An Overview of Methods to Analyze Dynamic PRA Data

D. Mandelli, D. Maljovec, C. Parisi, A. Alfonsi, C. Smith, C. Rabiti
Idaho National Laboratory (INL), Idaho Falls, USA

ABSTRACT: In the past decades, several numerical simulation codes have been employed to simulate accident dynamics. In order to evaluate the impact of uncertainties into accident dynamics, several stochastic methodologies have been coupled with these codes. These stochastic methods range from classical Monte-Carlo and Latin Hypercube sampling to stochastic polynomial methods. Similar approaches have been introduced into the risk and safety community where stochastic methods have been coupled with safety analysis codes in order to evaluate the safety impact of timing and sequencing of events. These approaches are usually called Dynamic PRA or simulation-based PRA methods. These uncertainties and safety methods usually generate a large number of simulation runs (database storage may be on the order of gigabytes or higher). The scope of this paper 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.

1 INTRODUCTION

In the past decades, several numerical simulation codes have been employed to simulate accident dynamics, e.g., RELAP5-3D (RELAP5-3D team 2005), MELCOR (Gauntt 2001). In order to evaluate the impact of uncertainties into accident dynamics, several stochastic methodologies have been coupled with these codes. These stochastic methods range from classical Monte-Carlo and Latin Hypercube sampling to stochastic polynomial methods. Similar approaches have been introduced into the risk and safety community where stochastic methods, e.g., RAVEN (Alfonsi 2014), ADAPT (Rutt et al. 2006), MCDET (Hofer 2002), have been coupled with safety analysis codes in order to evaluate the safety impact of timing and sequencing of events on the accident progression. These approaches are usually called Dynamic PRA methods. These simulation-based uncertainties and safety methods usually generate a large number of simulation runs which are typically discarded once coarse averaging coefficients (e.g., core damage frequency or sensitivity coefficients) are determined.

The scope of this paper is to present a broad overview of data mining methods and algorithms that can be used to analyze and extract useful information from large data sets containing time dependent data. In this context, extracting information means constructing input-output correlations, finding commonalities, and identifying outliers. Data mining is a fairly generic concept that entails the generation of information and knowledge from data.

The process of generation of information/knowledge can be performed in various ways depending on the type of application but it possible to classify data analysis approaches into three categories:

- *Reduced Order Modeling*: algorithms that 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 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 measure (i.e., a distance metric). This paper focuses on the latter category applied in particular to the analysis of time dependent data, i.e., simulated accident transients. By grouping

simulated transients, provided a set of similarity laws, it is possible to identify commonalities regarding initial and boundary conditions and accident progression.

We will describe several aspects that orbit around data mining of Dynamic PRA data such as:

- Data pre-processing: how the data is pre-processed prior behind analyzed
- Data representation: how each transient is represented from a mathematical point of view
- Similarity metrics: how distance among transient is measured and calculated

2 DATA SET FORMAT

We will indicate with Ξ the data set generate by any of the methods mentioned above which contain N time series[1] H_n:

$$\Xi = \{H_1, \dots, H_n, \dots, H_N\} \quad (1)$$

To preserve generality, we can assume that each scenario H_n contains three components:

$$H_n = \{\boldsymbol{\theta}_n, \boldsymbol{\Delta}_n, \boldsymbol{\Gamma}_n\} \quad (2)$$

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[2] from 0 to t_n):
$$\boldsymbol{\theta}_n = \{x_1^n, \dots, x_M^n\} \quad (3)$$
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[3].
- Discrete data $\boldsymbol{\Delta}_n$: which contains timing of events. Note that a generic event E_i^n can occur:
  - 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 paper focuses on the continuous part $\boldsymbol{\theta}_n$.

3 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 $\hat{x}_m^n$:

$$\hat{x}_m^n = \frac{x_m^n - mean(x_m^n)}{stdDev(x_m^n)} \quad (4)$$

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[4].

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 $\tilde{\boldsymbol{\theta}}_n(t')$ as follows:

$$\tilde{\boldsymbol{\theta}}_n(t') = \frac{\sum_{t=0}^{T_n} K(t - t')\, \boldsymbol{\theta}_n(t)}{K(t - t')} \quad (5)$$

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 reduces 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: less memory intensive and faster computations.

4 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 (Lint et al. 2003) 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.

4.1 *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

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

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

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

[4] 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)

(depending on the data set) can be memory and computationally intensive.

