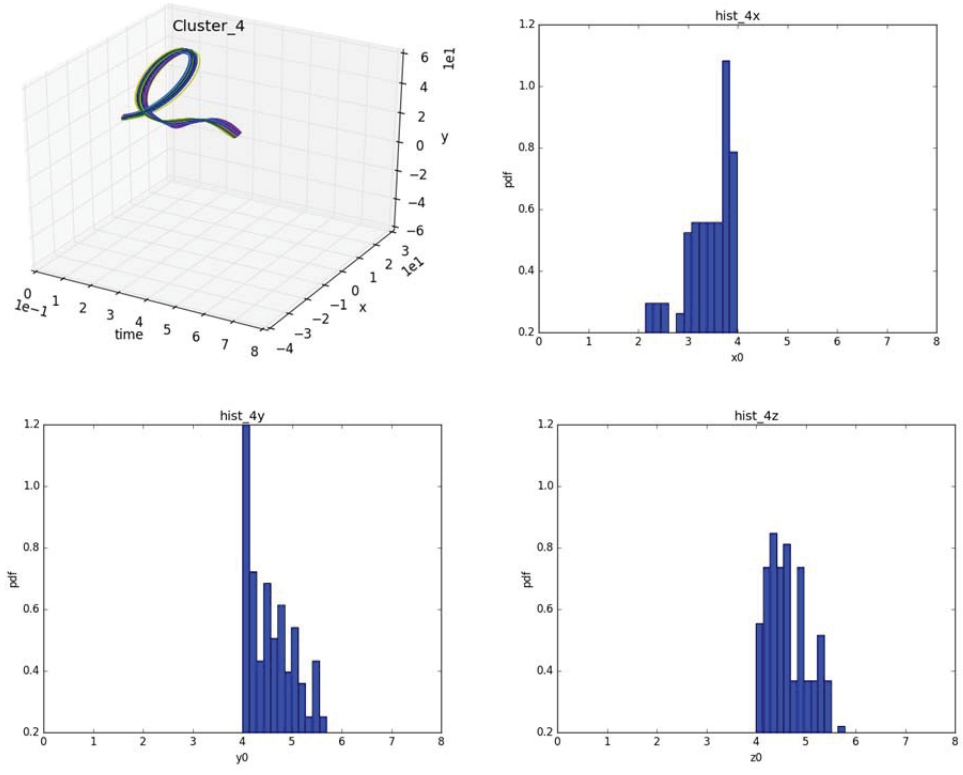

Figure 54. Analytical model analysis - Cluster 4 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

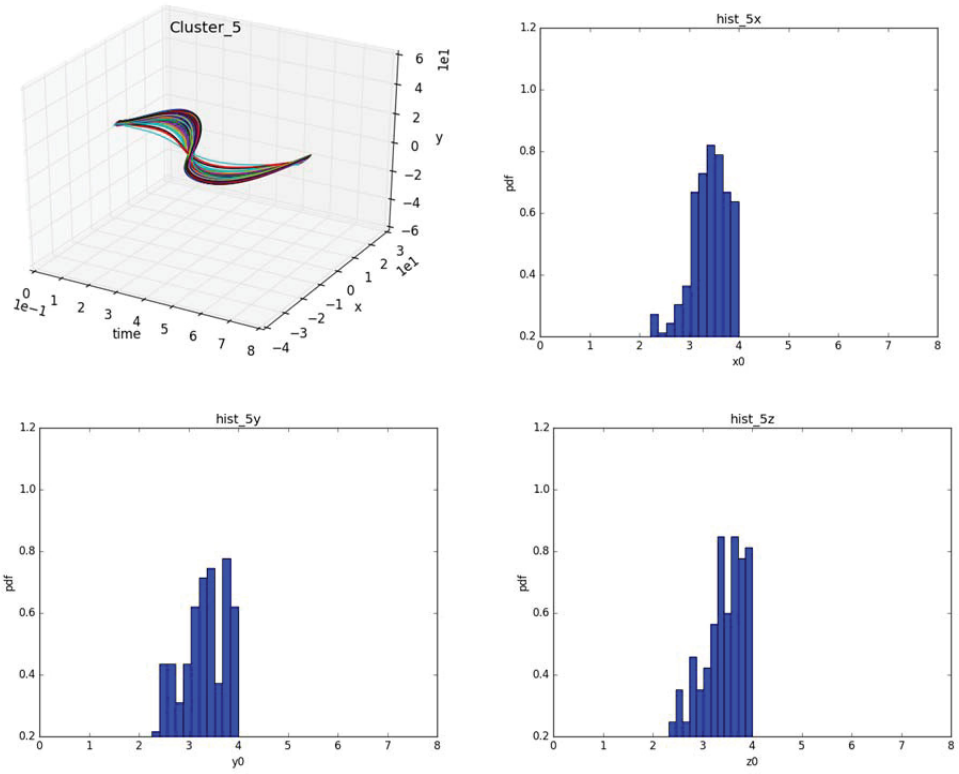

Figure 55. Analytical model analysis - Cluster 5 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

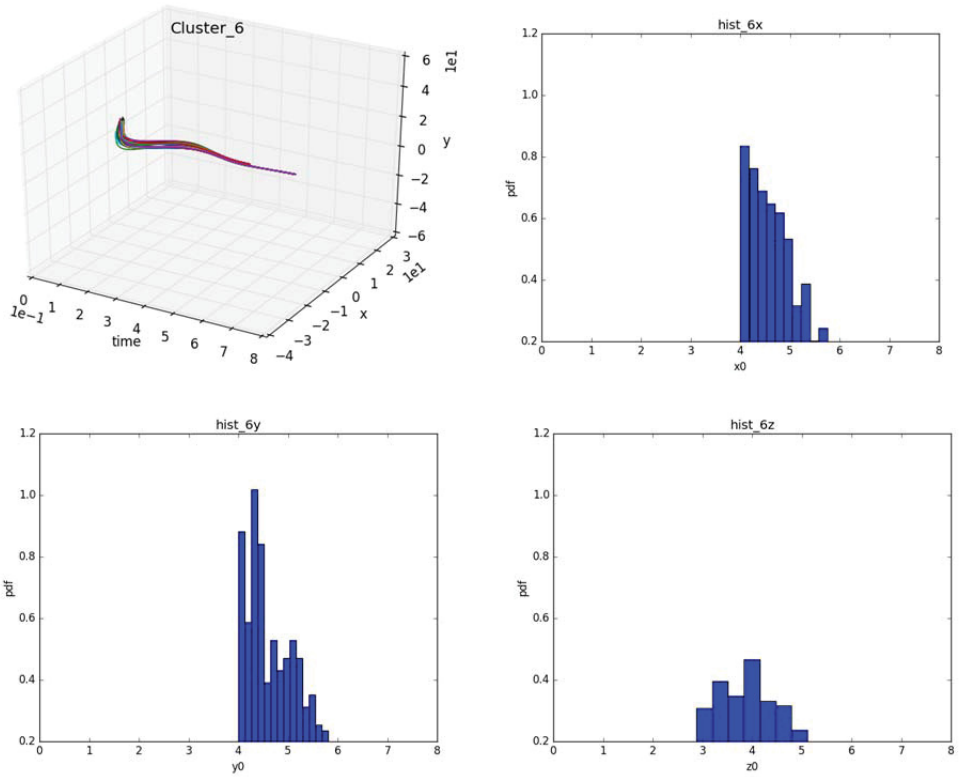

Figure 56. Analytical model analysis - Cluster 6 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

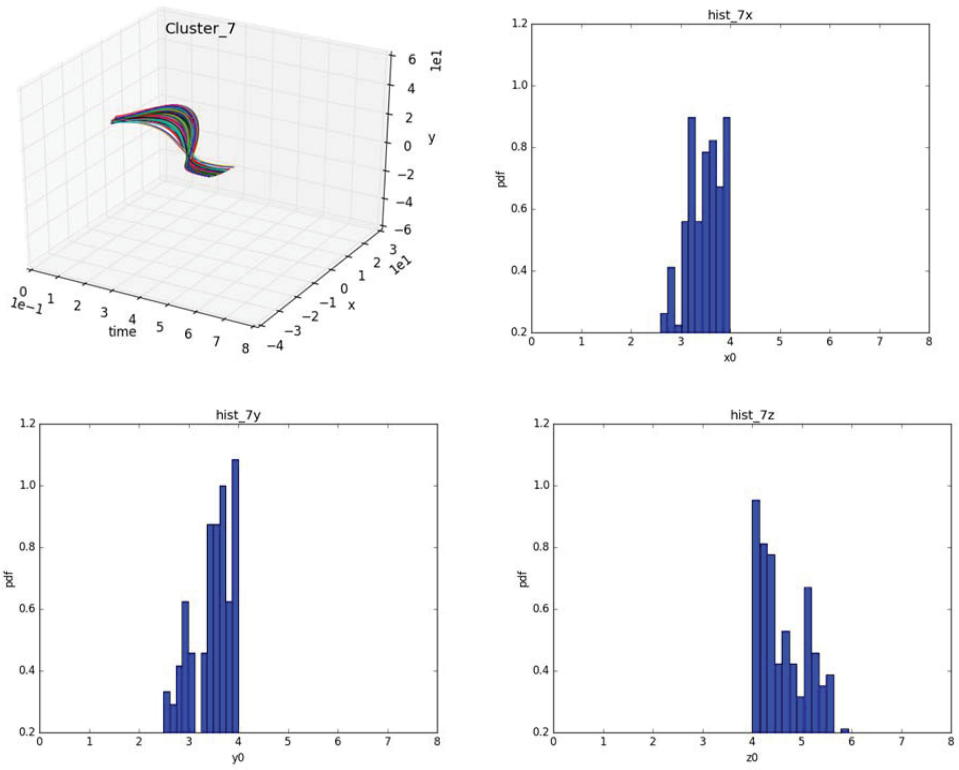

Figure 57. Analytical model analysis - Cluster 7 information: time series and hostrograms for the sampled values ($x(0)$, $y(0)$ and $z(0)$).

Given the description of the actual model wee were expecting 8 groups of scenarios depending the values of $x(0)$, $y(0)$ and $z(0)$. In order to investigate such results we mapped the expected with the obtained results in the following tables. Note that Cluster 6 incorporates these case that we initially thought would create different scenarios: $x(0), y(0) > 4$ independently of $z(0)$.

Table 2. Analytical test case: excepted vs. obtained clusters.

$z(0) < 4$		$x(0) < 4$	$x(0) > 4$
	$y(0) < 4$	Cluster 5	Cluster 2
	$y(0) > 4$	Cluster 1	Cluster 6

$z(0) > 4$		$x(0) < 4$	$x(0) > 4$
	$y(0) < 4$	Cluster 7	Cluster 3
	$y(0) > 4$	Cluster 4	Cluster 6

7.2.1 Analysis Summary

In summary, using the analytical model data set we were able to gather the following information:

- The hierarchical algorithm was able to clearly partition the original time series by considering the full temporal profile of the time series

- The obtained clusters (i.e., 7 clusters) are a subset of the expected obtained clusters (i.e., 8 clusters): this is due to the fact that the impact of $z(0)$ is negligible when $x(0) > 4$ and $y(0) > 4$ on the temporal profile of the simulations (i.e., cluster 6 comprise two clusters that were originally expected to identify). However, as predicted the impact of $z(0)$ is relevant for the other cases

7.3 Pump Controller

The second test case considers a pump controller model for a hypothetical simplified PWR model (see Figure 58) which consists of the following components:

- Reactor core (RX)
- Motor operated pump
- Pump digital controller
- Heat exchanger (HX)

This system is responsible to remove the decay heat generated from the core (RX) in order to avoid damage of the core itself. The objective is to maintain the temperature in the reactor core between 1500 C and 1600 C. The top-events are thus the following:

- **Success**: final temperature between 1500 C and 1600 C
- **Failure-high**: final temperature greater than 3000 C
- **Failure-low**: final temperature lower than 1600 C

While we assumed that both the HS and the pump are perfectly reliable components (i.e., no failure can be introduced), using [33] as a references, the digital pump controller reliability model has been performed using a continuous time Markov Chain formulation.

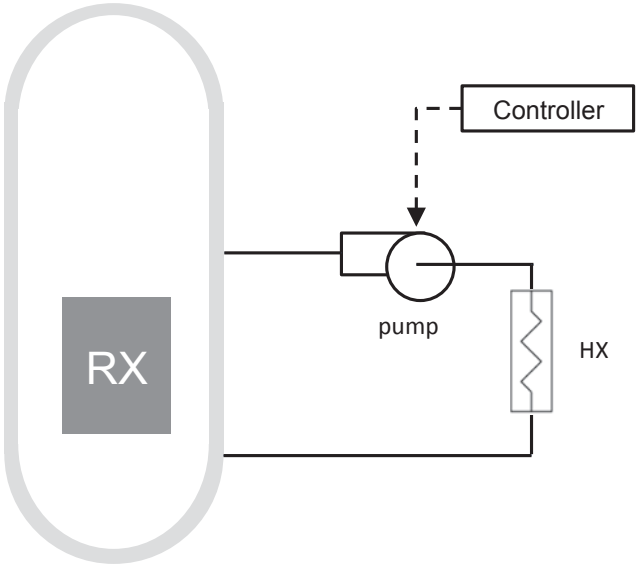

Figure 58. Pump controller test case: scheme of the system condiered.

In more detail, the controller has been modeled using 4 states (Figure 59):

- **State 0 - Operating**: controller operating as designed
- **State 1 - Failed closed**: controller failed by sending close signal to pump (i.e., pump not running)
- **State 2 - Failed stuck**: controller failed by sending oldest valid signal to pump

- **State 3 - Failed random**: controller failed by sending close signal to pump

For the scope of this report we have assumed a constant (in time) failure rate λ for all three transitions shown in Figure 59. The scope of this exercise is to identify the impact of both timing and type of failure on the system dynamics.

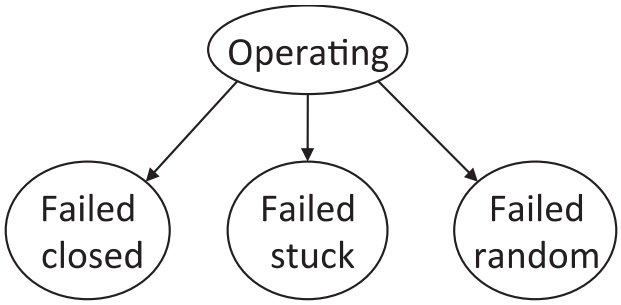

Figure 59. Continuos time Markov model for the pump controller.

In order to perform such analysis the model has been coded as a RAVEN external model (i.e., Python based code) which determine the temporal profile of core temperature give the two stochastic parameters:

- Pump controller failure time
- Pump controller failure mode

The dynamic of the system has been modeled using basic mass and energy conservation laws so no effective engineering conclusions can be gathered by this example. An example of scenario where no pump failure occurs is shown in Figure 60. Note that the transient has been divided along the temporal axis in 5 regions where:

- Pump is ON in regions 1, 3 and 5
- Pump is OFF in regions 2 and 4

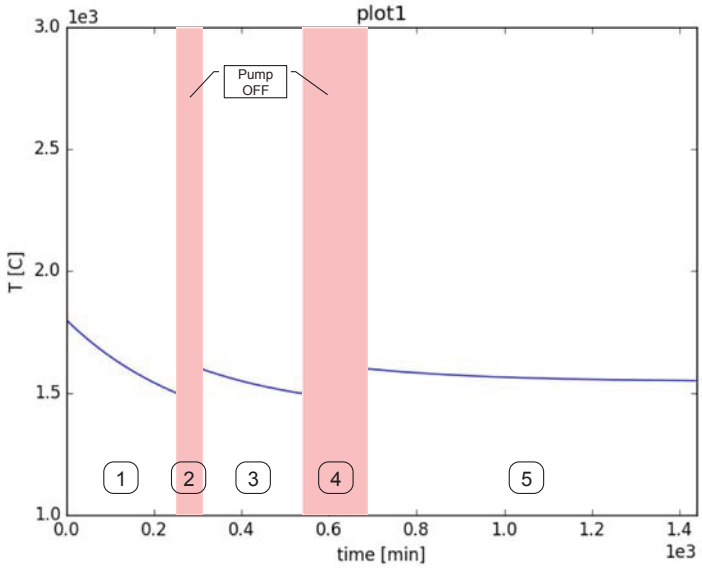

Figure 60. Pump controller: example of scenario where no pump failure occurs.

By using RAVEN we sampled the two stochastic parameters (see the histograms of these variables in Figure 62) using a Monte-Carlo algorithm and generated 1500 simulations as shown in Figure 61.

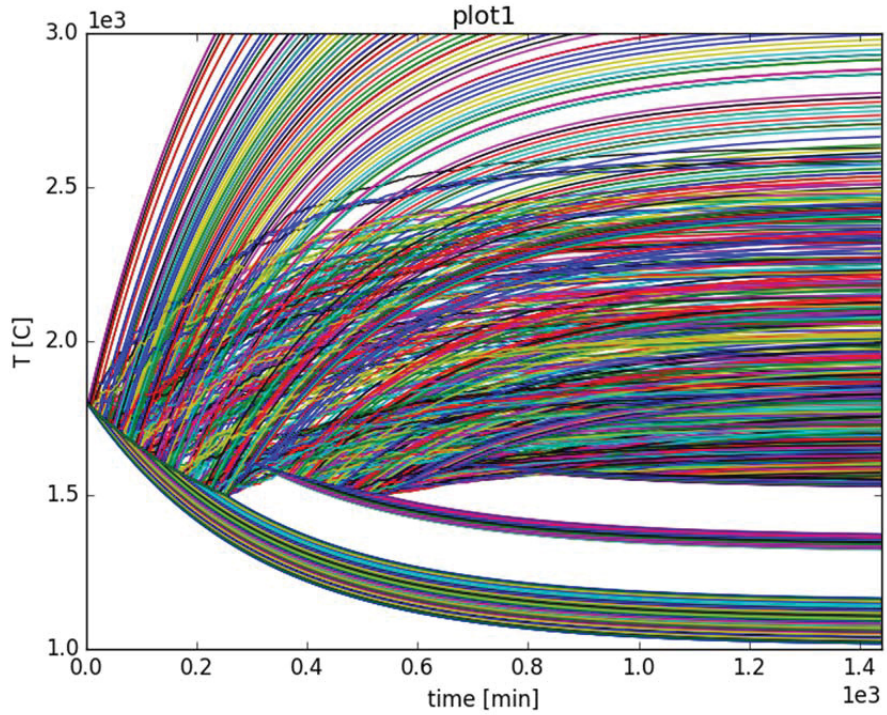

Figure 61. Pump controller: plot of the 1500 histories generated by RAVEN.

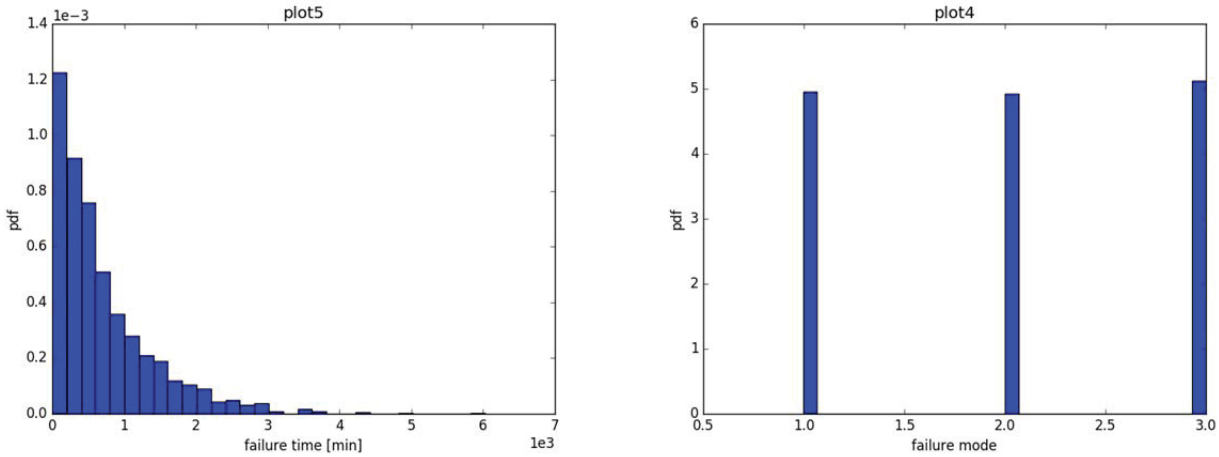

Figure 62. Histograms of the two stochastic variables, i.e., failure time (left) and failure mode (right) generated in the Monte-Carlo sampling process.

By observing only static values such as max or final temperature (see Figure 63) it is not really possible to extract valuable information from the data set.

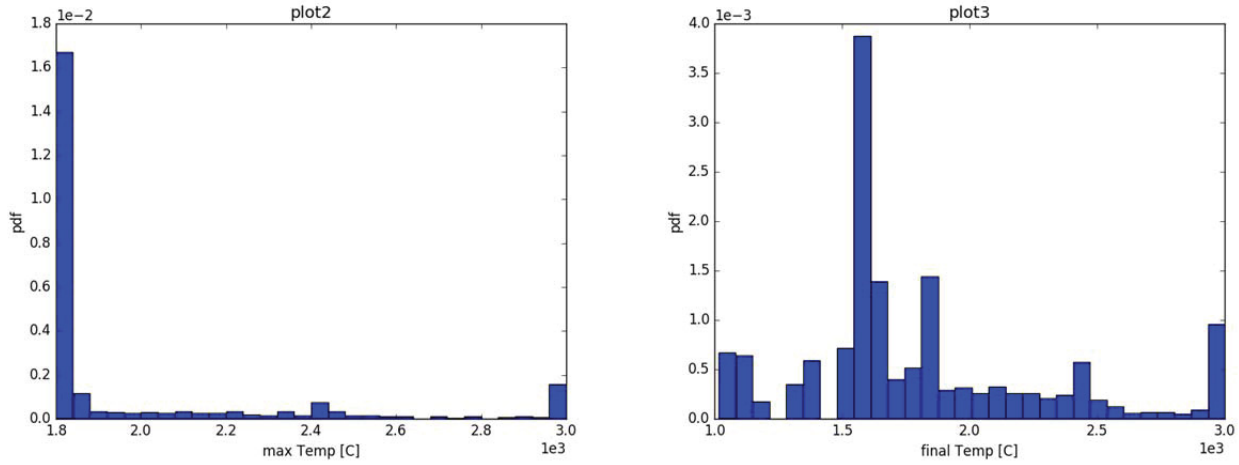

Figure 63. Histograms of max (left) and final (right) temperature of the simulations shown in Figure 61.

For the analysis of this dataset we have chosen to use hierarchical clustering using Euclidean distance as distance metrics. The dendrogram obtained is shown in Figure 64. From this dendrogram it is clearly possible to identify 3 clusters which lead us to choose a separation level equal to 10; the plot of the scenarios colored by the clustering label value is shown in Figure 65.

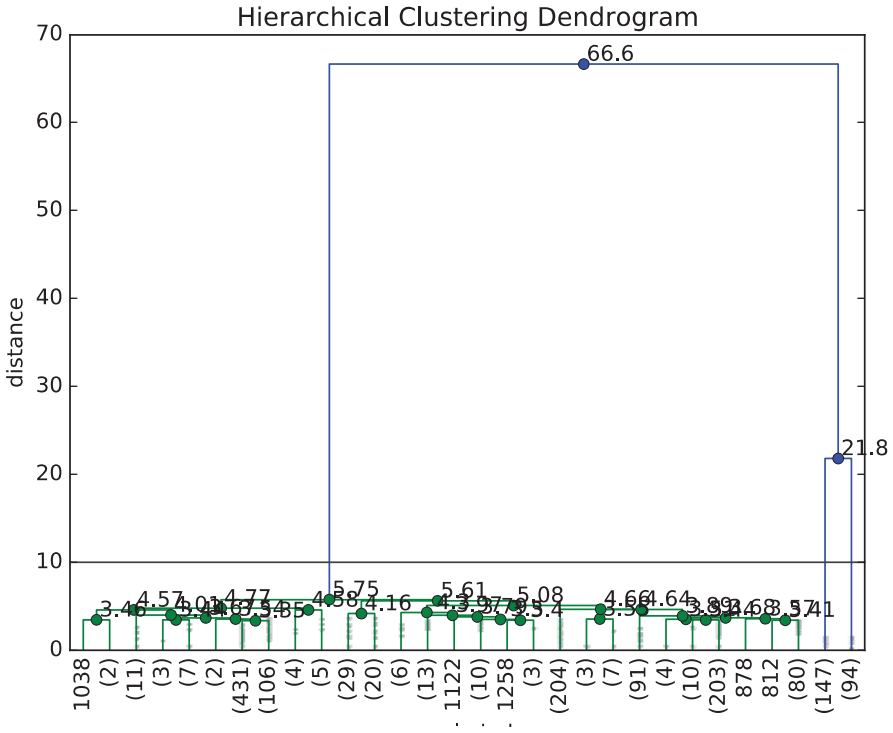

Figure 64. Dendogram obtained uisng hierarchical clustering (euclidean distance) for the dataset shown in Figure 61.

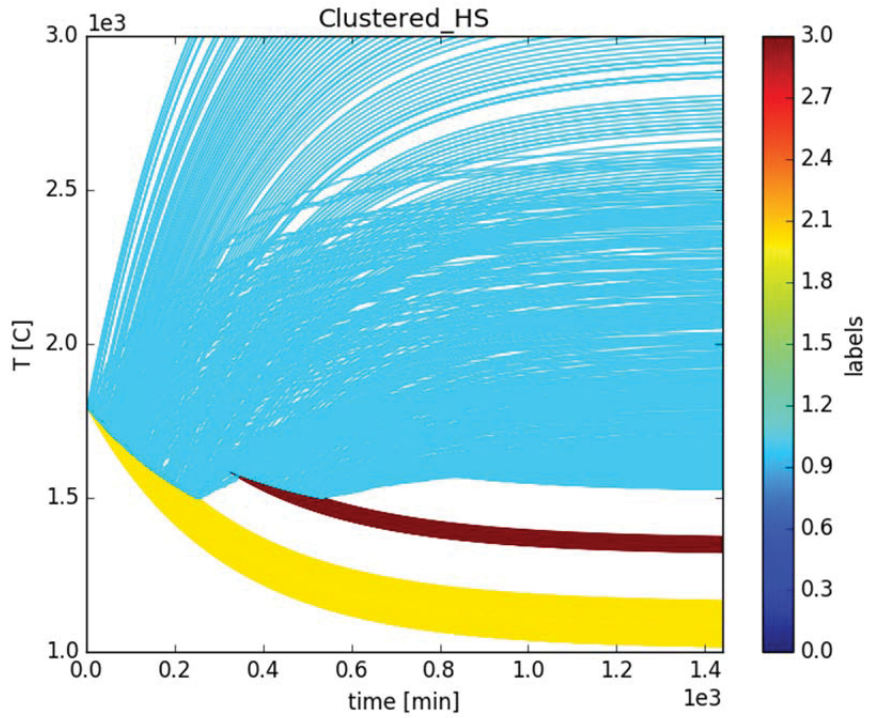

Figure 65. Plot of the 1500 histories generated by RAVEN (see Figure 61) colored based on the labels assigned by the hierarchical clustering (see Figure 64).

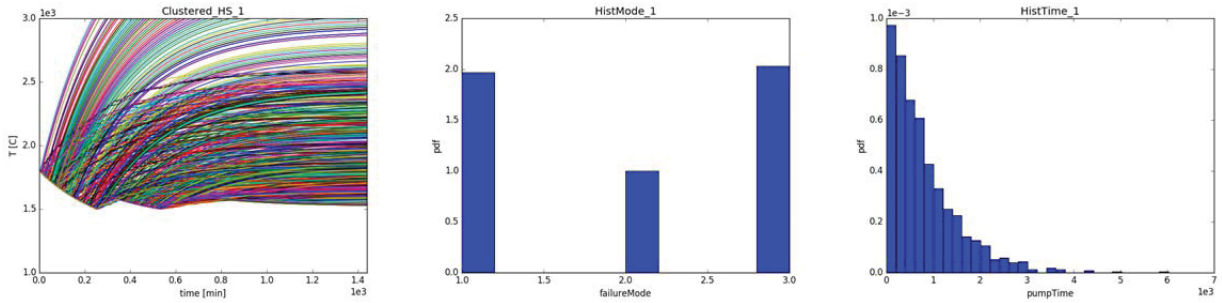

Figure 66. Cluster 1 (see Figure 64): plot of the histories (left), histograms of failure mode (center) and failure time (right).

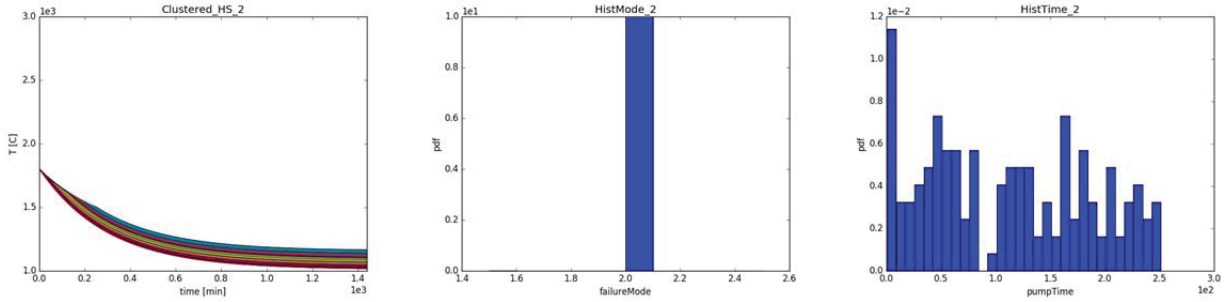

Figure 67. Cluster 2 (see Figure 64): plot of the histories (left), histograms of failure mode (center) and failure time (right).

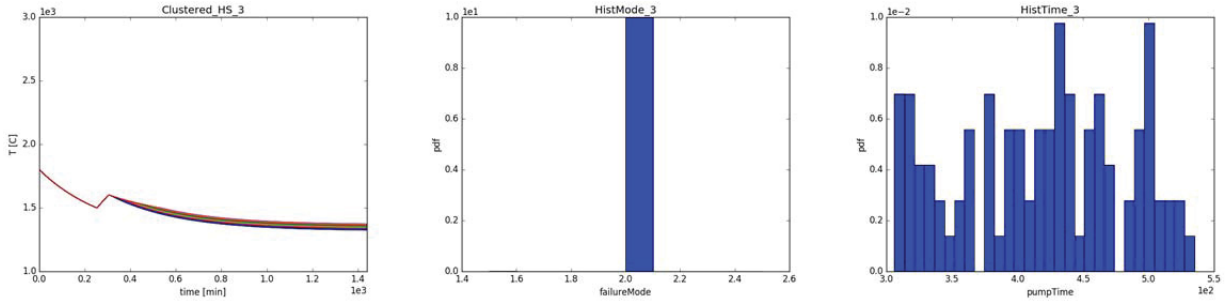

Figure 68. Cluster 3 (see Figure 64): plot of the histories (left), histograms of failure mode (center) and failure time (right).

The analysis of the obtained clusters is summarized in Figure 66 (Cluster 1), Figure 67 (Cluster 2), Figure 68 (Cluster 3). In order to describe the obtained results we refer also to Figure 60:

- Cluster 1 (see Figure 66) contains a large number of scenarios that lead to both "failure-high" and "success" top-events and no other particular information can be deduced

- Cluster 2 (see Figure 67) and 3 (see Figure 68) contain scenarios where pump controller in State 2 (i.e., stuck) while the pump was actually ON in region 1 and 3 respectively (Figure 60). They are all leading to the "failure-low" top-event

Given the fact that cluster 1 contains too much variety of scenarios, we performed hierarchical clustering on just the scenarios belonging to Cluster 1: i.e., sub-clustering. The obtained dendrogram is shown in Figure 69. In this case, 5 clusters were obtained; the plot of the scenarios colored by the clustering label value is shown in Figure 70.

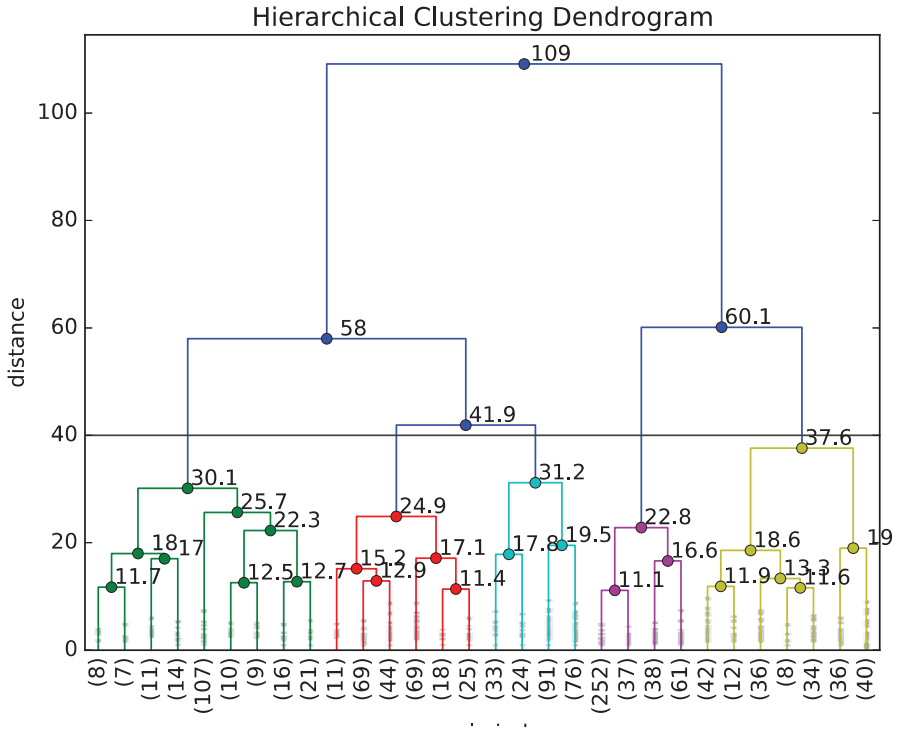

Figure 69. Dendogram obtained using hierarchical clustering (euclidean distance) for the Cluster 1 dataset shown in Figure 66.

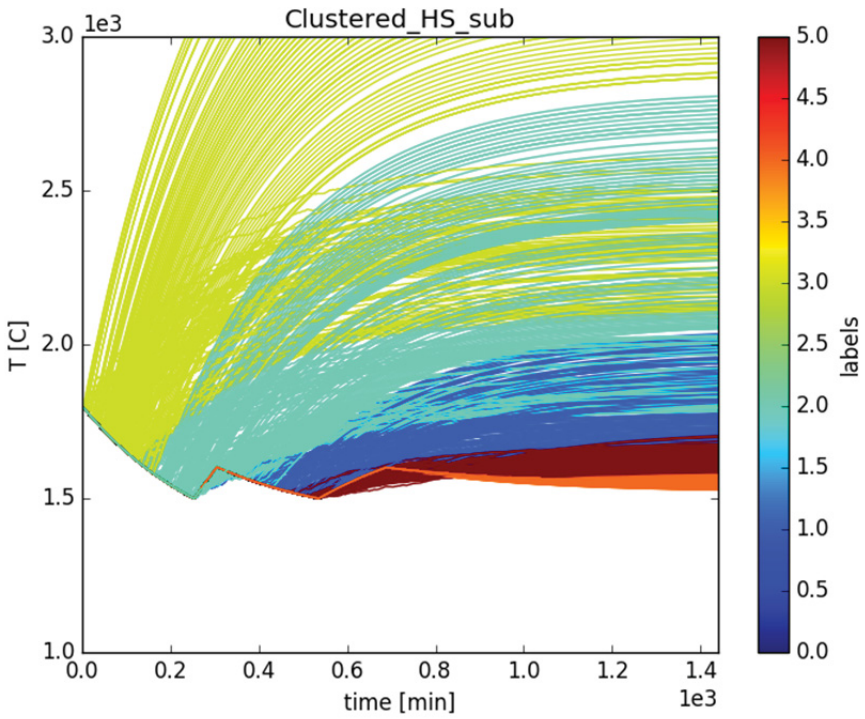

Figure 70 Plot of the histories belonging to Cluster 1 (see Figure 66) colored based on the labels assigned by the hierarchical clustering (see Figure 69).

The analysis of the obtained clusters is summarized in Figure 71 (Cluster 1), Figure 72 (Cluster 2), Figure 73 (Cluster 3), Figure 74 (Cluster 4) and Figure 75 (Cluster 5):

- Clusters 1, 2, 4 and 5 (see Figure 71, Figure 72, Figure 74 and Figure 75 respectively) contain scenarios where pump controller failed in any of the three modes (closed, stuck and random) but did not lead to the failure-high top event

- Cluster 3 (see Figure 73) contains scenarios where pump controller failed in only two modes (closed and random). Note that:

    - Pump failed in the first time region

    - Controller failure random did not lead failure-high top event; however, controller failure closed lead to very high core temperatures (including 3000 C)

  This cluster contains scenarios leading to failure-high and success top events. Note that if controller failure occurs prior to about 115 min that the simulation leads to failure-high top event.

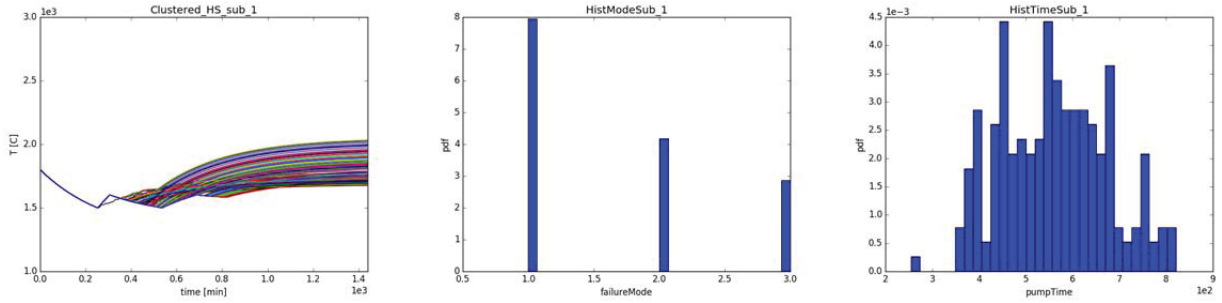

Figure 71. Cluster 1 (see Figure 70): plot of the histories (left), histograms of failure mode (center) and failure time (right).

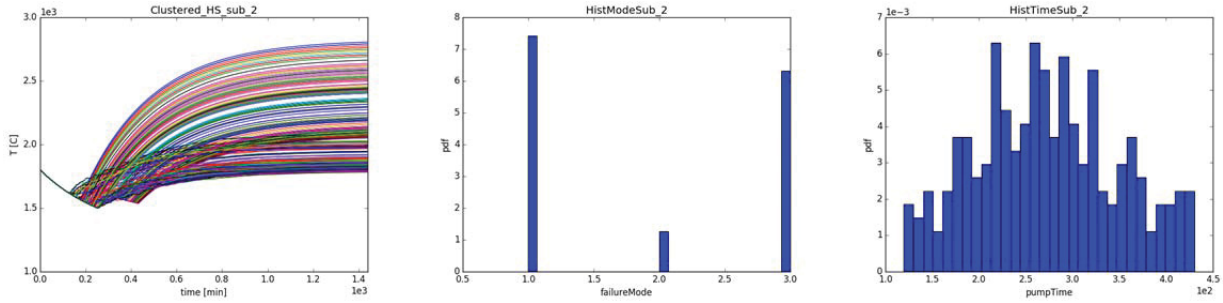

Figure 72. Cluster 2 (see Figure 70): plot of the histories (left), histograms of failure mode (center) and failure time (right).

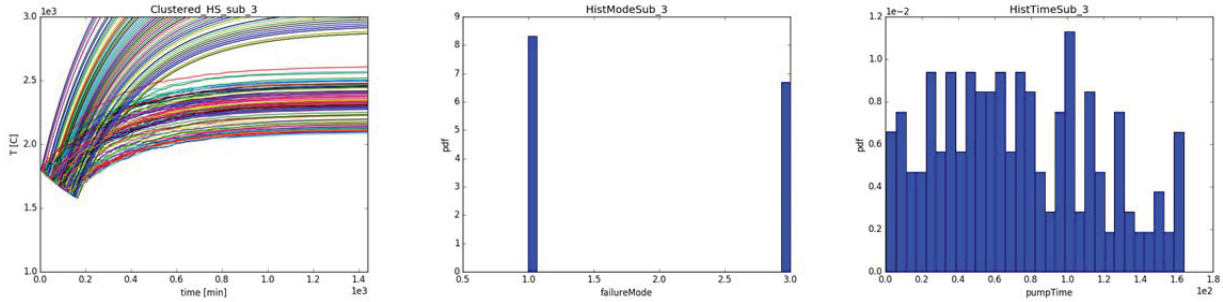

Figure 73. Cluster 3 (see Figure 70): plot of the histories (left), histograms of failure mode (center) and failure time (right).

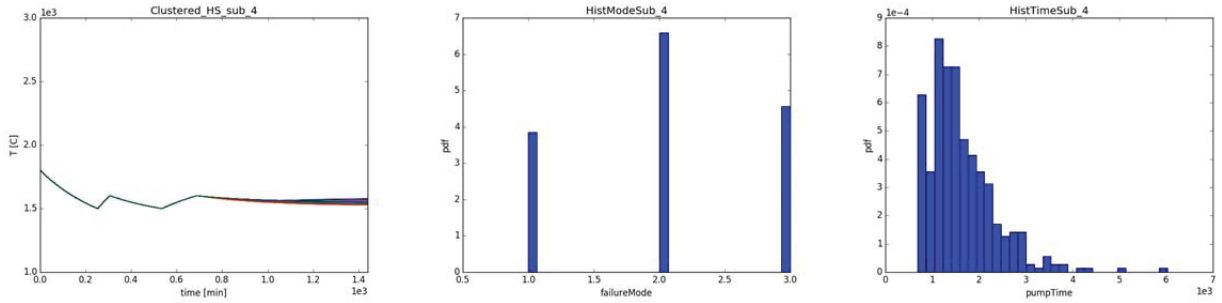

Figure 74. Cluster 4 (see Figure 70): plot of the histories (left), histograms of failure mode (center) and failure time (right).

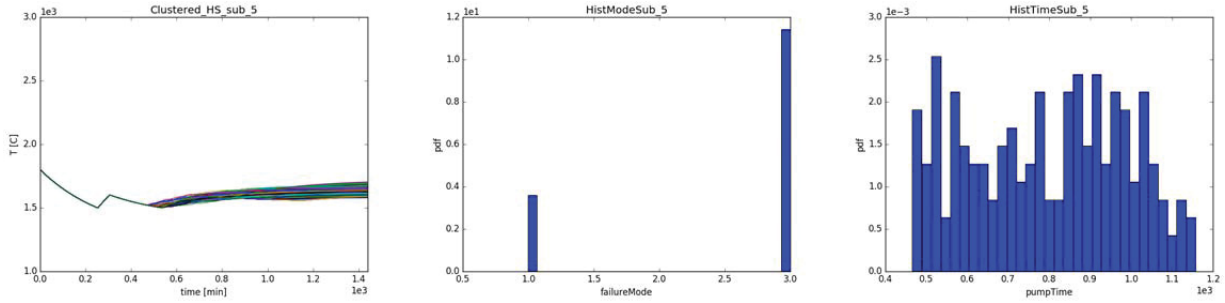

Figure 75. Cluster 5 (see Figure 70): plot of the histories (left), histograms of failure mode (center) and failure time (right).