4.2 *Polynomial*

The time series is approximated by a Taylor polynomial function up to a fixed degree and the vector of coefficients are retained as representatives for the time series (see Figure 1). 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 $x_1^n(t)$:

$$x_1^n(t) \cong \sum_{\varsigma=0}^{C} c_\varsigma \, t^\varsigma \tag{6}$$

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

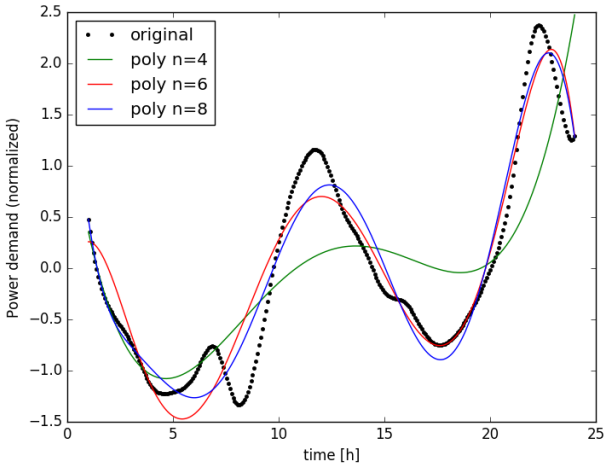

Figure 1. Polynomial approximation of a time series for several polynomial degrees.

4.3 *Chebyshev*

The Chebyshev representation follows exact principle presented above for the Taylor case:

$$x_1^n(t) \cong \left[\sum_{\varsigma=1}^{C-1} c_\varsigma \, T_\varsigma(t)\right] - \frac{1}{2} c_0 \tag{7}$$

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

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

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

4.4 *Legendre*

The Legendre polynomials are polynomials of the following form:

$$\begin{aligned} &P_0(t) = 1 \\ &P_1(t) = t \\ &P_2(t) = (2\,t^2 - 1)/2 \\ &P_3(t) = (5\,t^3 - 3\,t)/2 \\ &P_4(t) = (35\,t^4 - 30\,t^2 + 3)/8 \\ &\cdots \\ &T_{\varsigma+1}(t) = \frac{(2\,\varsigma - 1)\,t\,T_{\varsigma-1}(t) - (\varsigma - 1)T_{\varsigma-2}(t)}{\varsigma} \end{aligned} \tag{9}$$

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

4.5 *Laguerre*

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

$$\begin{aligned} &L_0(t) = 1 \\ &L_1(t) = 1 - t \\ &\cdots \\ &(n+1)L_n(t) - (2n + 1 - t)L_n(t) + n\,L_{n-1}(t) = 0 \end{aligned} \tag{10}$$

4.6 *Hermite*

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

$$\begin{aligned} &H_0(t) = 1 \\ &H_1(t) = t \\ &\cdots \\ &H_{n+1}(t) - 2\,t\,H_n(t) + 2\,n\,H_{n-1}(t) = 0 \end{aligned} \tag{11}$$

4.7 *Discrete Fourier Transform*

Similar to the polynomial representation, the time series is approximated by a Fourier series (see Figure 2) and the series coefficients are retained as representatives for the time series. The Fourier series is as follows:

$$x_1^n(t) \cong \frac{a_0}{2} \sum_{\varsigma=1}^{C} \left(a_\varsigma \cos(\varsigma\,t) + b_\varsigma \cos(\varsigma\,t)\right) \tag{12}$$

4.8 *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 full set of time series and not one time series separately.

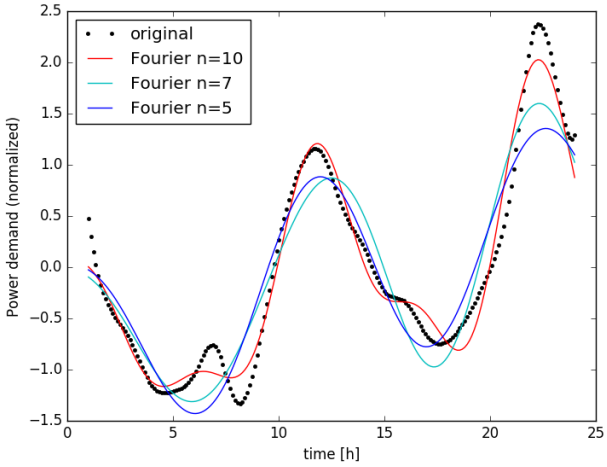

Figure 2. Fourier approximation of a time series for several polynomial degrees.

This is performed for each x_m $(m = 1, \dots, M)$ independently by:

1. Normalizing the data: zero mean and unit variance
2. Resampling the data set so that all time series are 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. 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
5. Projecting the original time series into the eigenvectors space (either reduced or not) and using the projection coefficients as time series representation format.

5 MEASURING SIMILARITY

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 $d(S,Q)$ has to obey the following rules:

$$\begin{cases} d(S,S) = 0 \\ d(S,Q) = d(Q,S) \\ d(S,Q) = 0 \text{ iff } S = Q \\ d(S,Q) \leq d(S,K) + d(K,Q) \end{cases} \quad (13)$$

When dealing with time series, the following two metrics are the most commonly used : Euclidean (Bryant 1985) and Dynamic Time Warping (DTW) (Berndt & Clifford 2004) 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.

5.1 *Euclidean*

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 (see Figure 3) (Chen et al. 2015):

$$d_2(S,Q) = \sqrt{\sum_{t=0}^{T_S} \left(x_1^S(t) - x_1^Q(t)\right)^2} \quad (14)$$

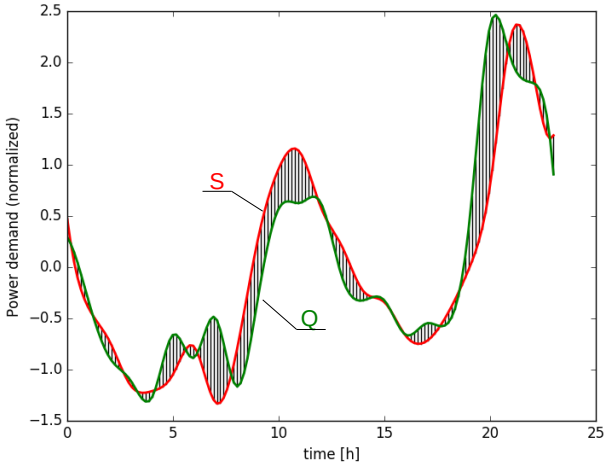

Figure 3. Euclidean distance metric for two time series S and Q. Each black segment represents: $x_1^S(t) - x_1^Q(t)$.

5.2 *Dynamic Time Warping*

This distance can be viewed as a natural extension of the Euclidean distance applied to time series (Berndt & Clifford 2004). Given two univariate time series S and Q having length T_S and T_Q respectively[5]. 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 white line in Figure 4):

$$d_{DTW}(S,Q) = \min\left(\sum_{k=1}^{K} w_k\right) \quad (15)$$

Each element of the path corresponds to a specific black segment in Figure 5.

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

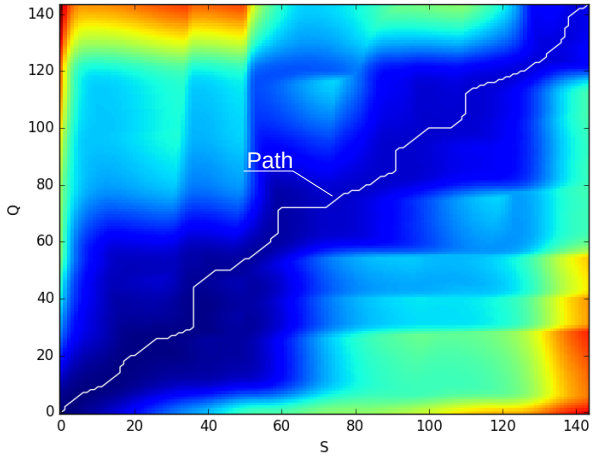

Figure 4. Colored plot of the distance matrix D for the time series S and Q plotted in Figure 5. White line represents the warp path w_k $(k = 1, \dots K)$.

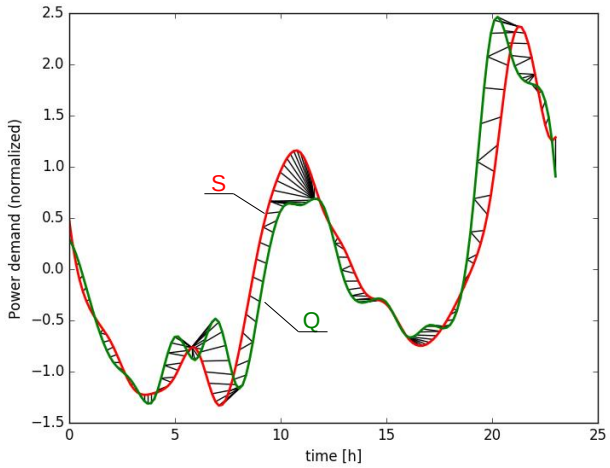

Figure 5. DTW distance metric for two time series S and Q. Each black segment represents an elements $w_k = (d_{i,j})_k$ of the warp path shown Figure 4.