7.3.1 Analysis Summary

In summary, using the analytical model data set we were able to gather the following information:

- Failure-high top event can be reached if controller failure to state 1 (i.e., failure closed) occurs prior to 115 min
- Failure-low top event can be reached if controller failure to state 2 (i.e., failure stuck) occurs in the time regions 1 and 3
- Controller failure to state 3 (i.e., failure random) do not lead to any failure top event independently of the controller failure time
- For these controller failure events the core temperatures are between 1500 C and 3000 C:
    - Controller failure to state 1 (i.e., failure closed) after to 115 min
    - Controller failure to state 2 (i.e., stuck) in time regions 2, 4 and 5

7.4 PWR Station Black-Out

In this study a Pressurized Water Reactor has been considered to analyze the evolution of a Station Black-Out (SBO) accident scenario. The system and the scenario have been simulated with MAAP5. Therefore, a dedicated MAAP5 input file has been built in order to model the sequence of events desired which consist in some procedures [34] to be adopted and which are shown in Figure 76. First procedure to be taken into account is the ECA-0-0, that considers some steps up to power recovery. When power is finally recovered then procedure ECA-0-1 or ECA-0-2 are entered according to some plant conditions.

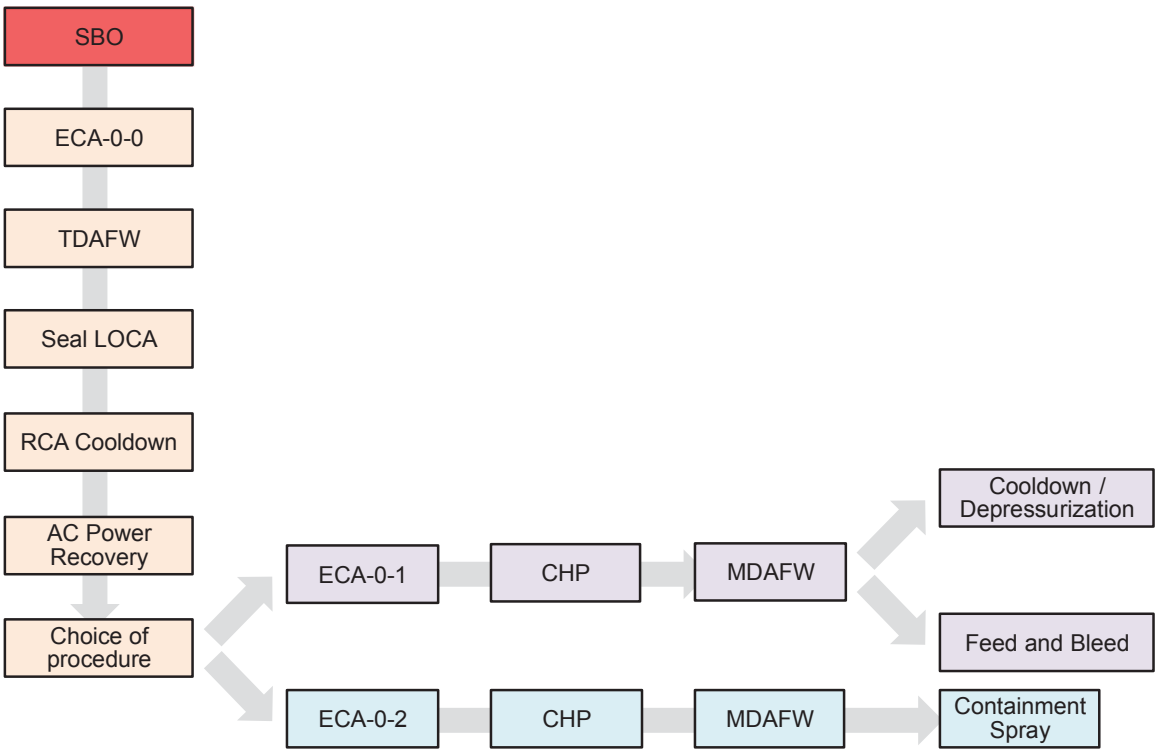

Figure 76. Scheme of the PWR Station Black-Out accident scenario.

The Dynamic Event Tree (DET) [15] analysis consists in creating a control logic within the Thermal-Hydraulic code simulating the scenario. This is performed by adding a series of trigger that are activated every time that a branching condition is met, creating consequently a set of branch. A DET analysis can be performed with coupling code RAVEN – MAAP5 as here shown. In particular, RAVEN drives the DET by creating, basing on the user defined conditions, the different branches every time that the simulation run is stopped, while MAAP5 run the TH simulation of the scenario modeled in the input each time that a new branch is created.

7.4.1 SBO Accident Scenario

The scenario modeled in the MAAP5 input is as follows:

- **ECA-0-0 procedure**
    - Initiating Event (t = 0): Station BlackOut (SBO);
    - t = 360 s: Turbine –Driven Auxiliary FeedWater (TDAFW) is called on demand;
    - t = 780 s: Seal Loss of Coolant Accident occurs (LOCA);
    - t = 1800 s: Depressurization of Steam Generators (SGs) to cooldown RCS;
    - Power Recovery can occur anytime during the simulation;
    - t = $T_{power\ recovery}$ + 630 s: Choice for the next procedure to be adopted. If:
        - Reactor Cooling System (RCS) subcooling < 25°F AND Pressurizer level < 14%
        - Safety Injection (SI) is required (High Pressure Injection or Containment Spray)
        - ECA-0-2 is required; otherwise ECA-0-1 is implemented.

- **ECA-0-1 ($t = T_{ECA-0-1}$)**
  - $t = T_{ECA-0-1}$ + 750 s: Charging Pump (CHP) is switched on ;
  - $t = T_{ECA-0-1}$ + 900 s: Motor-Driven Auxiliary FeedWater (MDAFW) is switched on (if the TDAFW has previously failed);
  - $t = T_{ECA-0-1}$ + 900 s, if the CHP works but no AFW is available: Feed and Bleed is implemented (pressurizer valves are open);
  - $t = = T_{ECA-0-1}$ + 1260 s and CHP failed: RCS cooldown is implemented:
  - when $T_{CORE,OUT}$ < 565 °F and pressurizer level < 50% m: cooldown is stopped and depressurization of primary circuit is performed;
  - if CHP and MDAFW both succeed to work than no other actions are taken.
  - In this condition, when requirements for entering ECA-0-2 are met, then ECA-0-2 procedure is entered.
- **ECA-0-2 ($t = T_{ECA-0-2}$)**
  - $t = T_{ECA-0-2}$ + 540 s: CHP is switched on;
  - $t = T_{ECA-0-2}$ + 630 s: MDAFW is switched on;
  - Containment sprays are on when the set point is triggered

7.4.2 DET Assumptions

For this scenario a DET has been generated. This DET is based on the following assumptions:

- AC Power Recovery Time follows a lognormal distribution with parameters $\lambda = 6.14$ and $k = 0.745$;
- DC battery failure is assumed to occur during the scenario according to a triangular distribution with min = 4h, max =6h and peak= 5h.

Branches occur in the DET according to the following trigger conditions:

- A branch occurs at t = 360 s from the beginning of the scenario to simulate the failure on demand of the turbine-driven auxiliary feedwater;
- At t = 780 s, multiple branches are considered in order to simulate different possible LOCA size: 21 gpm/RCP, 76 gpm/RCP, 182 gpm/RCP, 480 gpm/RCP are the four LOCAs leakage rates taken into account;
- When ECA-0-1 is entered and CHP is switched on, a branch occurs to consider the failure on demand of the CHP;
- Branching occur for the AC power recovery when the cumulative distribution function meets the thresholds: 0.05, 0.3, 0.4, 0.5, 0.7;
- Branch occurs for accounting DC failure when the corresponding cumulative distribution function meets thresholds equal to 0.1 and 0.9.

A total simulation time of 18 h = 64800 s has been considered. Simulation goes up to “core uncovery” or mission time. In this case, “Core uncovery” is the event corresponding to the core water level falling below the top of active fuel)

The DET simulation works as follows: a first simulation of MAAP5 is run. Every time that a branching condition is met, either expressed as probability or as value - (e.g. TDAFW is switched on), simulation run is stopped and a new set of simulations (branches) is spawned. The simulation goes on in each new branch up to a new branching condition is met.

Corresponding input parameters of this DET simulation are:

- TTAFW: turbine-driven auxiliary feedwater failure
- CHP: charging pump failure for ECA-0-1
- SEAL: seal LOCA occurrence
- DCOFF: DC batteries failure time
- ACREC: AC power recovery time

The simulation results in 126 branches simulated for a total of 54 completed histories. 7 out of 54 histories ends due to the occurrence of "core uncovery" that ranges from 5257 s to 12853 s. For the other 47 histories, "core uncovery" is not reached within the mission time. Time evolution of some variables of the system have been then plotted. Examples are shown in Figure 77, Figure 78 and Figure 79. The six output variable here considered are:

1. PSGGEN is the steam generator 1 pressure
2. WWBBN is the water flow through the loop 1 break
3. WWTOTFW is the total feed water flow
4. ZWDC2SG is the steam generator 1 down-comer water level;
5. MACUM is the mass of water in the accumulators tanks (total)
6. PPS is the primary system pressure
7. ZWCPS is the collapsed water level into the core

In addition, Figure 80 shows the histograms of the 5 input variables listed above. Note that since the DET strategy is basically a discrete event simulation sampling strategy the stochastic variables are sampled for specific finite values.

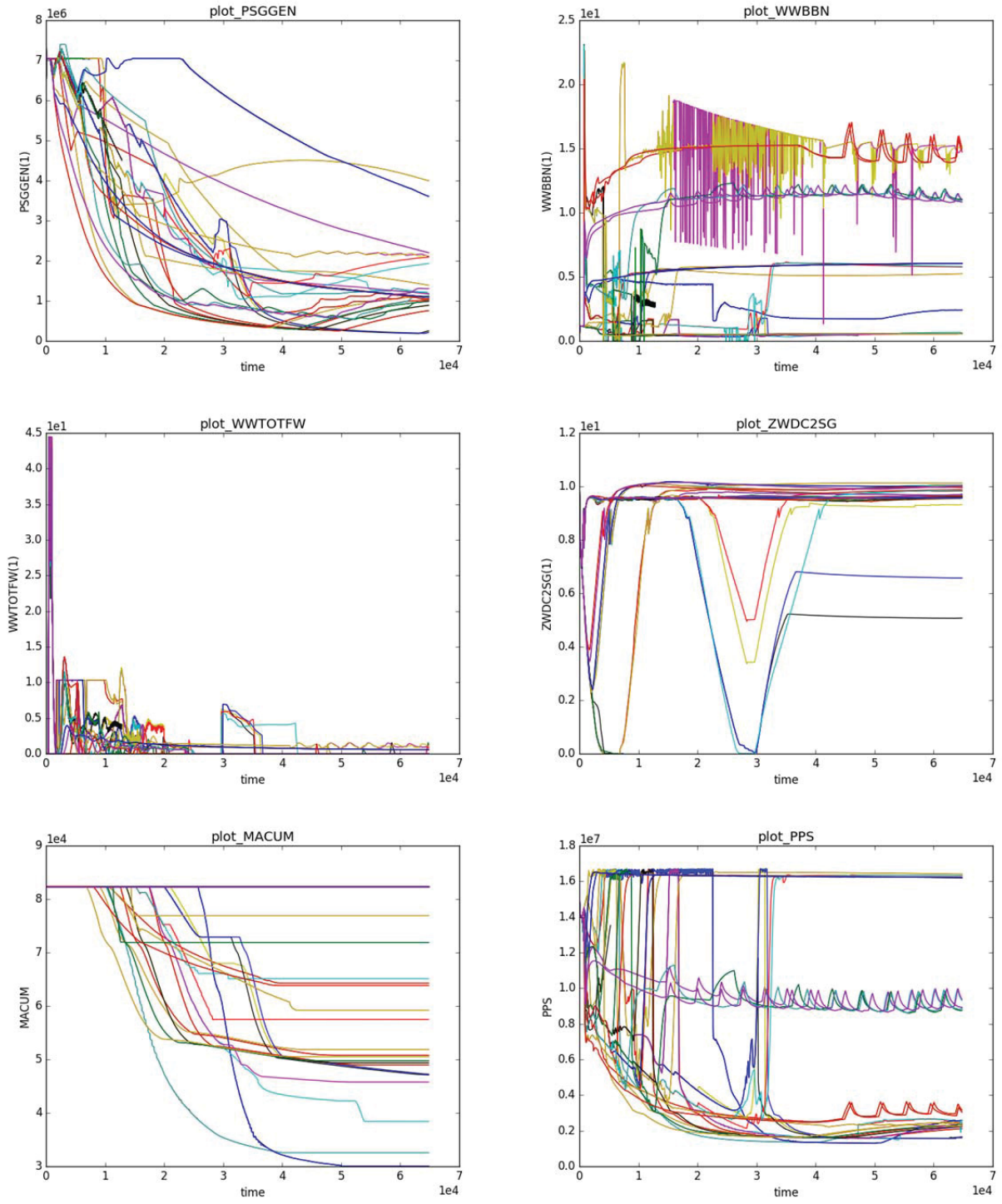

Figure 77. PWR Station Black-Out: plot of the output variables generated by RAVEN-MAAP (1).

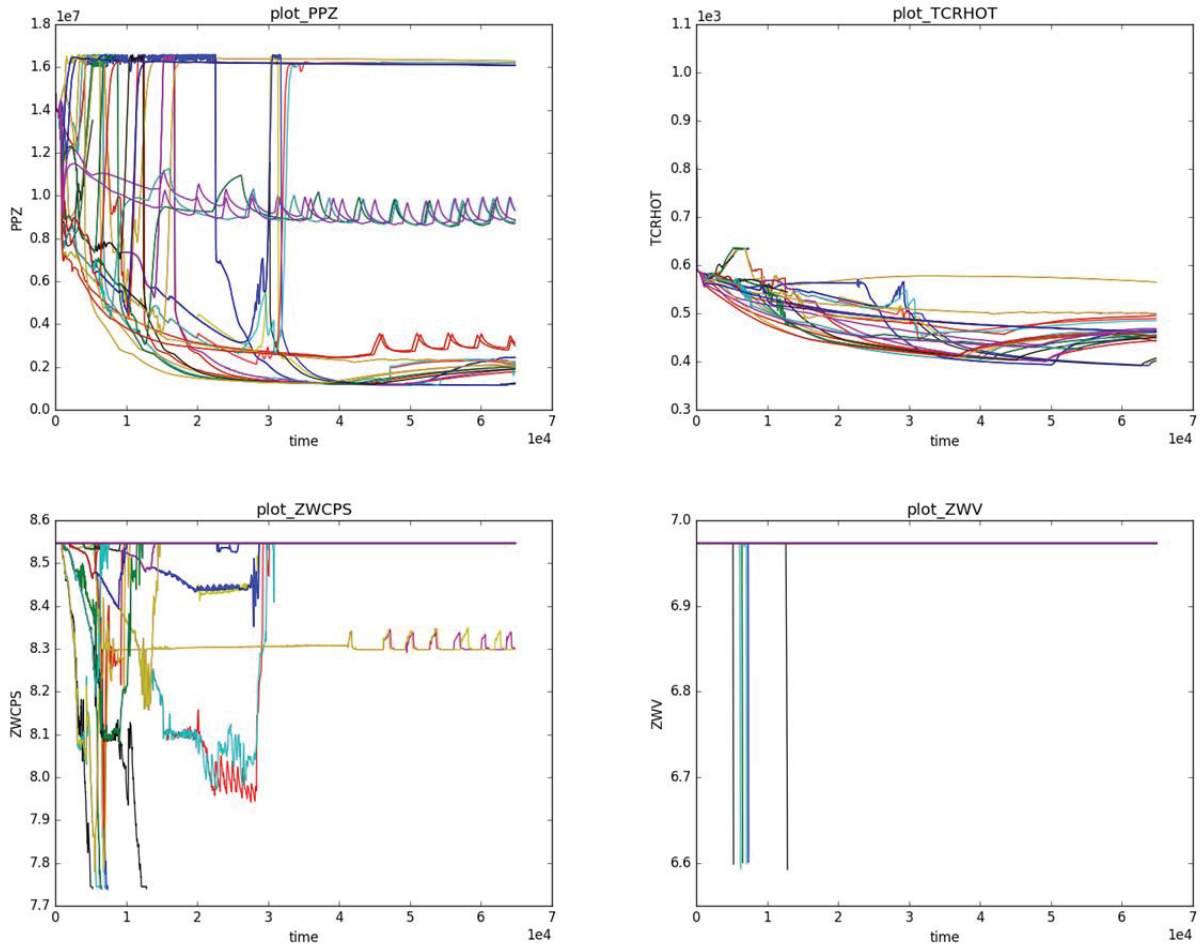

Figure 78. PWR Station Black-Out: plot of the output variables generated by RAVEN-MAAP (2).

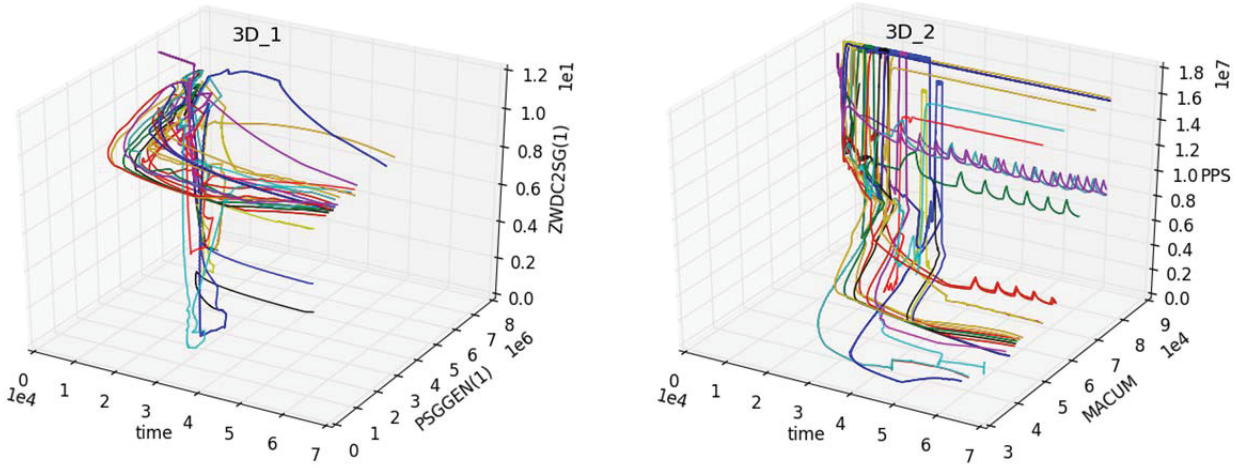

Figure 79. PWR Station Black-Out: 3D plot of the output variables generated by RAVEN-MAAP.

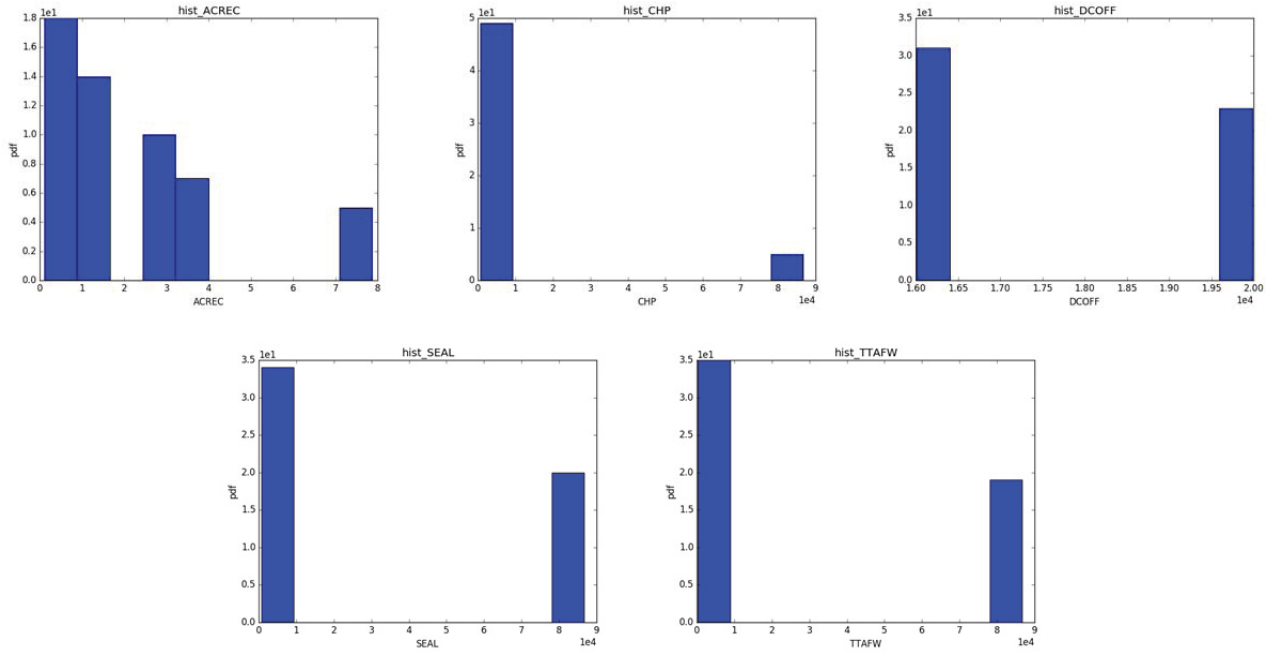

Figure 80. PWR Station Black-Out: histograms of the input variables sampled by RAVEN.

For this test case we chose to use a Mean-Shift clustering techniques (see Section 4.4.3) where we considered again the full temporal profile of the output variables. From this analysis we were able to obtain 6 clusters:

1. Cluster 1 (see Figure 81): this cluster contains scenarios characterized by early failure of the charging pump, early battery failure and a late turbine-driven auxiliary feed-water. This combination of events caused a late and rapid decrease in the reactor water level.

2. Cluster 2 (see Figure 82): this clusters contains scenarios characterized by early failure of the charging pump and small SEAL leakage. The combination of events lead to temporary drop of the water level within the core which is restored by the plant safety systems.

3. Cluster 3 (see Figure 83): this cluster contains a single scenario which is characterized by early failure of the charging pump, early battery failure and a late turbine-driven auxiliary feed-water. This is very similar to Cluster 1 except that the drop in the water level is much slower.

4. Cluster 4 (see Figure 84): this clusters contains scenarios characterized by early failure of the charging pump, early AC recovery and small SEAL leakage similar to Cluster 2. The combination of events lead to very quick drop of the water level within the core which is restored by the plant safety systems. The characteristic difference with Cluster 2 is that the primary system pressure reach a smaller steady state due to the fact that the operators in these scenarios are following ECA-0-2 procedure where feed and bleed is active.

5. Cluster 5 (see Figure 85): this clusters contains scenarios characterized by: a quick drop and recovery of core water level and a rapid drop in the primary system pressure. For some scenarios a quick recovery of system pressure and a second drop occurs. These scenarios are characterized by large seal LOCA and subsequently an early AC recovery after that ECA-0-1 is followed.

6. Cluster 6 (see Figure 86): this cluster contains a single scenario in which core water level slowly drops while system pressure quickly drops at about 50% its nominal value. Compared to cluster 3 which has a similar temporal behavior, it is caused by an early failure of the DC system and an early auxiliary feed-water failure.

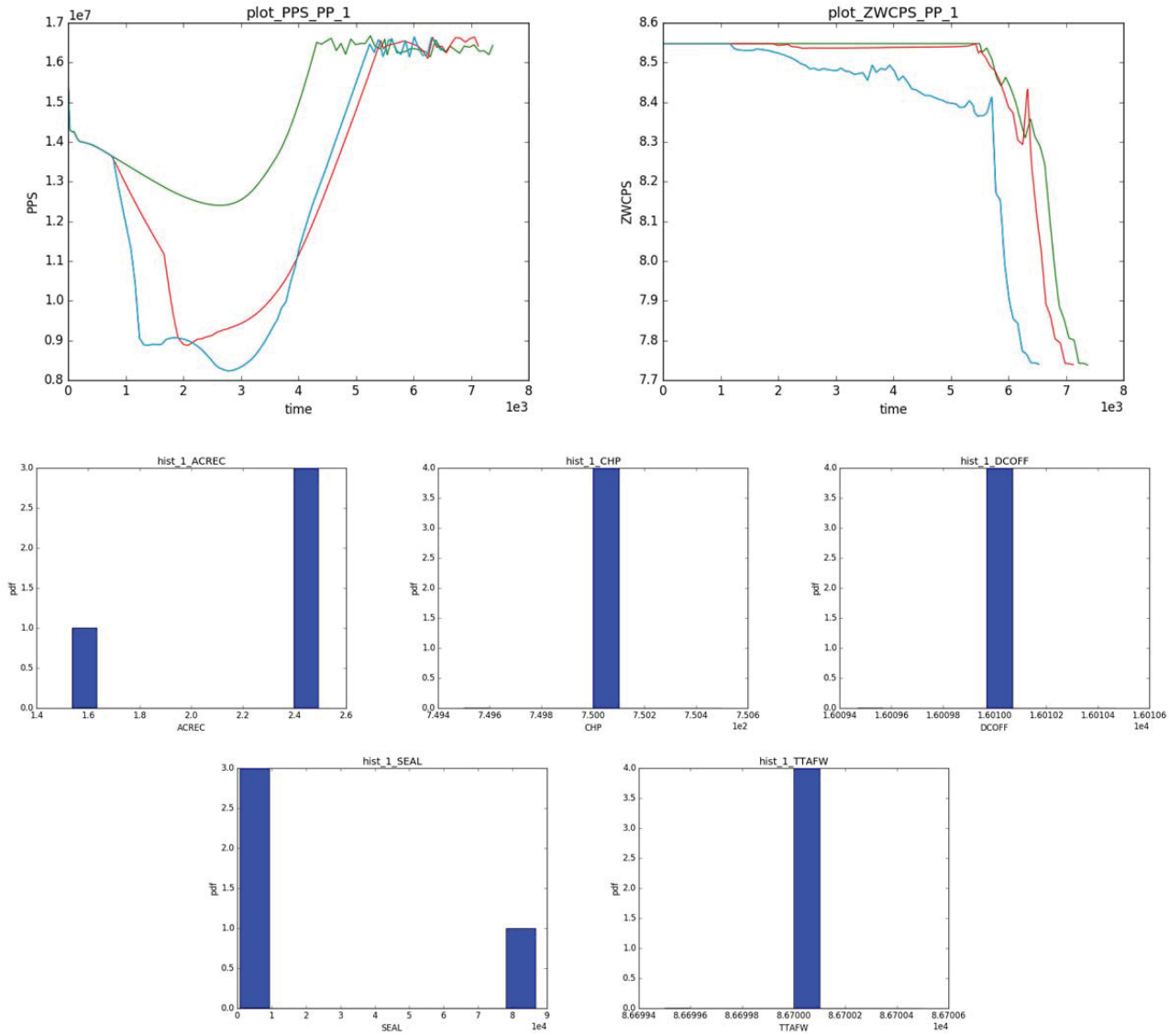

Figure 81. PWR Station Black-Out: analysis of Cluster 1.

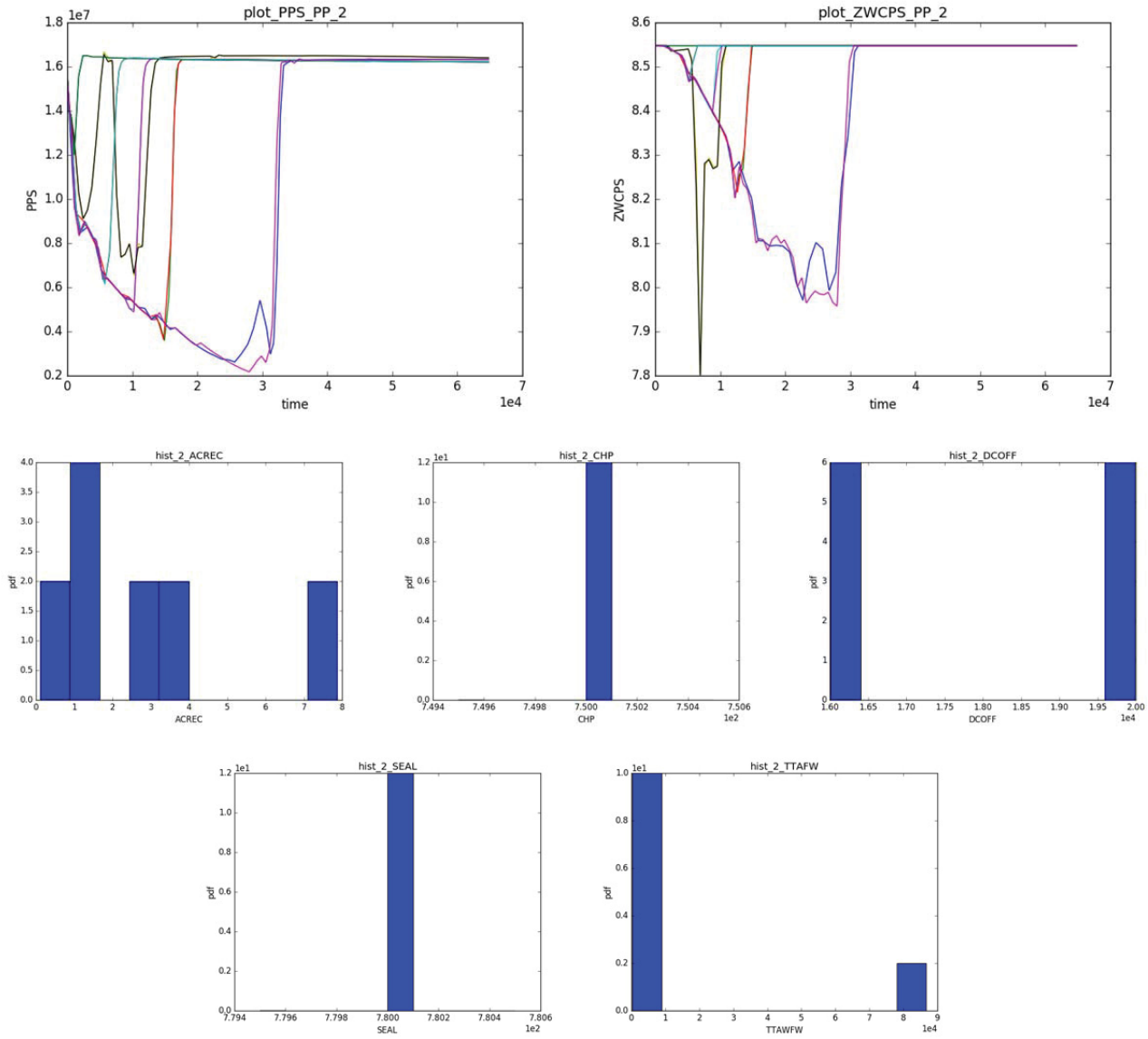

Figure 82. PWR Station Black-Out: analysis of Cluster 2.

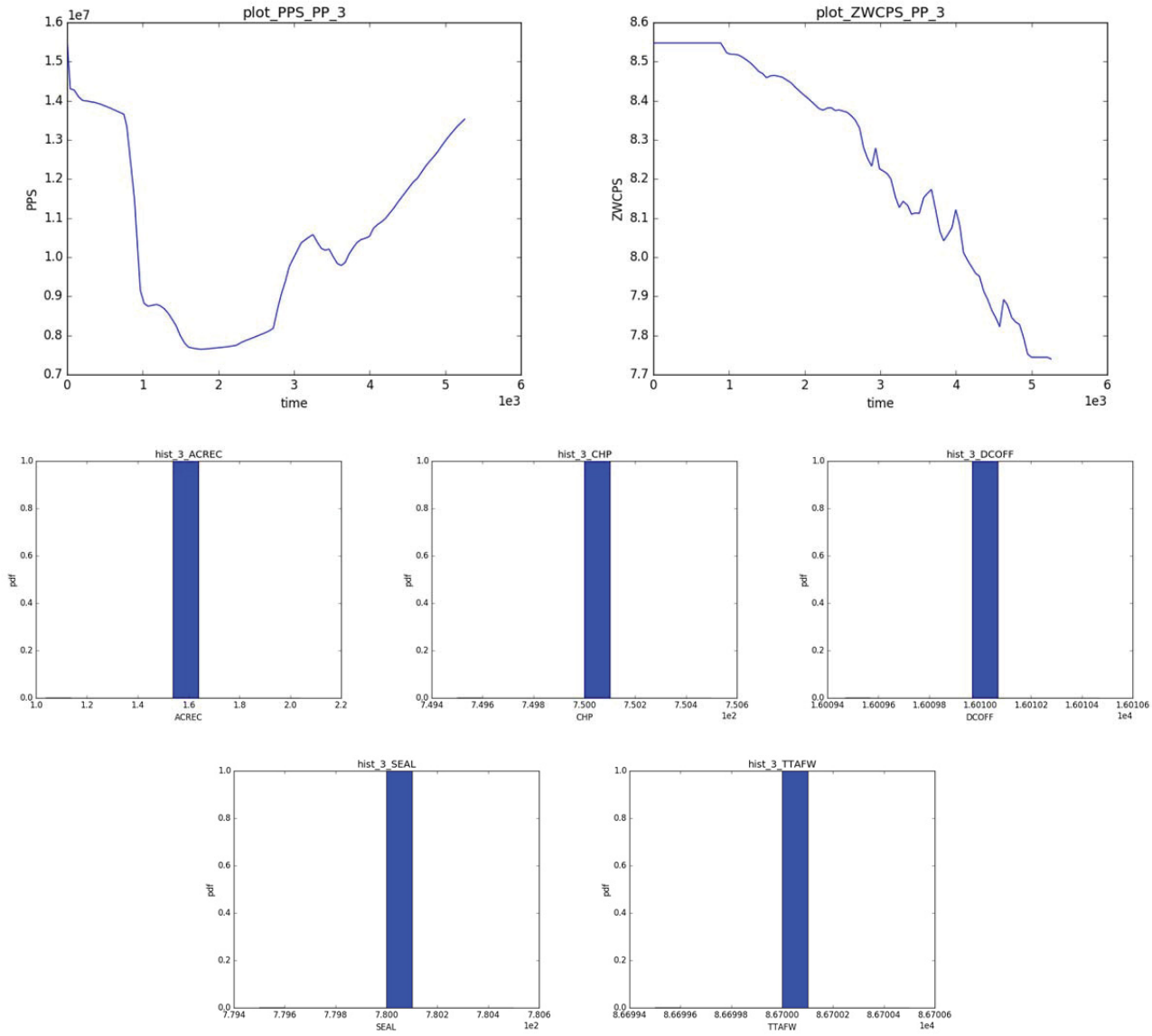

Figure 83. PWR Station Black-Out: analysis of Cluster 3.

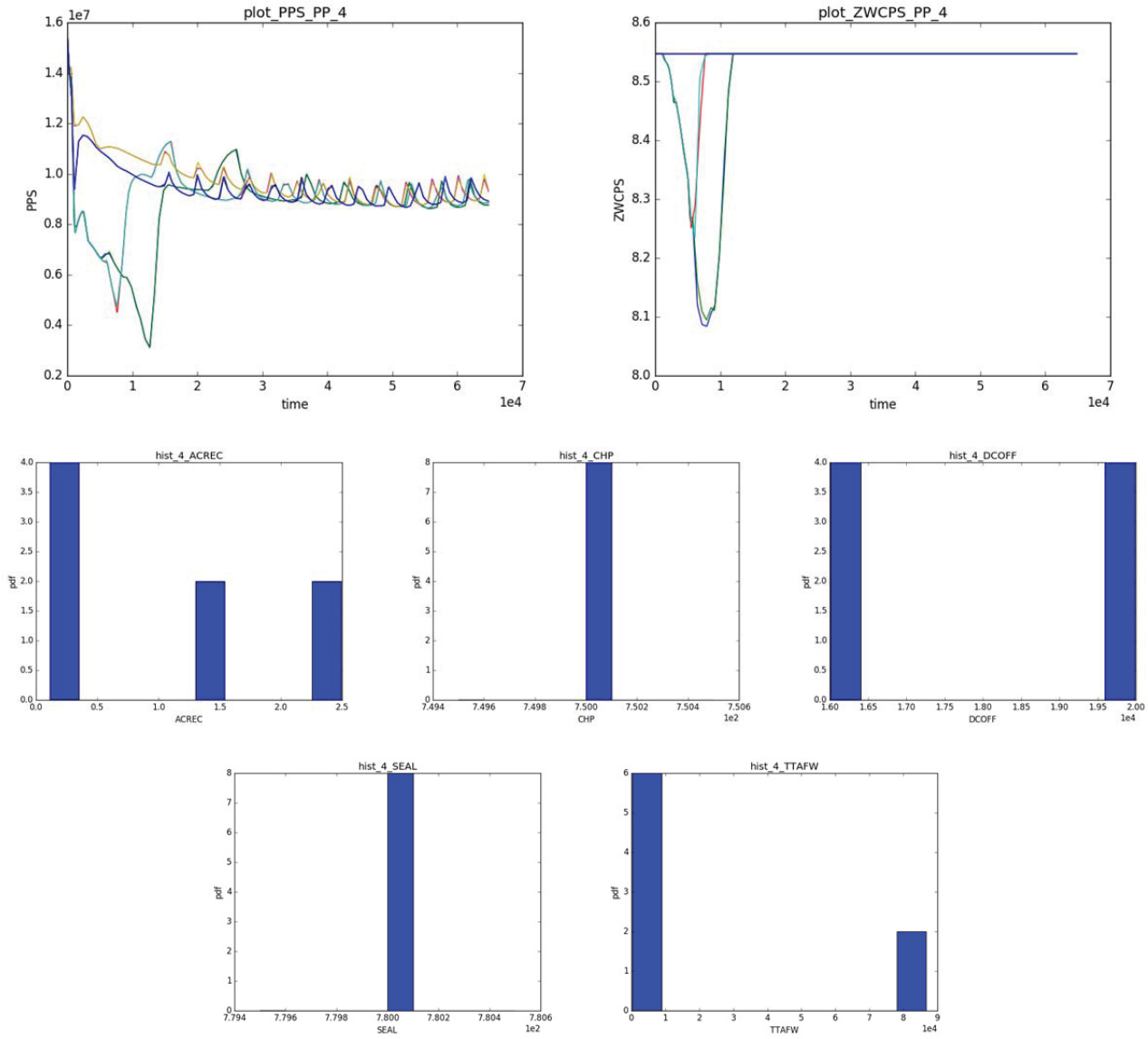

Figure 84. PWR Station Black-Out: analysis of Cluster 4.

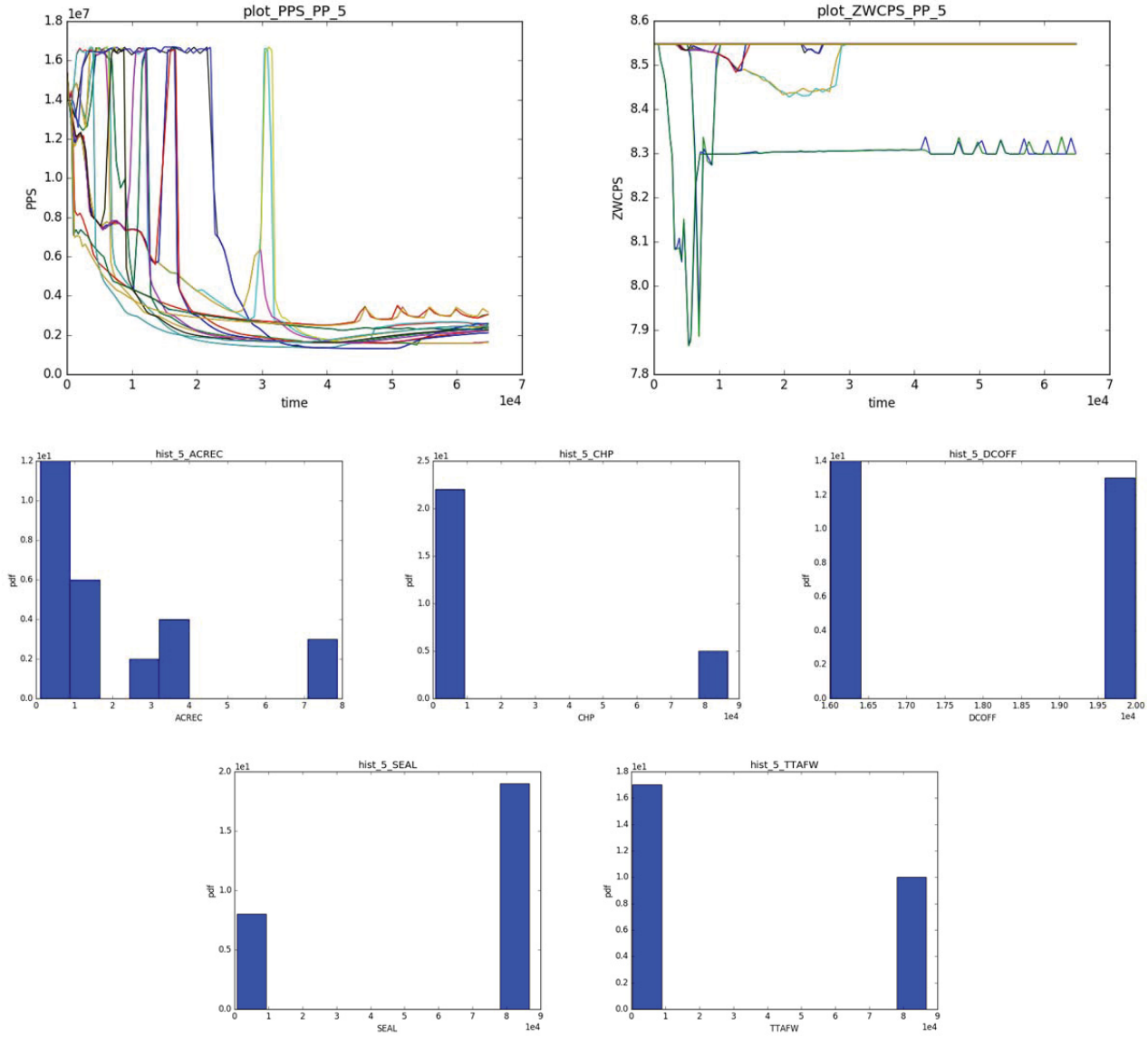

Figure 85. PWR Station Black-Out: analysis of Cluster 5.