6 DATA MINING TECHNIQUES: CLUSTERING

For the scope of this article we focused on two applications: data searching and clustering. While we believe clustering offers the best tools to "extract information" from data (see first section of this paper), time series searching algorithms allow the user to match time series coming from different data sets.

Data searching algorithms are an important class of data analysis tools that can be very useful to compare and analyze similarities between two time series data sets (e.g., for code validation). In our experience, the two most reliable methods are the following: K-Nearest Neighbors (KNN) and Kd-Tree (Bentley 1975).

From a mathematical viewpoint, clustering (Jain et al. 1999) 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:

$$\begin{cases} C_l \neq \emptyset \ \ \forall l = 1, \dots, L \\ \bigcup_{l=1}^{L} C_l = \Xi \end{cases} \tag{16}$$

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 (Jain et al. 1988), K-Means (Macqueen 1967) and Mean-Shift (Cheng 1995).

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.

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

1. *Approach 1*: 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
2. *Approach 2*: 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 input the solely distance matrix $\Delta = [\delta_{ij}]$ where each element δ_{ij} represent the distance between time series i and j.

6.1 *Approach 1*

The first approach we followed is to perform clustering time series using classical clustering algorithms (e.g., K-Means, Mean-Shift and Hierarchical) not directly on the time series but on the pre-processed data. This can be accomplished when one of the above-mentioned representations is chosen: polynomial, Fourier, or SVD. Each time series is represented as a multi-dimensional vector where each dimension of the vector represents the coefficient of a specific base: polynomial, sine/cosine, and Eigen-vector decomposition respectively.

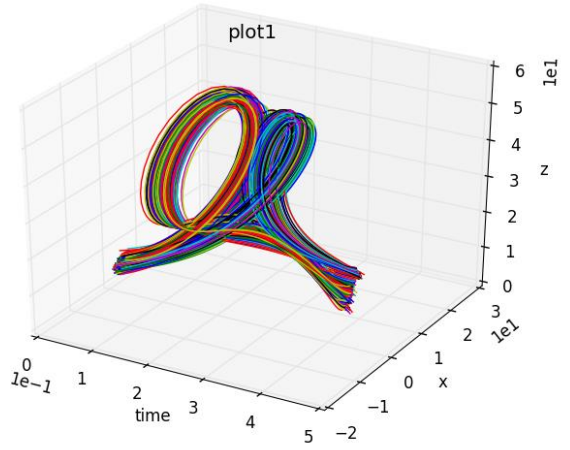

Figure 6. Plot of the time-dependent data set.

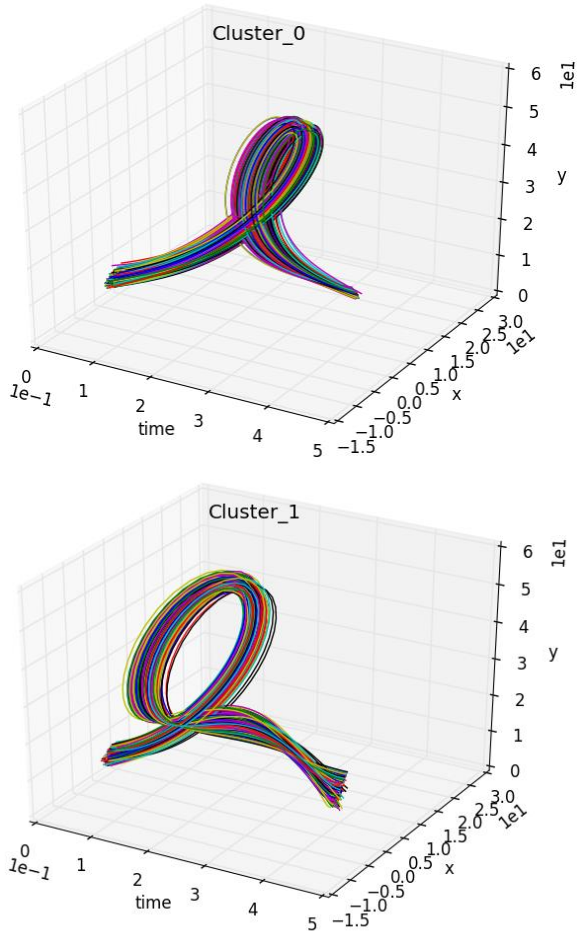

Figure 7. Plot of the time series belonging to each of the two clusters (cluster_0 and cluster_1) using Mean-Shift.