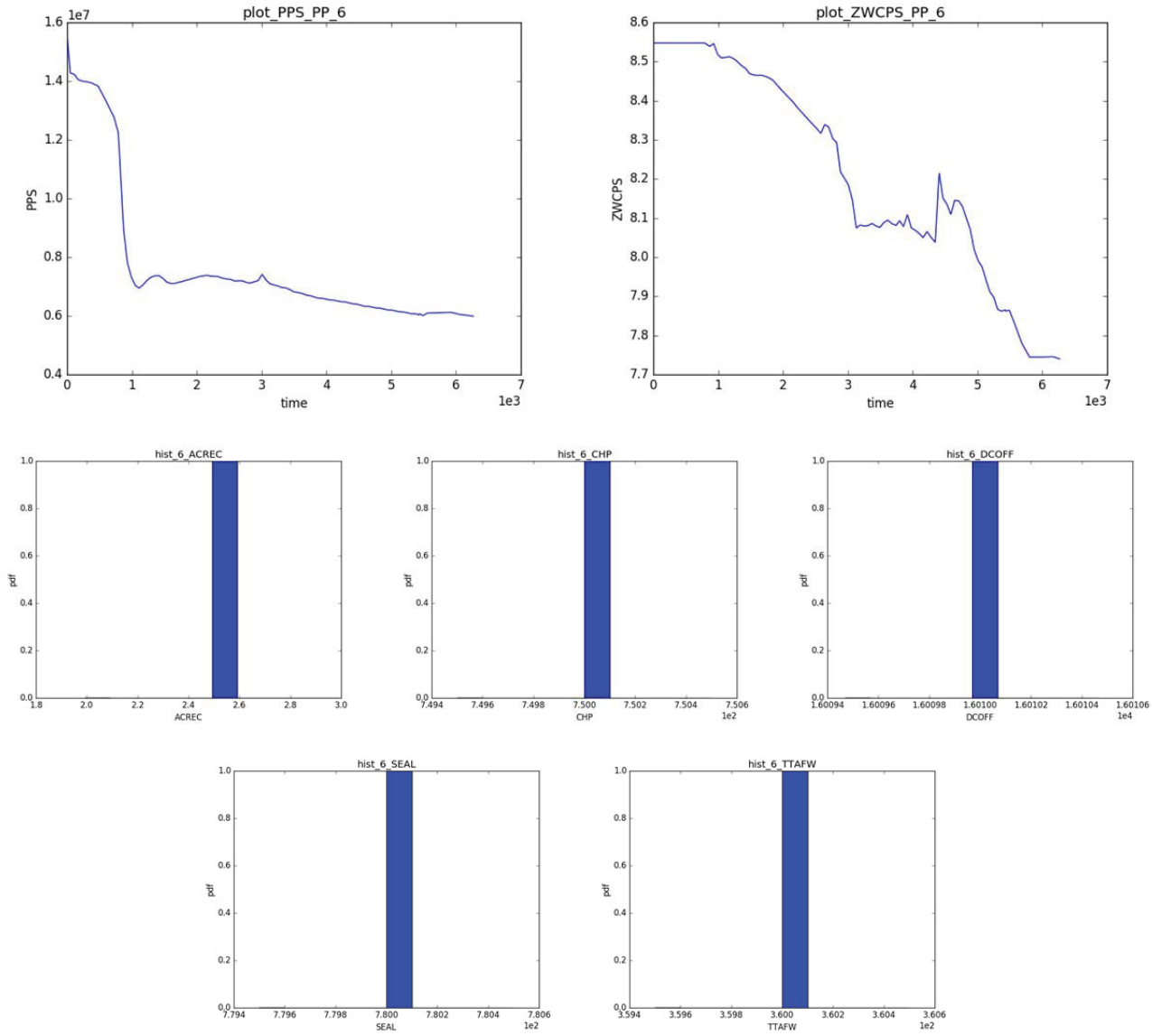

Figure 86. PWR Station Black-Out: analysis of Cluster 6.

7.4.3 Analysis Summary

In summary, using the analytical model data set we were able to gather the following information:

- The combination of early failures of both charging pump and battery failure along with a late turbine-driven auxiliary feed-water causes the system to reach a very fast and late core uncover event.

- A very early recovery of AC power is the only variable to counter a large seal LOCA. This events causes a quick drop an recovery of the water level and an end state primary pressure about half of what is expected.

- Both operating procedures (ECA-0-1 and ECA-0-1) are effective with an early AC power recovery (water level drop is very similar). A small seal LOCA gives more timing margins

- A large seal LOCA is a certain major contributor to system failure

7.5 Spent Fuel Pool

7.5.1 Model Description

In the framework of the DOE/LWRS/RISMC Project – Industry Application #2, a RELAP5-3D [35] code thermal-hydraulic model of a spent fuel pool (SFP) has been developed [36]. The scope of the model has been methodology testing for investigate External Events by deterministic and PRA codes. The RELAP5-3D SFP model has been coupled with the EMRALD and the RAVEN codes for estimating the fuel damage frequency in case of earthquake-induced components faults. The RELAP5-3D model is a simplified model of a NPP SFP (see Figure 87).

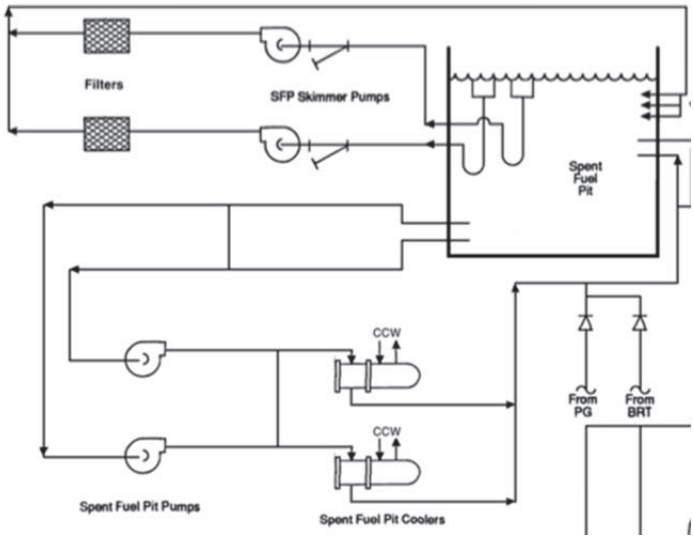

Figure 87. Model is a simplified model of a NPP SFP.

Since the scope of the RELAP5-3D model was methodology proving, a limited number of thermal-hydraulic nodes have been used in order to achieve fast-running calculations. E.g., one calculation simulating one-day transient (86400 seconds) could be run in 120 seconds of computer time. A sketch of the model is shown in Figure 88. The characteristics of the RELAP5-3D nodalization are reported in Table 3.

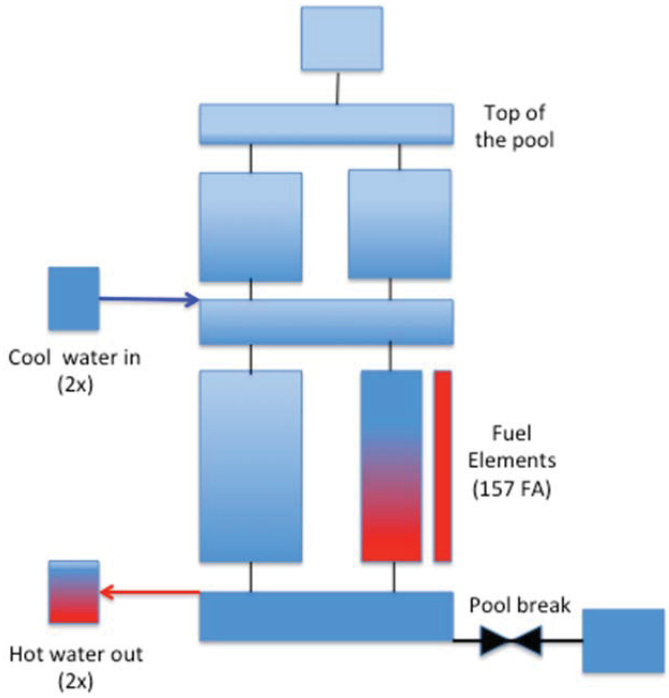

Figure 88. RELAP5-3D SFP model.

Table 3. RELAP5-3D SFP Nodalization characteristics.

Parameter	Value
Number of TH Nodes	43
Number of Junctions	49
Number of HS	16
Number of HS meshes	192

The heat load of the SFP is equivalent to the 1/3rd of the decay heat load of a ~2.5 GWth Westinghouse 3-Loop PWR core (157 Fuel Assemblies). The thermal-hydraulic channels of the fuel elements have the characteristics of a 15x15 Westinghouse PWR Fuel Assembly (see Figure 89). The water volumes of the SFP have been scaled according to the modeled heat load.

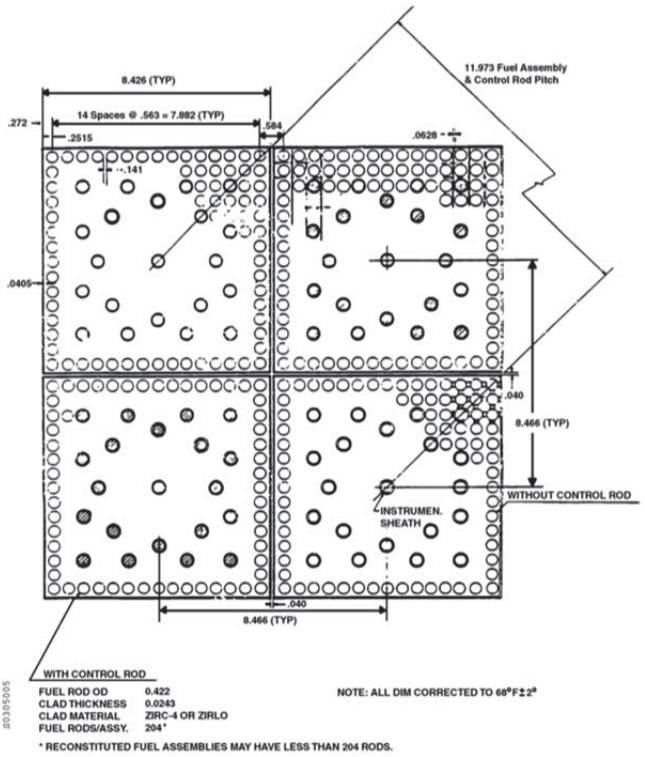

Figure 89. **15x15 PWR Westinghouse Fuel.**

The two independent cooling systems have been modeled as boundary conditions by time-dependent junctions. They remove warm water from the pool bottom and inject the cooled water on top of the fuel assemblies. Natural circulation can be calculated by the code as well thanks to the nodalization set-up. The top of the pool is connected with a time-dependent volume modeling SFP building atmosphere.

In order to simulate a SFP break by earthquakes, two valves have been added on the SFP bottom. They can simulate large and medium breaks, with an opening time of 5 seconds. The steady-state conditions are achieved by the model are reported in Table 4.

Table 4. RELAP5-3D SFP steady-state conditions.

Parameter	Value
Average Water Temperature	320 K
Fuel Heat Load	1.1 MW

A set of control logics has been implemented for simulating possible transients. They are reported below:

- Cooling pumps trip if one of these conditions are met
  - SFP liquid level < 0.1 m
  - SFP temperature > 349 K
- Operator SFP refill by emergency pumps actuation if both conditions are met:
  - Both recirculation pump failed
  - Time > Recovery Time (e.g., 30 minutes)
- End of calculations if fuel failure: T_{clad} > 1477 K (2200 F)

As an example, a loss of both cooling systems without recovery action by the operator is reported hereafter. The cooling systems fails after 1hr, leading to the boiling and to the dryout of the SFP water (water level drops to the top of active fuel in ~6 hours). Then fuel overheating and failure is achieved ~10 hours (Figure 4-6).

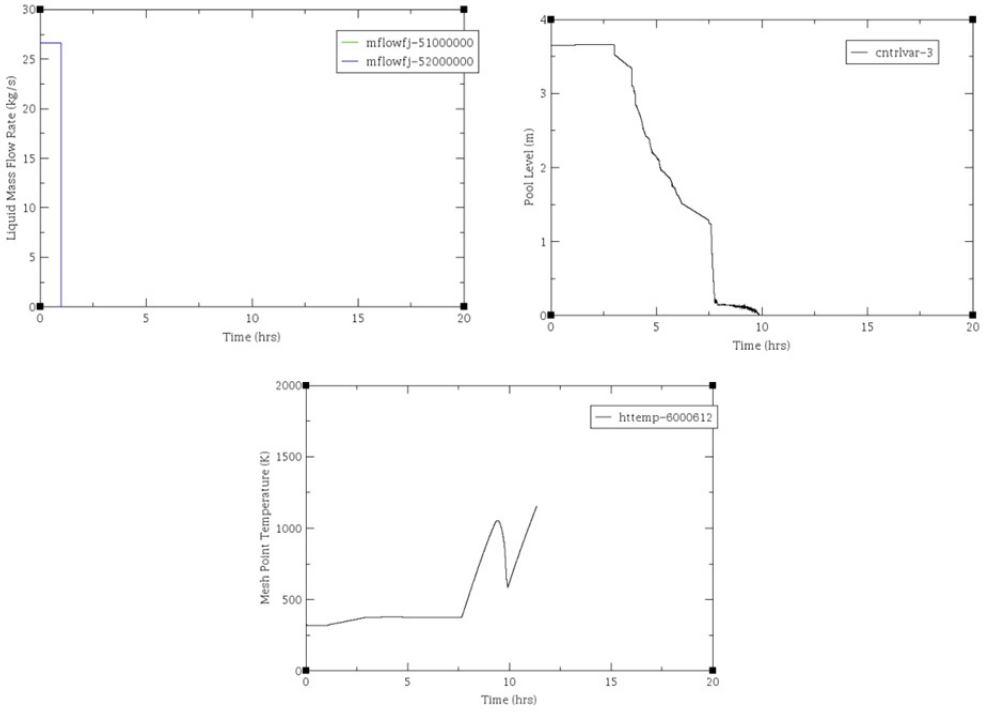

Figure 90. SFP test case: plot of SFP cooling system mass flow, SFP water level and Fuel Clad Temperature for an example scenario.

7.5.2 SFP Data Analysis

Using RAVEN we performed a Monte-Carlo sampling of the stochastic variables and generated a database of 2000 time series. Due to the heavy calculation load, this operation was performed on high performance computing (HPC) system available at INL: falcon cluster server[18]. The sampled stochastic variables are the following:

1. 20600700:6 which is time of LOCA
2. 20600560:6 which is pump1 failure time
3. 20600570:6 which is pump2 failure time
4. 1000101:3 which is size of the break
5. 20600610:6 which is the time required for operator to perform recovery action

The resulting database (HDF5 format) was downloaded and analyzed on a personal laptop using again RAVEN. The plot of the time series for six of the output variables are shown in Figure 91 while the histograms of the final clad temperature and simulation ending time are shown in Figure 92.

In order to determine the output variables to be considered in the clustering process we plotted in 3D the time series by considering a different pair of out variables for each plot (see Figure 93). Note from Figure 93

[18] 600 TFlops of performance, 19872-core SGI ICE X distributed memory cluster, 103 TB total memory.

that all 6 variables are highly correlated; thus we have decided to perform clustering on the time series by considering only one output variable: httemp_60008_12. As a record, the histograms of the 5 sampled variables are shown in Figure 94.

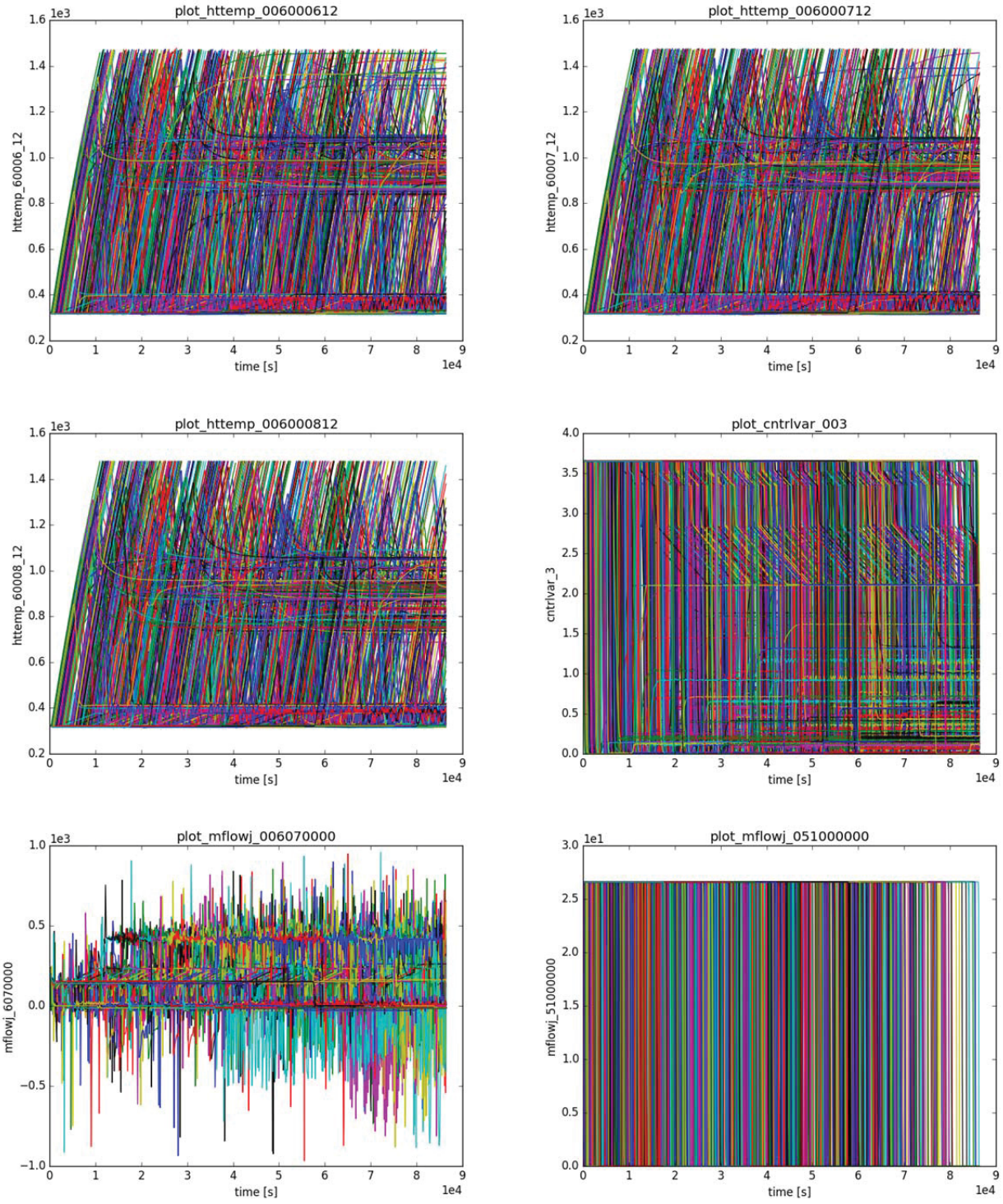

Figure 91. SFP test case: plot of all time series generated by RAVEN-RELAP5.

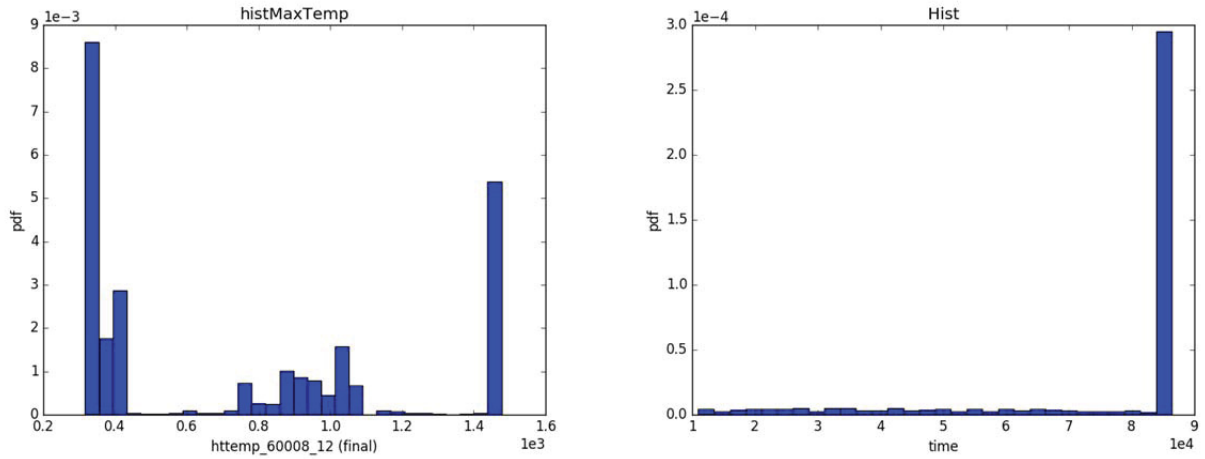

Figure 92. SFP test case: histograms of the temperatures at the end of the simulation (left) and simulation end timings (right).

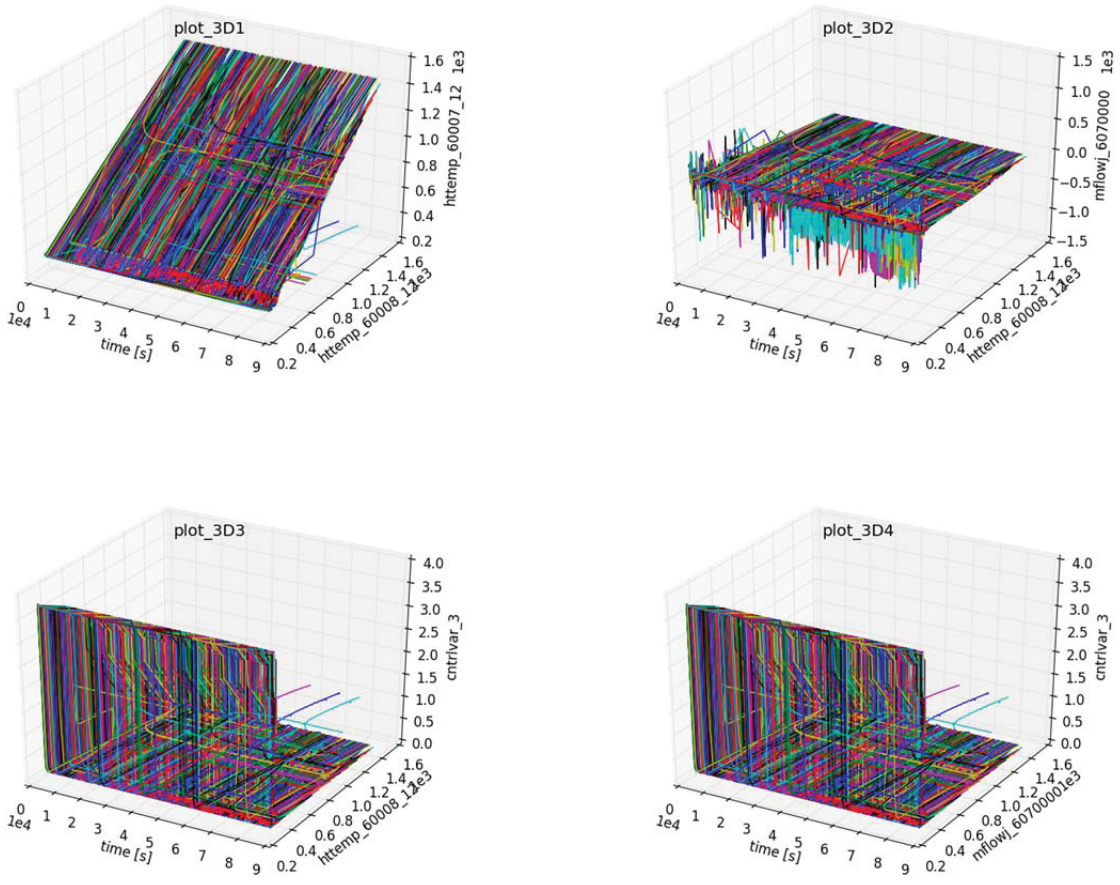

Figure 93. SFP test case: 3D plots of the five output variables; note the high correlations among them.

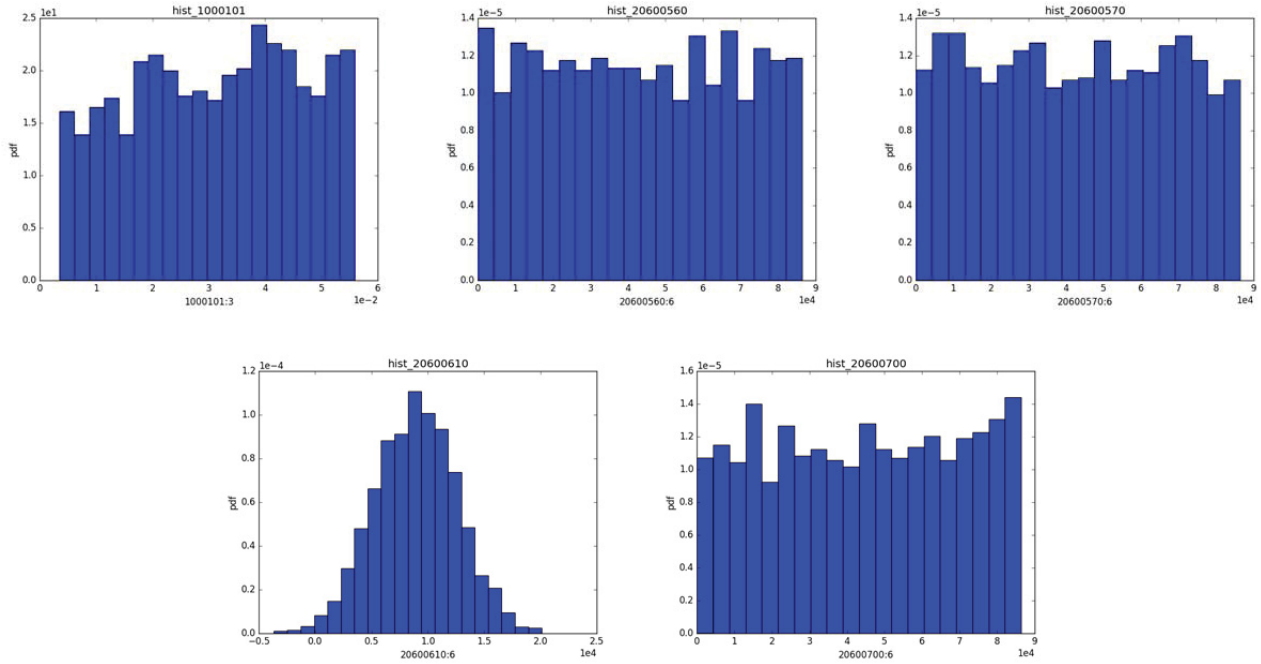

Figure 94. SFP test case: histograms of the five sampled variables.

Regarding the data analysis of this big dataset, we performed a series of clustering operations using several algorithms as follows:

1. We initially performed a hierarchical clustering coupled with DTW distance metric (Figure 95). From here we were able to clearly partition the data set into two clusters:

    a. Cluster 1 (see Figure 96): this cluster contains all simulations that led to system failure outcome. In addition this cluster contains also a few simulations that even though they did not led to system failure. By looking at the Figure 96, these simulations can be observed at the far right of the top left plot. These simulations would have actually led to system failure outcome if the simulation end-time stopping condition would have not met; thus, they can be considered as "false positives".

    b. Cluster 2 (see Figure 97): this cluster contains all scenarios that led to system success outcome. Note that many scenarios have very high clad temperatures due to the fact that system recovery occurred prior system failure event.

2. We then considered time series contained in Cluster 2 and on it we performed a further clustering using Mean-Shift. Given the structure of this data set, we were able to obtain again two clusters: Cluster 1_1 (see Figure 98) and Cluster 1_2 (see Figure 99). By looking at the temporal profiles of the scenarios contained in each cluster it is possible to observe that the temperatures at the end of the transients are significantly different: in the [350,400] interval for Cluster 1_1 and in the [700,1100] interval for Cluster 1_2.

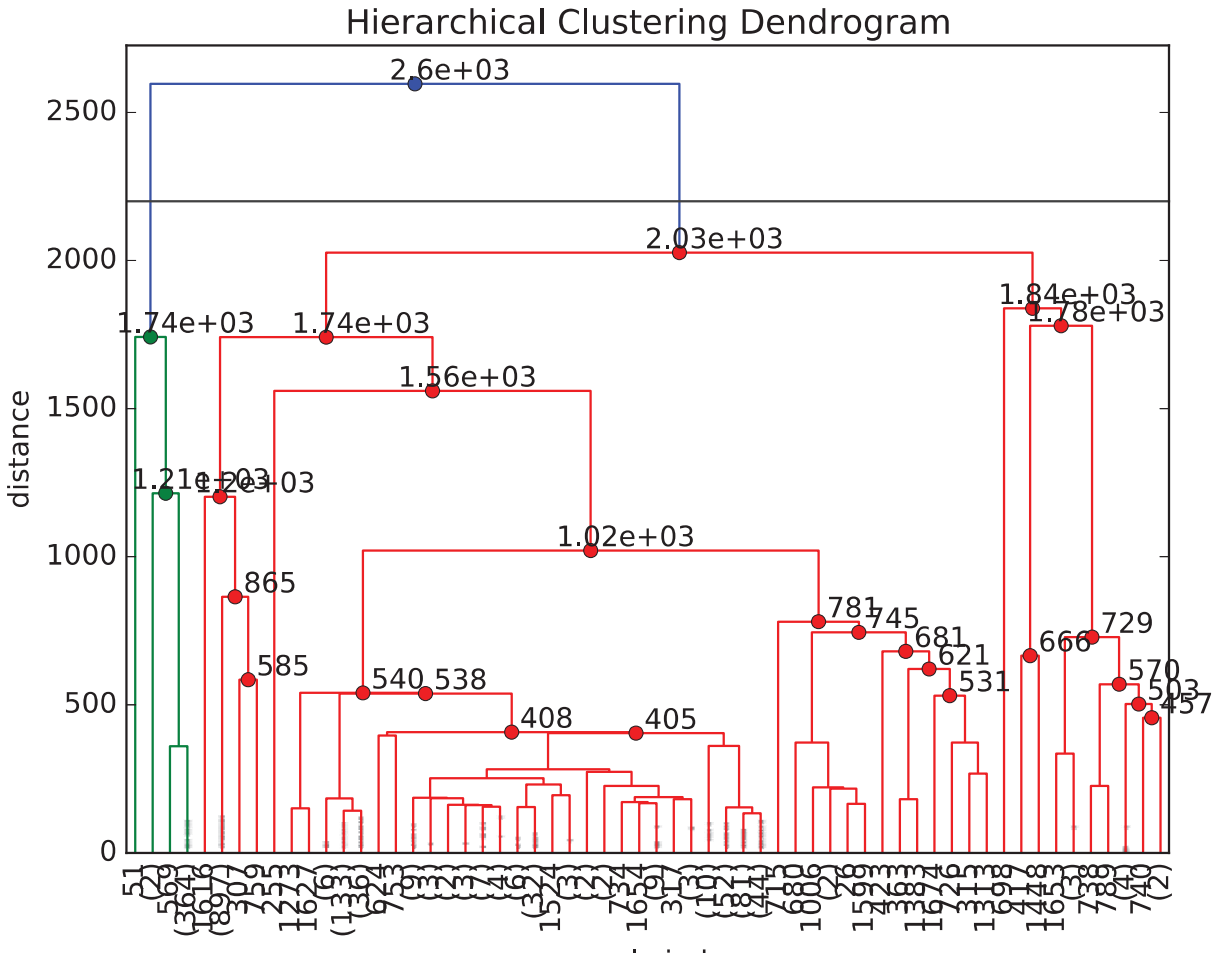

Figure 95. SFP test case: dendrogram obtained by the hierarchical algorithm.

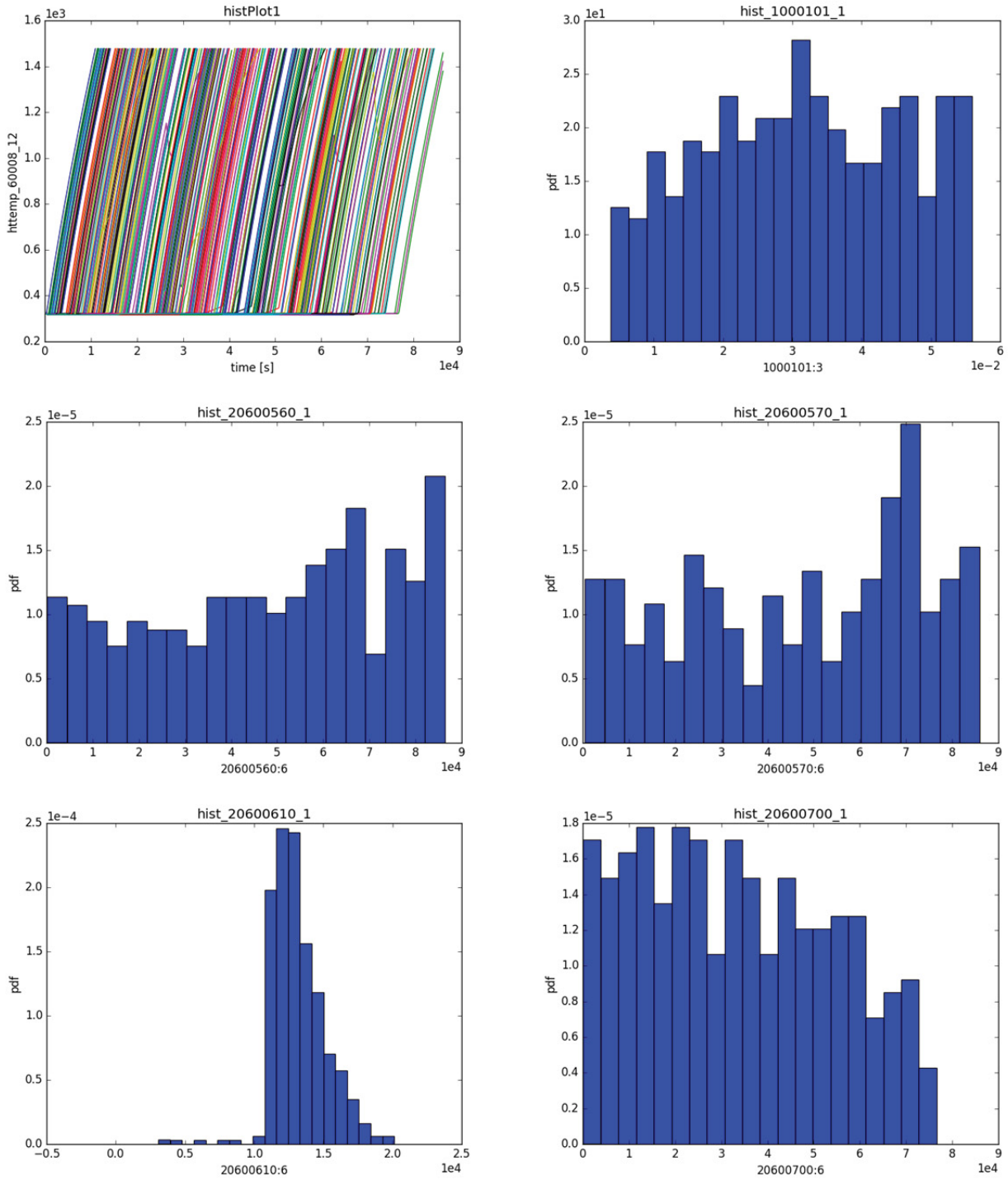

Figure 96. SFP test case: Cluster 1 data analysis. Plot of the time histories contained in this cluster and histograms of the sampled variables.

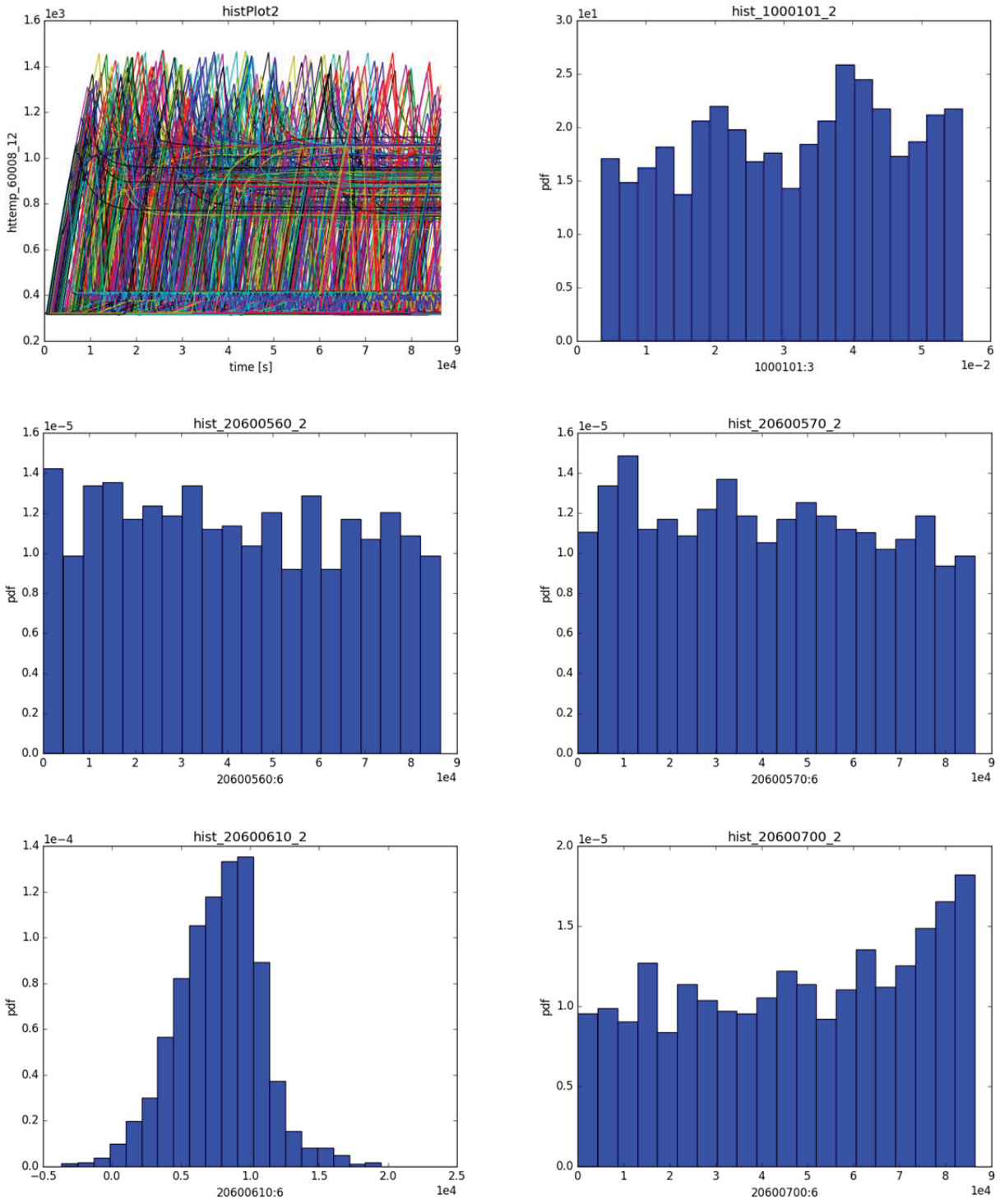

Figure 97. SFP test case: Cluster 2 data analysis. Plot of the time histories contained in this cluster and histograms of the sampled variables.

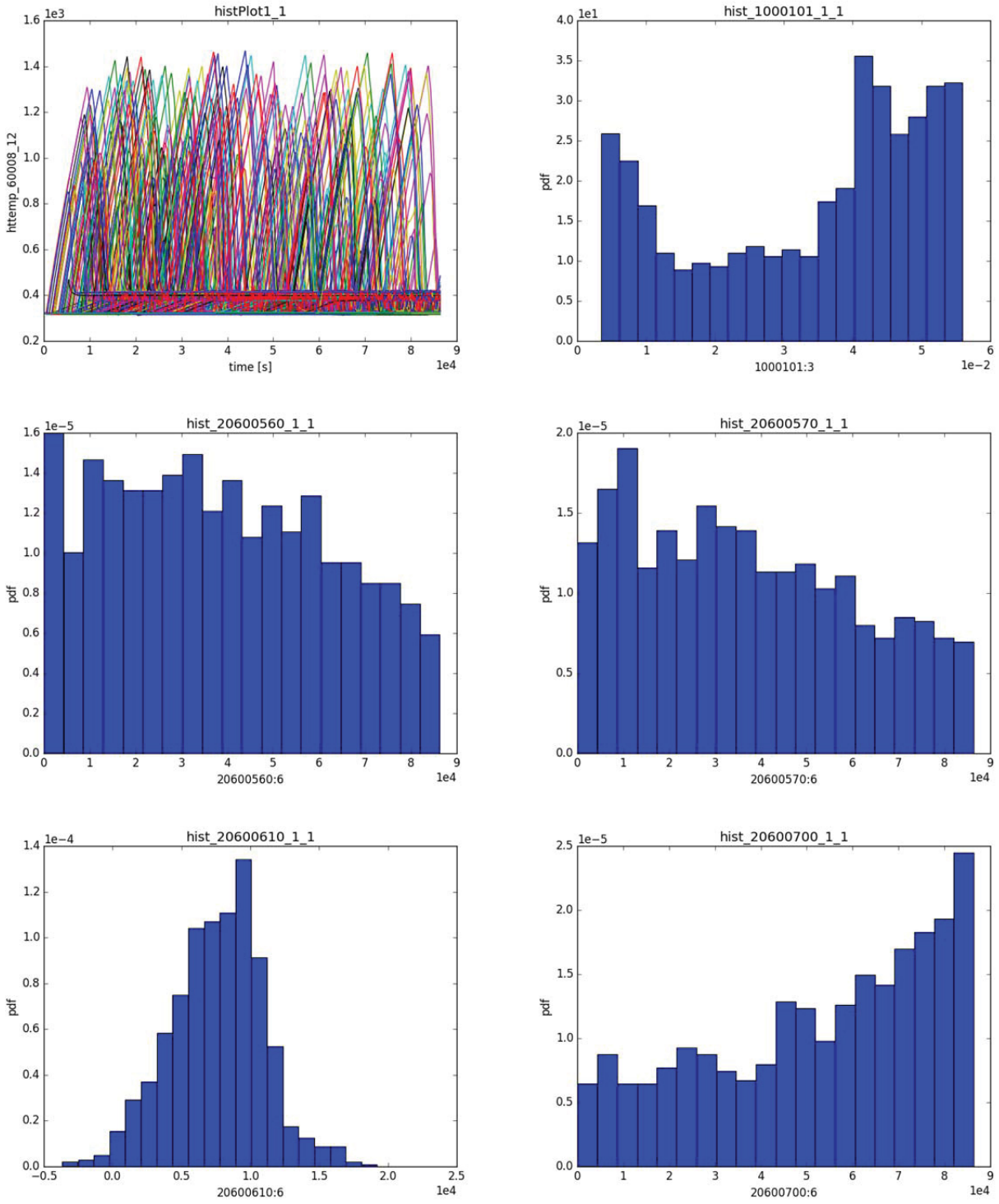

Figure 98. SFP test case: Cluster 1_1 data analysis. Plot of the time histories contained in this cluster and histograms of the sampled variables.

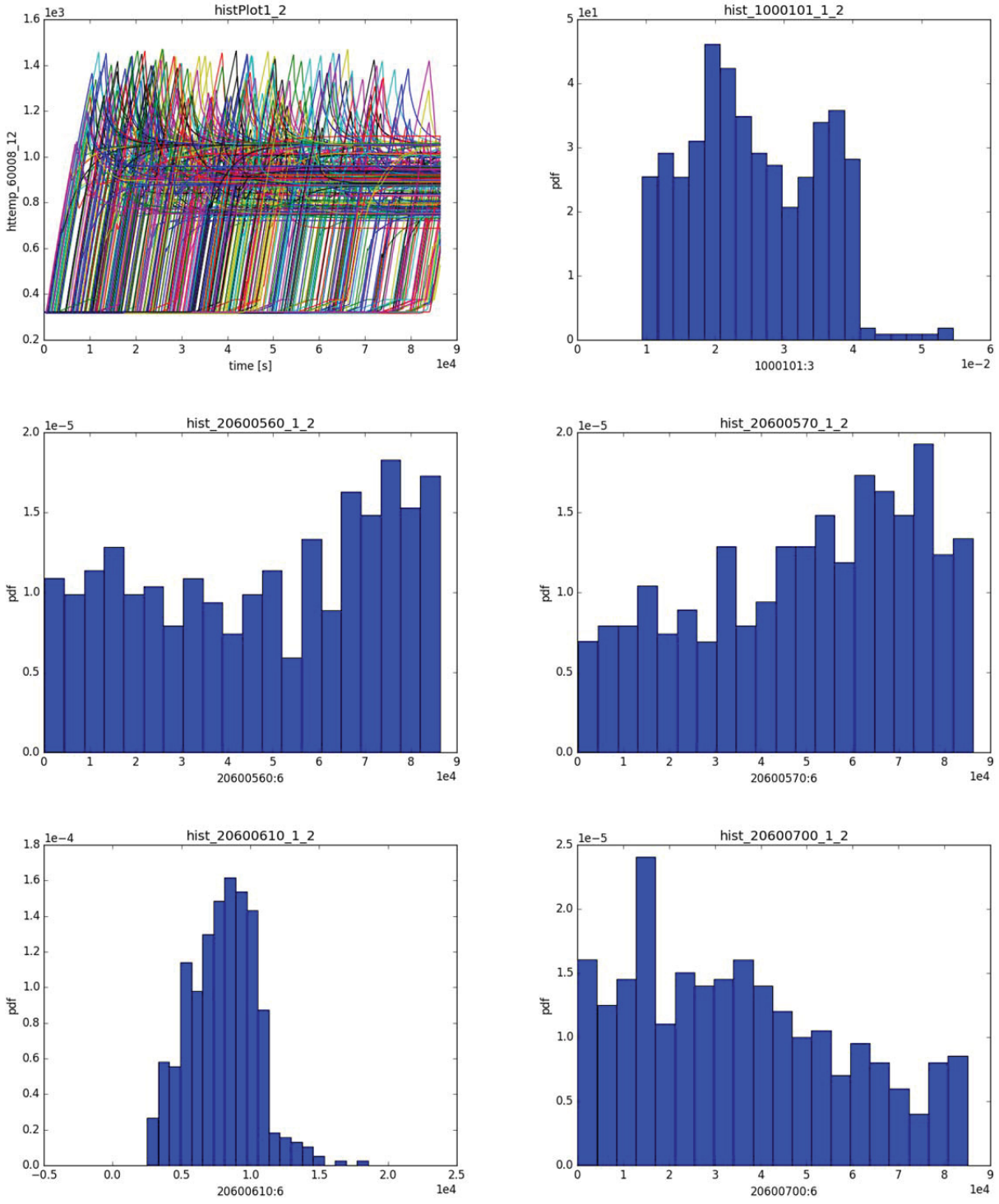

Figure 99. SFP test case: Cluster 1_2 data analysis. Plot of the time histories contained in this cluster and histograms of the sampled variables.