This is possible by performing the following steps in RAVEN (see data set shown in Figure 6):

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. This step might include also time-series synchronization: all time-series are sampled at the same time instants.
3. Convert data: convert the data from time-dependent to static data (e.g., polynomial representation)
4. Cluster data: perform the actual clustering of the data (e.g., through Mean-Shift algorithm) on the converted data. The time series data shown in Figure 6 has been partitioned in two clusters (see Figure 7) and in addition the algorithm has provided the average time series for each cluster (see Figure 8).

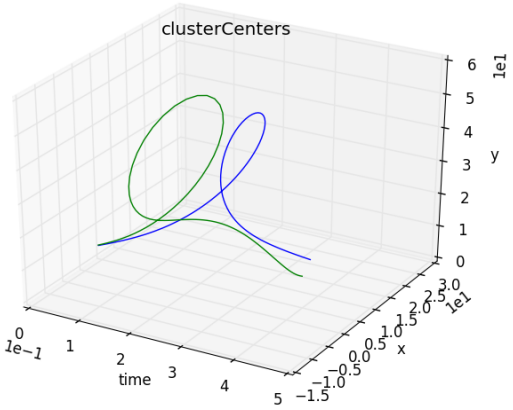

Figure 8. Plot of the cluster centers for each of the two clusters.

6.2 *Approach 2*

This approach 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 (e.g., consider the data set shown in Figure 9)

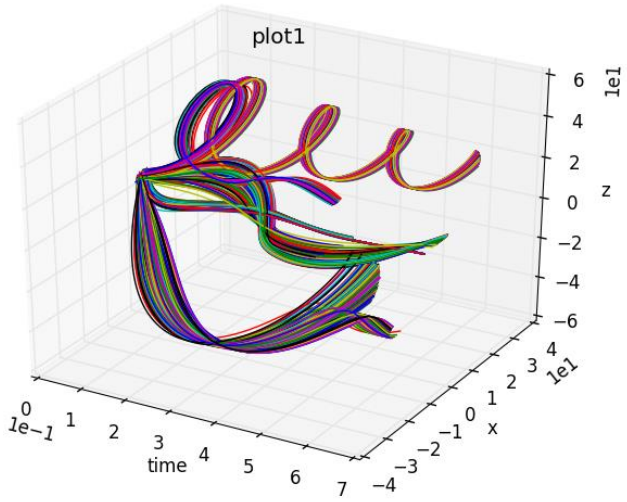

Figure 9. Data set generated by RAVEN containing multiple discountinuites and having variable time length.

2. Re-sample data: perform a temporal re-sampling of the data. 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. The dendrogram on the data set shown in Figure 9 is shown in Figure 10.

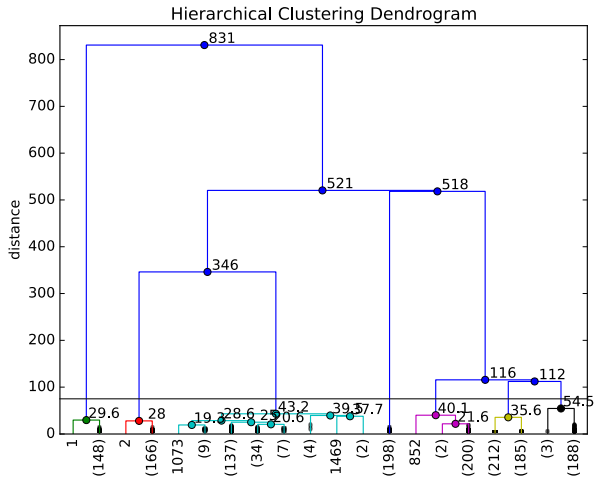

Figure 10. Dendrogram obtained using the RAVEN hierarchical algorithm for the data set shown in Figure 9.

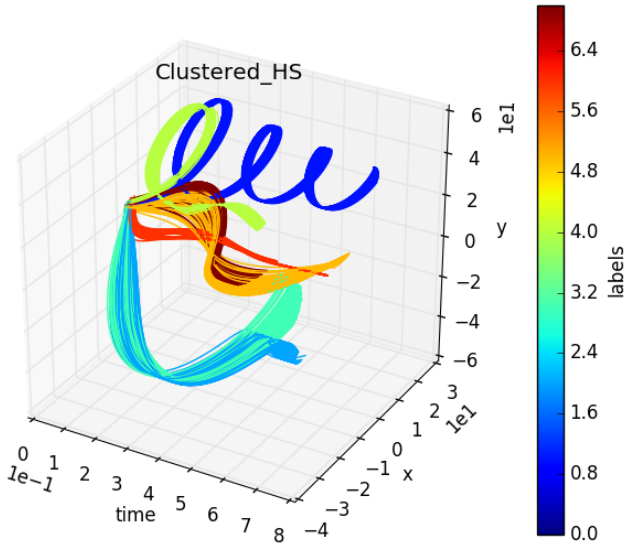

Figure 11. Plot of the clustered data set shown in Figure 9 colored by the cluster labels; each of the 9 clusters (from 1 through 9) correspond a color using Hierarchical algorithm (see Figure 10).

7 SPENT FUEL POOL ANALYSIS

In the framework of the DOE/LWRS/RISMC Project – Industry Application #2, a RELAP5-3D code thermal-hydraulic model of a spent fuel pool (SFP) has been developed (Parisi et al. 2016). The scope of the model has been methodology testing for investigate External Events by deterministic and PRA codes. The RELAP5-3D model is a simplified model of a NPP SFP (see Figure 12).

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.

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. The water volumes of the SFP have been scaled according to the modeled heat load.

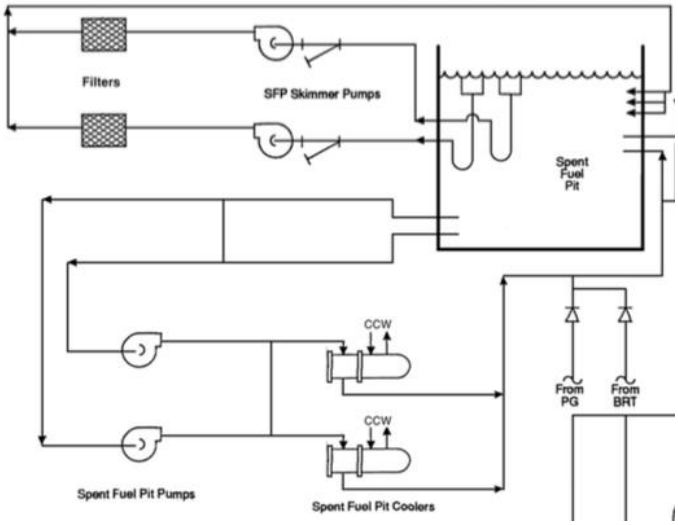

Figure 12. Simplified model of a NPP SFP.

Using RAVEN we performed a Monte-Carlo sampling of the stochastic variables and generated a database of 2000 time series. The sampled stochastic variables are the following:

1. 20600700:6 time of LOCA
2. 20600560:6 pump1 failure time
3. 20600570:6 pump2 failure time
4. 1000101:3 size of the break
5. 20600610:6 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 1.

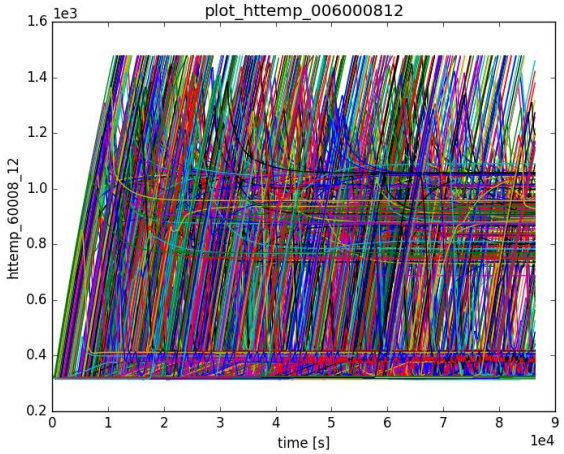

Figure 13. Plot of time series generated by RAVEN-RELAP5.

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. From here we were able to clearly partition the data

set into two clusters (see dendrogram of Figure 14):

a. Cluster 1 (see Figure 15 top): 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 Figure 15, 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 15 bottom): 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 16 top) and Cluster 1_2 (see Figure 16 bottom). 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.