We investigated furthermore the differences among Cluster 1_1 and Cluster 1_2 by considering the SFP water level (see Figure 100). Figure 100 shows that SFP the water level distribution has lower mean and variance for Cluster 1_2. By observing both Figure 99 and Figure 100 we can deduce that the only a

combination of seal LOCA size (1000101:3), seal LOCA timing (20600700:6) and operator action timing (20600610:6) values create such different behavior. Thus, we studied the combination of these three input variables by scatter plotting each simulation run in this 3D input space (see Figure 101). What we obtained is very unexpected:

- Simulation runs in Cluster 1_2 are characterized by values the [0.01,0.04] interval range for seal LOCA size and lower values for seal LOCA timing

- Simulation runs in Cluster 1_1 are characterized by values outside the [0.01,0.04] interval range for seal LOCA size and lower values for seal LOCA timing and higher values of LOCA timing

- Note that the cloud of points shown in Figure 101 of Cluster 1_1 appear to surround the left, right and top boundaries of the cloud of points of Cluster 1_2

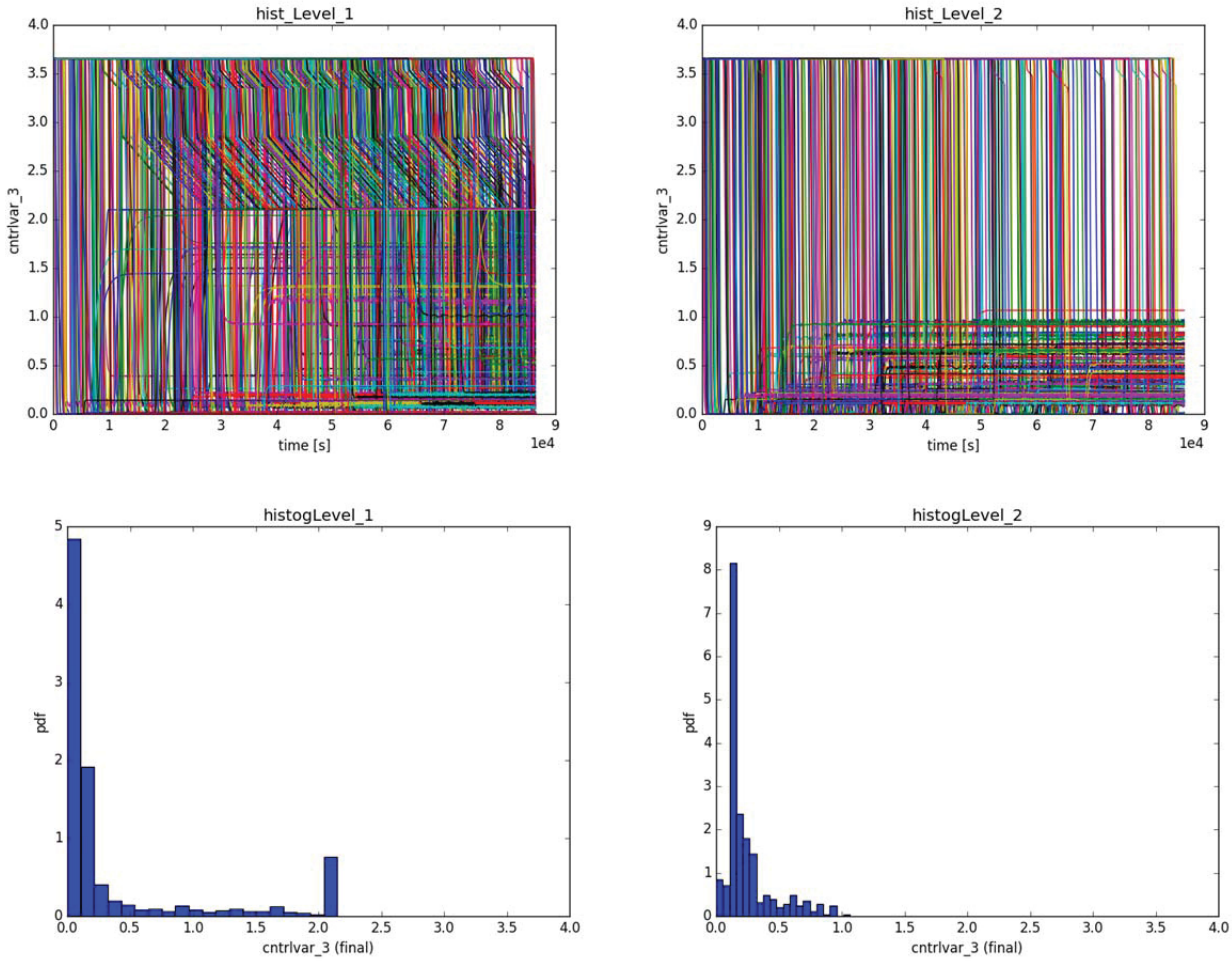

Figure 100. Plot of SFP water level temporal profiles (top) and histogram of water level at the end of simulation time (bottom) for the scenarios belonging to Cluster 1_1 (left) and Cluster 1_2 (right).

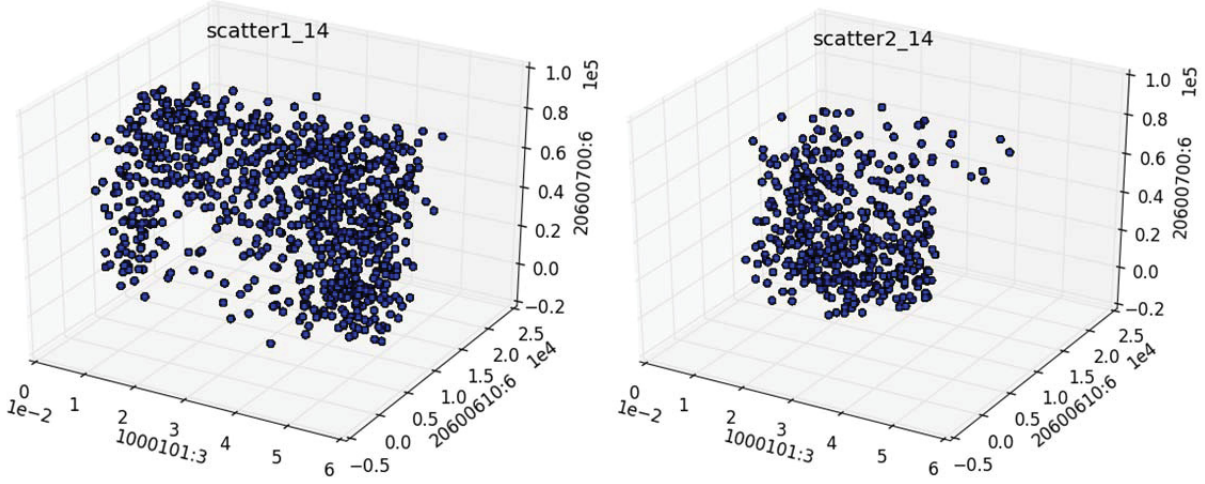

Figure 101. Scatter plot of three stochastic variables for the scenarios in Cluster 1_1 and Cluster 1_2: seal LOCA size (1000101:3), seal LOCA timing (20600700:6) and operator action timing (20600610:6).

7.5.3 Analysis Summary

In summary, using the analytical model data set we were able to gather the following information:

- The first level clustering revealed a small subset of simulations (6 runs) which even though they could be considered success only because the mission time stopping condition happened right before failure outcome was met. Thus, these simulations can be considered as false positive (i.e., simulation runs characterized by a false successful system outcome). Only by employing a first-derivative time series re-sampler coupled with hierarchical clustering with DTW warping it was possible to discover these false positive. The factor that clearly characterizes these clusters is the operator recovery timing. A value of 10000 seconds for operator recovery timing is the threshold level.

- The most relevant data exploration occurred at the second level clustering where only simulation runs leading to a successful outcome. Here we noticed two clear trends in the generated simulations:.

  By looking at Figure 92 (i.e., the histogram of the final temperature for all simulation runs) it was possible to identify such distinction but classical statistical methods would have failed to create a clear partition of the dataset. We were able to obtain such clear separation by partitioning the data set into three clusters (i.e., Cluster 1, Cluster 1_1 and Cluster 1_2[19]) obtained by employing clustering in two levels (see Figure 102).

  The scatter plots shown in Figure 101 indicate that depending on the values of seal LOCA size, seal LOCA timing and operator action timing create two behave in the SFP in term of final clad temperature and SFP water level.

[19] Recall that Cluster 2 is the union of Cluster 1_1 and Cluster 1_2

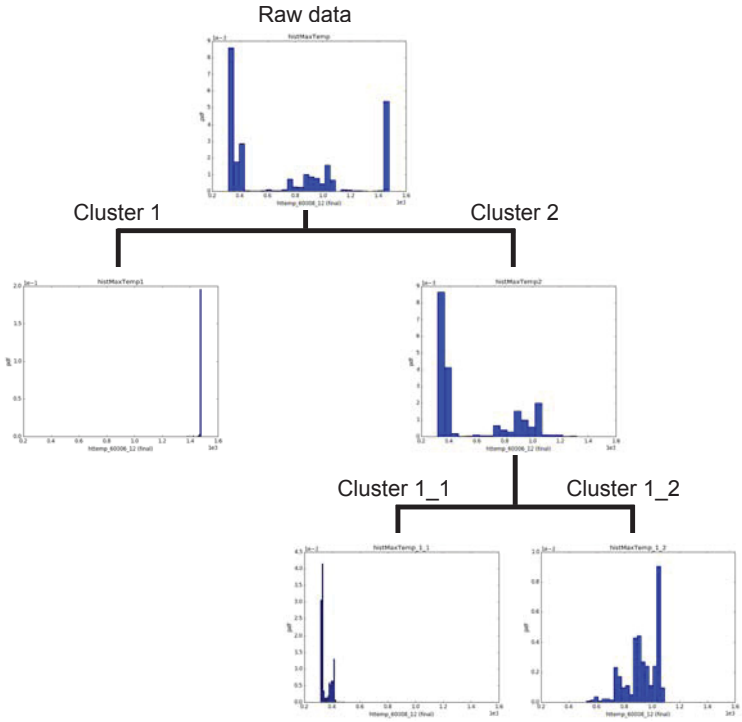

Figure 102. Histogram of final clad temperature for the clusters obtained in both clustering levels.

8. DATA VISUALIZATION

In this section we briefly describe one of the most recent development in RAVEN regarding analysis of complex data set. This development initially started as an INL internal LDRD as a collaboration between INL and University of Utah (Scientific Computing and Imaging Institute). The summary of the LDRD R&D activity is summarized in [37]. In this section we briefly described what features has been added into RAVEN and how they can be employed to analyze complex data sets using the SFP test case.

8.1 Background

The topological post-processor applies a technique called Morse-Smale regression (MSR) [38] in the context of sensitivity analysis. MSR builds upon a domain partitioning of a dataset induced by the Morse-Smale complex (MSC) and employs a linear regression fit within each partition, thereby exploiting its monotonicity. Figure 103 shows the MSC of a 2D test function with four maxima and nine minima. The MSC decomposes the domain into 16 partitions such that each partition can be well approximated by a linear model as shown in Figure 103. The MSC itself has been successfully utilized in visual exploration of simulation data modeled as high-dimensional scalar functions [39].

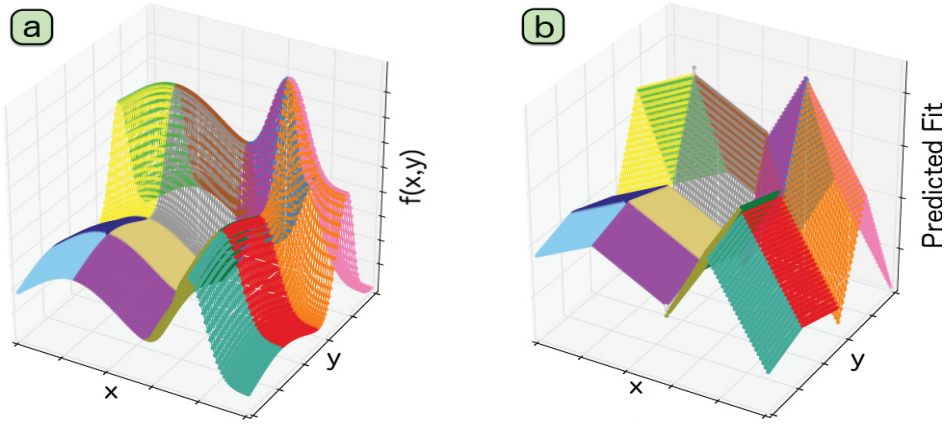

Figure 103. (a) Morse-Smale complex of a 2D height function that induces a partitioning of the domain. (b) Linear models are fit to each monotonic partition.

The topological characteristics of the MSC are at the core of MSR. Here, we give a high-level description; see [40] for details. The MSC partitions the domain of a scalar function into monotonic regions, where points in each region have gradient flow that begins at the same local minimum and ends at the same local maximum of the function. Furthermore, the MSC can be simplified based on the notion of topological persistence to create a hierarchy of features that can be simplified according to a scale parameter differentiating signal from noise [41]. For a fixed scale, the main idea behind the simplification is to merge its corresponding partitions based on a measure of their significance (i.e., persistence); see Figure 104 below for an example applied to the MSC of a 2D height function.

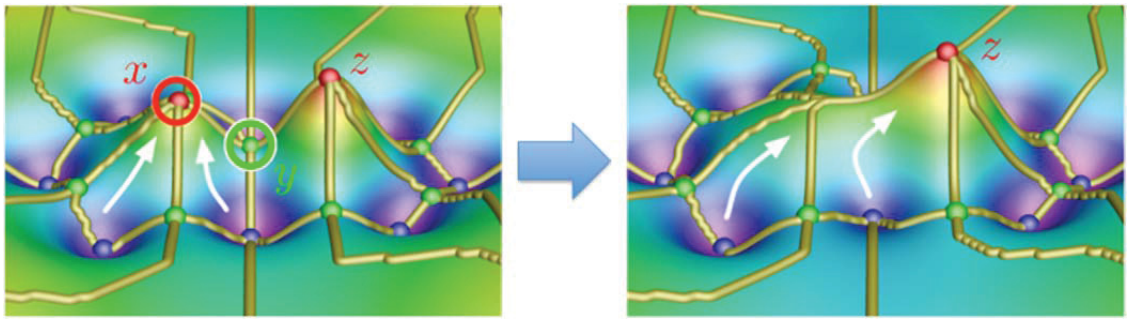

Figure 104. Example of persistence simplification of a local maximum (x) by pairing and cancelling it with the saddle point (y). The result is shown at right where the gradient is simulated to flow to the more persistent local maximum (z).

For point cloud data, the MSC can be approximated [42], enabling MSR to be applied in high dimensions. Points are connected by neighborhood graphs such as the k-nearest neighbor (kNN) graph, and gradients are estimated along the edges of the graph. In our context, we utilize similar approximation schemes [43] where points are connected using the relaxed Gabriel graph, which have been shown to give superior results in extracting topological features [44] compared to the kNN graph.

For our analysis, we use least square linear regression to fit each partition. To obtain the coefficient estimates, such a least squares fitting minimizes the sum of squared residuals. For a given partition with n data points, let $\mathbf{y} = [y_1,..., y_n]^T$ be the n-by-1 vector of observed response values, $\mathbf{X}$ be the n-by-m design matrix of the model (that is, X_{ij} is the *j*-th dimension of the *i*-th data point), and β be the m-by-1 vector of coefficients. We minimize the error estimate:

$$s(\beta) = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{m} X_{ij}\beta_j \right)^2 \tag{33}$$

In matrix form, we obtain the coefficient estimates $\hat{\beta}$ in the following way:

$$\hat{\beta} = arg\min_{\beta} s(\beta) = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{34}$$

We use the regression coefficients $\hat{\beta}_i$ ($1 \le i \le m$) to evaluate the sensitivity of the *i*-th dimension.

For a given partition fitted with a linear regression model, it is important to evaluate how well the data points fit the model by computing the *coefficient of determination*, or the R^2 score. Given a partition with n data points, for the *i*-th data point, y_i is the observed response value and $\hat{y}_i = \sum_{j=1}^{m} X_{ij}\beta_j$ is the fitted response value. The coefficient of determination is computed as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{35}$$

We extend such a notion by ranking and considering *how many* input dimensions are sufficient to provide an optimal fit. We select a subset of input dimensions for the *n* data points, apply least square linear regression on these points with reduced dimensions, and evaluate the R^2 score of the linear fit. The closer the value of R^2 is to 1, the better the linear regression fits the data with the selected subset of input dimensions.

8.2 SFP Data Analysis

We have pre-processed the time-dependent SFP simulation data (see Section 7.5) by extracting the maximum temperature (httemp_60006_12) from each simulation in order to analyze the data using RAVEN's

topological post-processor. The topological post-processor is useful for breaking down multimodal data into monotonic or approximately monotonic regions/clusters where coherent sensitivity information can be extracted per cluster of the data. The partitioning is hierarchical and thus can be interactively explored in order to determine if the data is multimodal and how well each cluster can be described by a local linear fit. For more detail, [45] explains the user interaction and visual interface of the topological post-processor available in RAVEN and the details of the algorithm are given in Section 8.1.

In this study, we analyzed the maximum temperature of the RELAP5 simulations as a function of a five-dimensional input space defined by the variables 1000101:3, 20600560:6, 20600570:6, 206000610:6, and 20600700:6.

The RAVEN high-dimensional data visualization tool applied to the SFP is shown in Figure 105. This tools re-construct the desired output variable (i.e., max clad temperature) as a function of the 5 input variables (previously sampled by RAVEN). The top-left image shows the topological connections between local minima (blue triangle in the top-left figure) and local maxima (red triangle in the top-left figure). For the SFP case 4 connections were identified (green, purple, kaki and blue). For each connection a sensitivity analysis is shown in the right plots of Figure 105. In addition, Figure 106 and Figure 107 shows the capabilities recently developed in RAVEN where a scatter plots of the scenarios contained in each minima-maxima connection in the input-sapce can be generated.

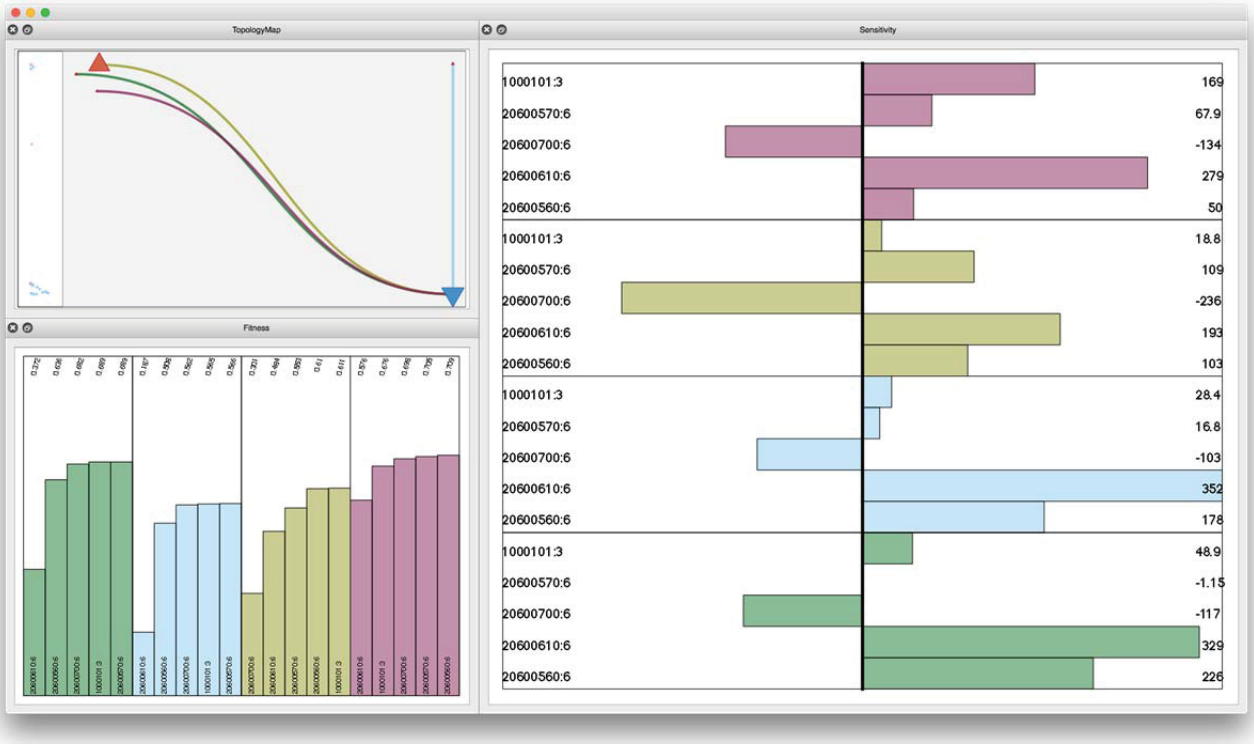

Figure 105. RAVEN high-dimensional data visualization tool for the SFP data (1).