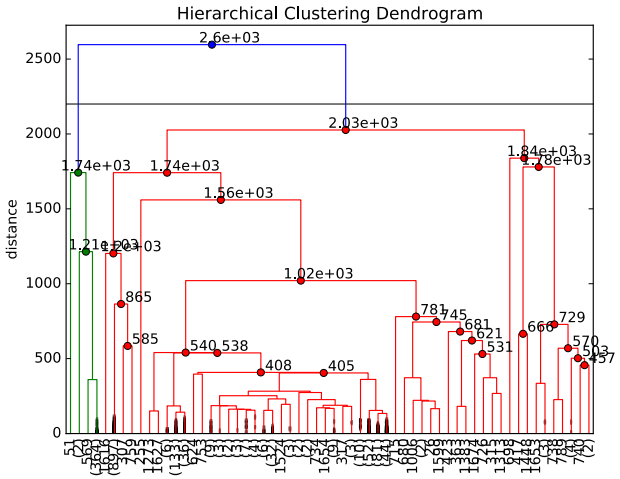

Figure 14. Dendrogram obtained by the hierarchical algorithm.

By analyzing the distribution of the input paramteres for both clusters we were able to deduce that 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.

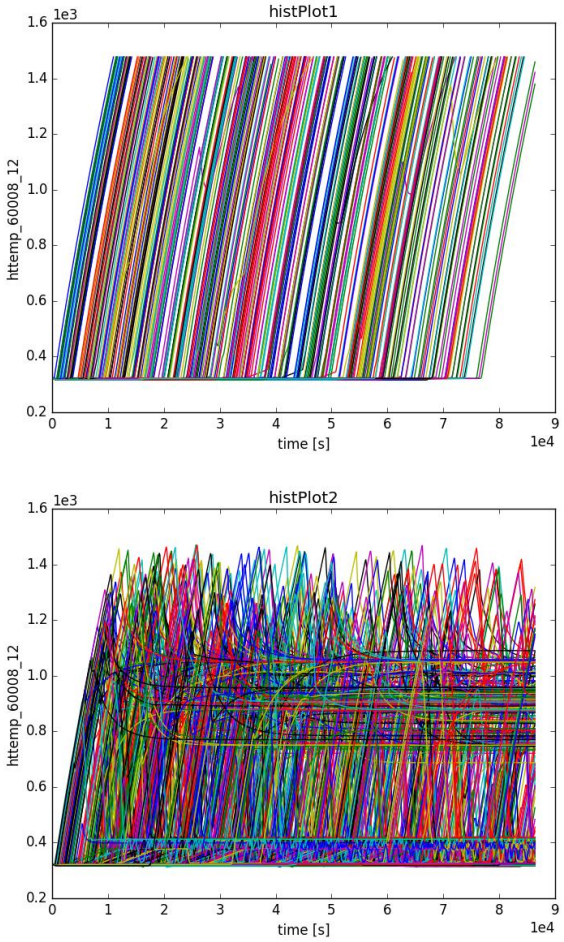

Figure 15. Plot of the time histories contained in Cluster 1 (top) and Cluster 2 (bottom)

Thus, we studied the combination of these three input variables by scatter plotting each simulation run in this 3D input space (see Figure 17). What we obtained is 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 17 of Cluster 1_1 appear to surround the left, right and top boundaries of the cloud of points of Cluster 1_2.

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.

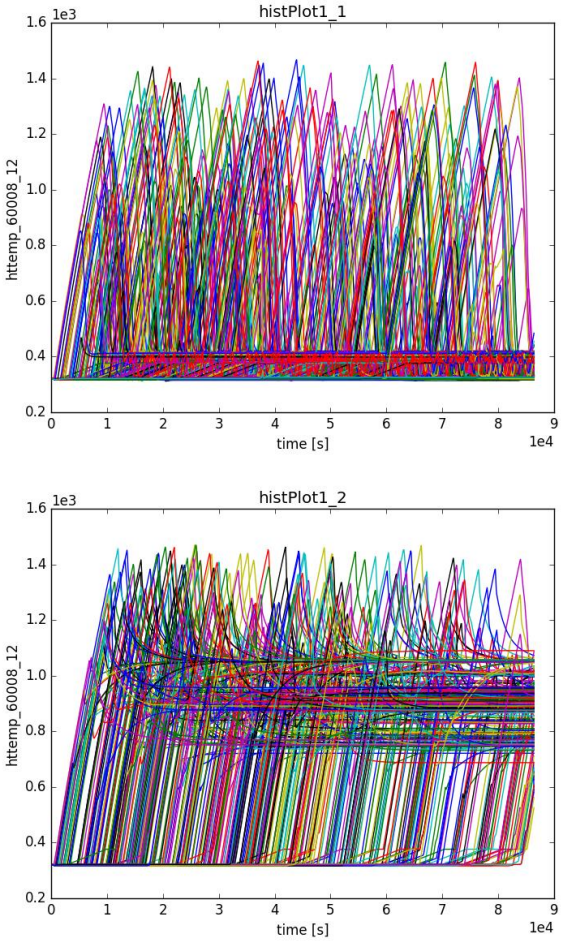

Figure 16. Plot of the time histories contained in Cluster 1_1 (top) and Cluster 1_2 (bottom)

- 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 16 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[6]) obtained by employing clustering in two levels. The scatter plots shown in Figure 17 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.

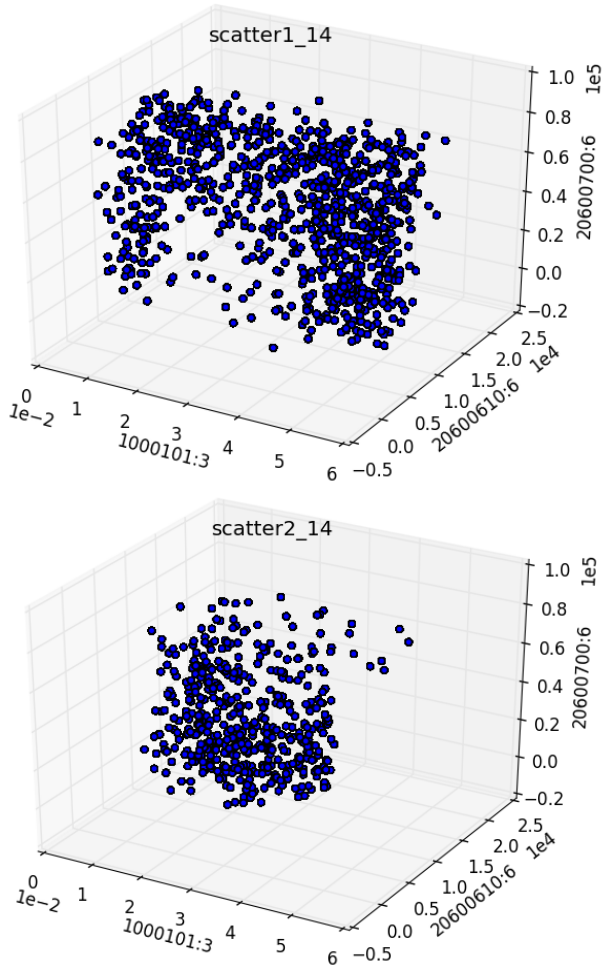

Figure 17. Scatter plot of three stochastic variables for the scenarios in Cluster 1_1 and Cluster 1_2: seal LOCA size (1000101:3), seal LOCA timing (20600700:6) and operator action timing (20600610:6).

8 CONCLUSIONS

In this paper we have presented an overview of methods that can be employed to analyze time dependent data. We cover all main aspects of a typical analyze ranging from data pre-processing, metric choice, data searching and clustering. These algorithms have been developed or are under current development within the RAVEN statistical framework.

REFERENCES

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).

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

D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *AAAI Work-*

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

shop on Knowledge Discovery in Databases, pp. 229-248 (1994).

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

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

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/].

R. O. Gauntt, "MELCOR computer code manual, version 1.8.5", vol. 2, rev. 2. Sandia National Laboratories, NUREG/CR-6119 (2001).

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

E. Hofer, M. Kloos, B. Krzykacz-Hausmann, J. Peschke, and M. Woltereck, "An Approximate Epistemic Uncertainty Analysis Approach In The Presence Of Epistemic And Aleatory Uncertainties," *reliability engineering and system safety*, **77**, pp. 229-238 (2002).

K. S. Hsueh and A. Mosleh, "the development and application of the accident dynamic simulator for dynamic probabilistic risk assessment of nuclear power plants," *reliability engineering and system safety*, **52**, pp. 297-314 (1996).

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

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).

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).

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).

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 (2016).

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

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).

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