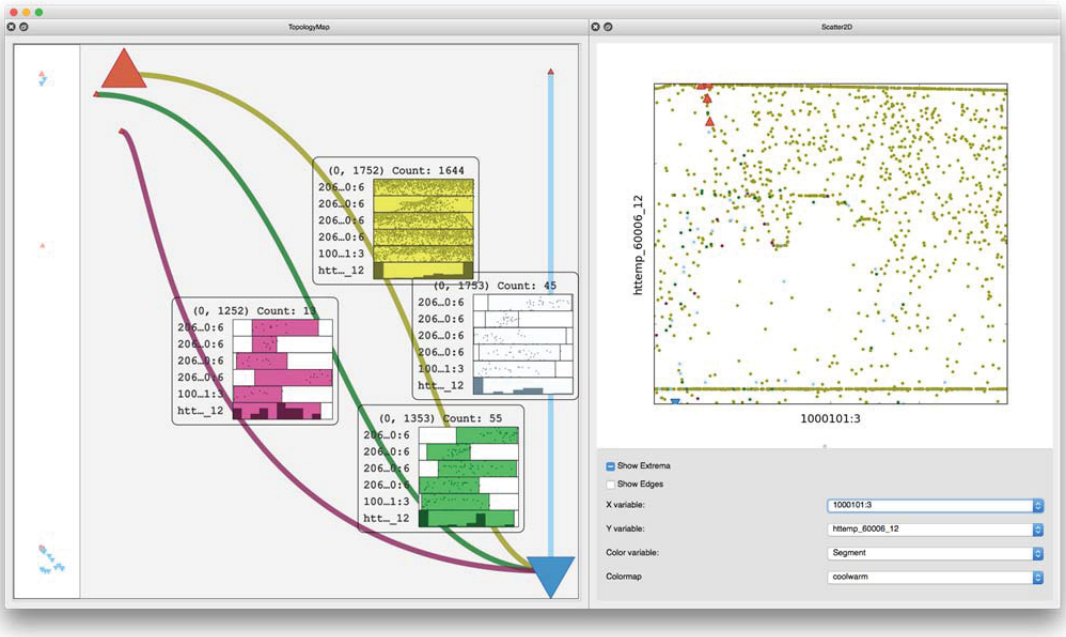

Figure 106. RAVEN high-dimensional data visualization tool for the SFP data (2).

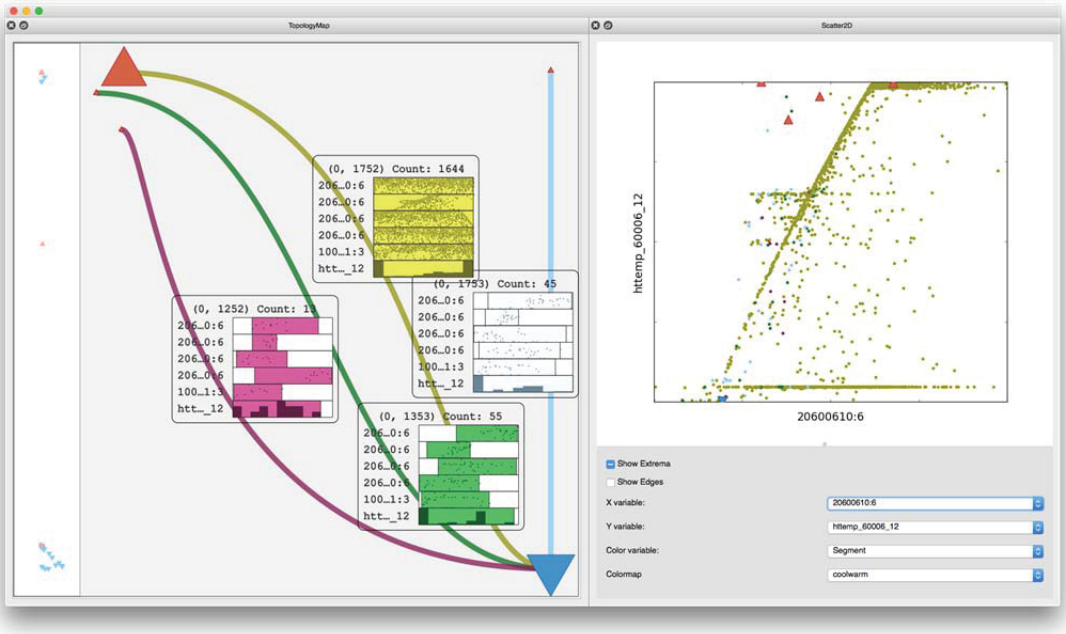

Figure 107. RAVEN high-dimensional data visualization tool for the SFP data (3).

9. CONCLUSIONS

This report summarizes the R&D activities performed during the FY2016 regarding data mining to time-dependent data. We have structured this development to leverage the existing RAVEN capabilities (i.e., data mining of static data) and include new algorithms and methods ad-hoc to analyze time-dependent data.

We have indicated several data analysis directions that can be employed to analyze time-dependent data. It is important to highlight that these directions are not valid only for RISMC type of applications. Regarding RISMC type of applications we have shown several test cases that employed both analytical models but also nuclear industry codes such as RELAP5-3D and MAAP.

The objective of these test cases was to provide possible analysis direction when dealing with nuclear transients. We have shown how it was possible to create a mapping between:

- simulation temporal profile
- simulation outcome
- timing and sequencing of events.

This is the kind of information that is relevant from a PRA point of view (and thus also a RISMC one).

It is relevant to highlight that, even though it was not explicitly shown in this report, such analysis methods and algorithms were successfully employed to detect and resolve issues and errors located in the model and in the data generation step. Some of these issues that could have not been detected with classical data analysis methods were for example:

- Erroneous implementation of the system dynamic (e.g., wrong simulation time step)
- Erroneous implementation of the system control logic (e.g., wrong activation threshold of components)
- Wrong implementation of the data management of the statistical framework during the data generation phase (e.g., Monte-Carlo sampling or Dynamic Event Tree)

Thus, such analysis greatly improved the quality and the understanding of the data generated. Classical methods to perform data analysis would have not been able to reach such a level of data exploration.

In this report we have also shown how it was possible to measure reliability importance of components in a simulation based PRA environment (such as RISMC). The analyses presented in this report resemble the ones performed using risk-importance measures in a classical PRA environment (such as SPAR models). In our case we focused the attention more the link between timing/sequencing of events and sampled variables with simulation temporal profile without explicitly considering any probabilistic type of information (see confirmation in the next section).

In conclusion, the development of the RAVEN code during FY16 has provide capabilities to the code itself in terms of time-dependent data mining that no other software product (commercial or open-source) can offer. These capacities must be looked not only from a RISMC (or in general simulation-based PRA) perspective but also to a more broad engineering perspective that includes: scenario forecasting (predictive capabilities), code calibration on time-dependent data etc.

9.1 Possible Future Developments

Given what has been presented in this report we thought it would be worth presenting a set of development directions which would further improve the effectiveness of RAVEN to perform data mining:

- Interactive data mining: this capability would allow the user to actively explore data while RAVEN is running and perform any type of data post-processing without the need to modify the RAVEN input file

and re-run RAVEN itself. As an example would be make the hierarchical clustering interactive so that the user can interactively change the threshold level and every time the threshold is change RAVEN perform the processing of each obtained cluster in real-time.

- In the previous section we mentioned the concept of risk-importance measures. These concepts belong mainly to classical PRA methods (which are based on static logic structures such as event-trees and fault-trees). The next suggest development would include the inclusion and adaption of similar risk-importance measures in a simulation-based PRA environment such as RISMC.
- As of now RAVEN input file (which is in .xml format) can be edited using any available text editor (freely available on both Mac and Linux). We have experienced that the RAVEN input file for the data mining of complex data sets (e.g., SFP data) may be very long and complex. This can cause input error that can be caught after RAVEN has completed the analysis. Thus we recommend the development of a Graphic User Interface (GUI) that can facilitate the user to create complex RAVEN input files. The ideal case would be a drag-and-drop user interface where RAVEN components are graphically shown and linked together. An example of open-source is Sirius (http://www.eclipse.org/sirius/overview.html) and it is shown in Figure 108.

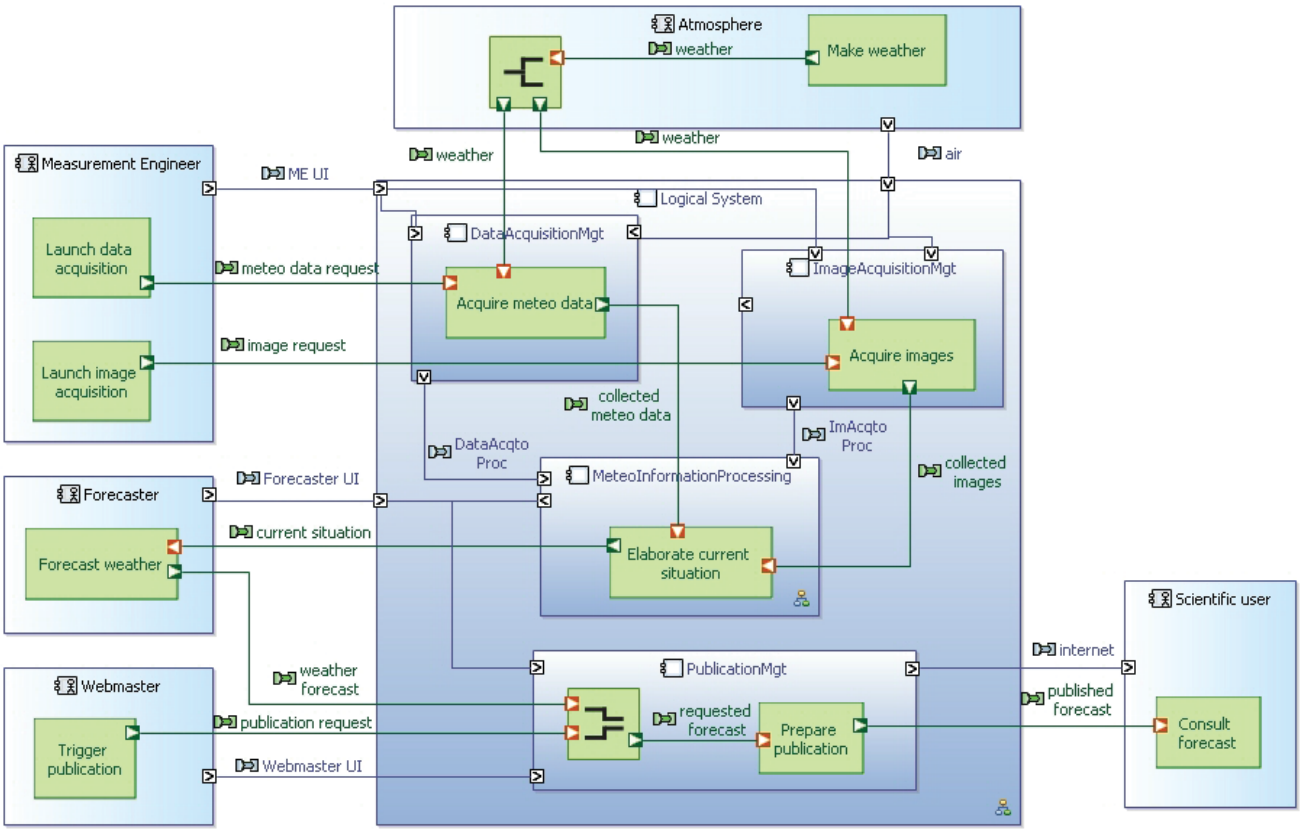

Figure 108. Example of drag-and-drop GUI where components are graphically shown and linked together.

10. REFERENCES

[1] C. Smith, C. Rabiti, and R. Martineau, "Risk Informed Safety Margins Characterization (RISMC) Pathway Technical Program Plan", Idaho National Laboratory technical report: INL/EXT-11-22977 (2011).

[2] D. Mandelli, C. Smith, C. Rabiti, A. Alfonsi, R. Youngblood, V. Pascucci, B.Wang, D. Maljovec, P.-T. Bremer, T. Aldemir, A. Yilmaz, and D. Zamalieva, "Dynamic PRA: an overview of new algorithms to generate, analyze and visualize data," in Proceeding of American Nuclear Society (ANS), Washington (DC), 2013.

[3] D. Mandelli, C. Smith, T. Riley, J. Nielsen, J. Schroeder, C. Rabiti, A. Alfonsi, J. Nielsen, R. Kinoshita, D. Maljovec, B. Wang, and V. Pascucci, "Overview of new tools to perform safety analysis: BWR station black out test case," in Proceedings for PSAM 12 Conference, Honolulu (2014).

[4] D. Mandelli, Z. Ma, and C. Smith, "Comparison of a traditional probabilistic risk assessment approach with advanced safety analysis," in Proceeding of American Nuclear Society (2014).

[5] D. Mandelli, C. Smith, A. Alfonsi, C. Rabiti, J. Cogliati," Improved Sampling Algorithms in the Risk-Informed Safety Margin Characterization Toolkit", Idaho National Laboratory technical report: INL/EXT-15-35933 (2015).

[6] C. Rabiti, A. Alfonsi, D. Mandelli, J. Cogliati, R. Martinueau, C. Smith, "Deployment and Overview of RAVEN Capabilities for a Probabilistic Risk Assessment Demo for a PWR Station Blackout Idaho National Laboratory technical report: INL/EXT-13-29510 (2013).

[7] D. Mandelli, S. Prescott, C. Smith, A. Alfonsi, C. Rabiti, J. Cogliati, R. Kinoshita, "Modeling of a Flooding Induced Station Blackout for a Pressurized Water Reactor Using the RISMC Toolkit," *in ANS PSA 2015 International Topical Meeting on Probabilistic Safety Assessment and Analysis Columbia, SC*, on CD-ROM, American Nuclear Society, LaGrange Park, IL, 2015.

[8] R. L. Boring, R. Benish Shirley, J. C. Joe, D, Mandelli, and C. Smith, "Simulation and Non-Simulation Based Human Reliability Analysis Approaches", Idaho National Laboratory technical report: INL/EXT-14-33903 (2014).

[9] A. David, R. Berry, D. Gaston, R. Martineau, J. Peterson, H. Zhang, H. Zhao, L. Zou, "RELAP-7 Level 2 Milestone Report: Demonstration of a Steady State Single Phase PWR Simulation with RELAP-7," Idaho National Laboratory technical report: INL/EXT-12-25924 (2012).

[10] A. Alfonsi, C. Rabiti, D. Mandelli, J. Cogliati, R. Kinoshita, and A. Naviglio, "RAVEN and Dynamic Probabilistic Risk Assessment: Software Overview," in *Proceedings of European Safety and Reliability Conference ESREL* (2014).

[11] D. Gaston, C. Newman, G. Hansen and D. Lebrun-Grandi, "MOOSE: A parallel computational framework for coupled systems of nonlinear equations," *Nuclear Engineering Design*, **239**, pp. 1768-1778 (2009).

[12] B. Spencer, Y. Zhang, P. Chakraborty, S.B. Biner, M. Backman, B. Wirth, S. Novascone, J. Hales, "Grizzly Year-End Progress Report", Idaho National Laboratory technical report: INL/EXT-13-30316 (2013).

[13] E. Zio, M. Marseguerra, J. Devooght, and P. Labeau, "A concept paper on dynamic reliability via Monte Carlo simulation," in *Mathematics and Computers in Simulation*, pp. 47–371 (1998).

[14] J. C. Helton and F. J. Davis, "Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems," *Reliability Engineering & System Safety*, **81** – 1 (2003).

[15] B. Rutt, U. Catalyurek, A. Hakobyan, K. Metzroth, T. Aldemir, R. Denning, S. Dunagan, and D. Kunsman, "Distributed dynamic event tree generation for reliability and risk assessment," in *Challenges of Large Applications in Distributed Environments*, pp. 61-70, IEEE (2006).

[16] H. S. Abdel-Khalik, Y. Bang, J. M. Hite, C. B. Kennedy, C. Wang, "Reduced Order Modeling For Nonlinear Multi-Component Models," *International Journal on Uncertainty Quantification*, **2** - 4, pp. 341-361 (2012).

[17] K. Inaba, *Boost C++ Library Programming*, Shuwa System, ISBN: 4-7980-0786-2 (2004).

[18] S. Sonat, D. Maljovec, A. Alfonsi, and C. Rabiti, "Developing and Implementing the Data Mining Algorithms in RAVEN", Idaho National Laboratory technical report: INL/EXT-15-36632 (2015).

[19] J. Cogliati, J. Chen, J. Patel, D. Mandelli, D. Maljovec, A. Alfonsi, P. Talbot, C. Rabiti, C. Wang, "Time-Dependent Data Mining in RAVEN", Idaho National Laboratory technical report: INL/EXT-16-39860 (2016)

[20] D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *AAAI Workshop on Knowledge Discovery in Databases*, pp. 229-248 (1994).

[21] V. Bryant, *Metric Spaces, Iteration and Application*, Cambridge University Press (1985).

[22] L. Steen, J. Seebach, *Counterexamples in Topology*, Dover (1995).

[23] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," ACM Computing Surveys, **31**, no. 3, pp. 264-323 (1999).

[24] J. L. bentley, "Multidimensional Binary Search Tree Used for Associative Searching," in *Communications of the ACM*, **18**, pp. 509-517 (1975).

[25] J. B. Macqueen, "Some methods for classification and analysis of multivariate observations," in *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, **1**, pp. 281-297, University of California Press (1967).

[26] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**, no. 8, pp. 790-799 (1995).

[27] A. K. Jain, K. Dubes, and C. Richard, *Algorithms for clustering data*, Upper Saddle River, NJ (USA): Prentice-Hall, Inc. (1988).

[28] D. Mandelli, A. Yilmaz, T. Aldemir, K. Metzroth, and R. Denning, "Scenario clustering and dynamic probabilistic risk assessment," *Reliability Engineering & System Safety*, **115**, pp. 146-160 (2013).

[29] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Mining and Knowledge Discovery*, **26**, no. 2, pp. 275–309 (2013).

[30] D. Mandelli, C. Smith, A. Yilmaz, and T. Aldemir, "Mining nuclear transient data through symbolic conversion," in *Proceedings of ANS PSA 2013*, American Nuclear Society, LaGrange Park, IL (2013).

[31] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," *Workshop on Research Issues in Data Mining and Knowledge Discovery, the 8th ACM SIGMOD* (2003).

[32] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen and G Batista, *The UCR Time Series Classification Archive*, (2015) [www.cs.ucr.edu/~eamonn/time_series_data/].

[33] T. Aldemir, S. Guarro, D. Mandelli, J. Kirschenbaum, L. Mangan, P. Bucci, M. Yau, E. Ekici, D. Miller, X. Sun, and S. Arndt, "Probabilistic risk assessment modeling of digital instrumentation and control systems using two dynamic methodologies", *Reliability Engineering and System Safety*, **95**, no. 10, pp. 1011-1039 (2010).

[34] K. Metzroth, "A Comparison of Dynamic and Classical Event Tree Analysis for Nuclear Power Plant Probabilistic Safety/Risk Assessment", Doctor of Philosophy Thesis, Ohio State University, Nuclear Engineering (2011).

[35] RELAP5-3D Code Development Team, RELAP5-3D Code Manual (2005).

[36] C. Parisi, S. Prescott, R. Yorg, J. Coleman, R. Szilard, "External Events Analysis for LWRS/RISMC Project: Methodology Development and Early Demonstration", Transactions of the American Nuclear Society, Vol. 114, No. 1, Pages 570-571. New Orleans, Louisiana, June 12-16, 2016.

[37] D. Maljovec, S. Liu, B.Wang, D. Mandelli, P. T. Bremer, V. Pascucci, and C. Smith, "Analyzing simulation-based PRA data through traditional and topological clustering: a BWR station blackout case study," Reliability Engineering & System Safety, **145**, no. 1, pp. 262-276 (2015).

[38] F. Chazal, L. J. Guibas, S. Y. Oudot, and P. Skraba. Analysis of scalar fields over point cloud data. In ACM-SIAM SODA, 2009.

[39] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. Discrete Comput. Geom., 37 (1): 103-120, 2007.

[40] C. Correa and P. Lindstrom. Towards robust topology of sparsely sampled data. IEEE TVCG, 17 (12): 1852-1861, Dec 2011.

[41] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. Discrete Comput. Geom., 30 (87-107), 2003.

[42] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In IEEE FOCS, Washington, DC, 2000.

[43] J. H. Friedman. Multivariate adaptive regression splines, Ann. Statist., vol. 19, no. 1, pp. 1–67, 03 1991.

[44] S. Gerber, P. Bremer, V. Pascucci, and R. Whitaker. Visual exploration of high dimensional scalar functions. IEEE TVCG, 16(6), Nov 2010.

[45] S. Gerber, O. Rübel, P. Bremer, V. Pascucci, and R. Whitaker. Morse-Smale Regression. Manuscript, 2011.

[46] D. Maljovec, B. Wang, D. Mandelli, P. Bremer, and V. Pascucci. Analyzing dynamic probabilistic risk assessment data through clustering. PSA, 2013.

[47] D. Maljovec, B. Wang, V. Pascucci, P. Bremer, M. Pernice, D. Mandelli, and R. Nourgaliev. Exploration of high-dimensional scalar functions for nuclear reactor safety analysis and visualization. M&C, 2013.

[48] D. Maljovec, B. Wang, P. Rosen, A. Alfonsi, G. Pastore, C. Rabiti, and V. Pascucci, Rethinking sensitivity analysis of nuclear simulations with topology, in Proceedings of IEEE Pacific Visualization (PacificVis), 2016.