J Intell Manuf (2012) 23:481–495
DOI 10.1007/s10845-010-0387-2

Visual analysis of quality-related manufacturing data using fractal geometry

Noa Ruschin-Rimini · Oded Maimon · Roni Romano

Received: 14 January 2009 / Accepted: 8 February 2010 / Published online: 28 March 2010

Abstract Improving manufacturing quality is an important challenge in various industrial settings. Data mining methods mostly approach this challenge by examining the effect of operation settings on product quality. We analyze the impact of operational sequences on product quality. For this purpose, we propose a novel method for visual analysis and classification of operational sequences. The suggested framework is based on an Iterated Function System (IFS), for producing a fractal representation of manufacturing processes. We demonstrate our method with a software application for visual analysis of quality-related data. The proposed method offers production engineers an effective tool for visual detection of operational sequence patterns influencing product quality, and requires no understanding of mathematical or statistical algorithms. Moreover, it enables to detect faulty operational sequence patterns of any length, without predefining the sequence pattern length. It also enables to visually distinguish between different faulty operational sequence patterns in cases of recurring operations within a production route. Our proposed method provides another significant added value by enabling the visual detection of rare and missing operational sequences per product quality measure. We demonstrate cases in which previous methods fail to provide these capabilities.

Keywords Data mining · Sequence mining · Quality engineering · Chaos game representation · Iterated function system

N. Ruschin-Rimini (✉) · O. Maimon · R. Romano
Department of Industrial Engineering, Tel-Aviv University, Tel-Aviv, Israel
e-mail: noarusch@post.tau.ac.il

Introduction

Extensive research had been performed in order to improve manufacturing quality by identifying the causes for product defects. The main methods utilized for the detection of quality-related data in manufacturing processes are Statistical Process Control (SPC) methods and data mining techniques.

SPC methods monitor manufacturing processes in order to identify events which have a significant effect on product quality based on statistical techniques. These methods collect data from samples at various points within the process, and attempt to detect process variations that affect the quality of the end products. Ben-Gal et al. (2003) categorize SPC methods into two major categories: (1) Methods for processing *independent data*, where observations are not interrelated, versus methods for processing *dependant* data. (2) Methods that are *model specific*, requiring a priori assumptions on the process characteristics, usually defined by an underlying analytical distribution or closed-form expression, such as autoregressive integrated moving average (ARIMA), versus methods that are termed *model generic*, which try to estimate the underlying model with minimum a priori assumptions.

Data mining classification techniques are often applied for improving manufacturing quality. The data mining task is to detect the manufacturing parameters (the input attributes) that affect product quality (the target attribute) and to train a classifier from sample data. The trained classifier can then predict product quality based on manufacturing parameters.

The main distinction between classical SPC methods and data mining techniques in relation to improving manufacturing quality is that SPC methods often aim to monitor the process and not to identify the relationship between the target attribute and the input attributes. Data mining is considered "a secondary data analysis of large databases"

(Hand 1998), since the primary purpose of the database is data collection. Moreover, the volume of the collected data makes it impractical to explore and detect complex relations between the parameters of different processing steps using standard statistical procedures (Rokach et al. 2008).

Many classification data mining problems for improving manufacturing quality analyze the effect of operation settings on product quality, e.g., Rokach and Maimon (2006) and Rokach (2008). Da Cunha et al. (2006) were the first to analyze the effect of the production sequence on product quality. They suggest that the analysis of production sequence is significant for assemble-to-order production strategies. According to this type of strategy, the same product components can be used to configure a variety of end products. The end products are modular, flexible and customized according to the customer's request. As result of the growing trend of mass customization, it is impossible for the manufacturer to build all possible configurations in advance. Accordingly, assembly is performed only when the confirmed customer order is received. Nevertheless, in order to compete with other manufacturers, the assembly lead-time should be as short as possible and any rework due to quality problems should be avoided. Since the quality tests of the components can be performed in advance, the focus should be on the quality of the final assembly operations. Da Cunha et al. (2006) provide an industrial example of electrical wire harnesses in the automobile industry. There are millions of different wire harnesses for a unique car model, mainly because wire harnesses control many functions (such as the electrical windows) and every function has different versions depending on various parameters such as engine type.

The fractal visual analysis approach presented in this paper uses production data to determine the sequence of final assembly operations that minimizes the risk of producing faulty products. It extends the research of Da Cunha et al. (2006) by providing a visual application, enabling a production engineer to visually detect operational sequence patterns that cause product defects. Moreover, it overcomes the limitations of existing methods such as the *n*-gram approach utilized by Da Cunha et al. (2006), by detecting faulty operational sequence patterns of any length, without predefining the pattern length, and by distinguishing between different faulty operational sequence patterns, even in cases where operations are performed more than once in the same manufacturing process.

The proposed approach is based on an Iterated Function System (IFS) for producing a fractal representation of manufacturing processes. For demonstration purposes, a software application is developed for visual detection of faulty operational sequence patterns. The application is aimed at production engineers, and comprises such features as color codes and zoom functions, in order to facilitate the visual analysis process. Visual data analysis usually allows faster data exploration and often provides better results, especially in cases where automatic algorithms fail (Keim 2002). Since the proposed method and application are aimed at production engineers, it is of great importance that the faulty operational sequence pattern detection process and results be intuitive and easy to understand.

Review of existing methods for discovering production sequences affecting product quality

In this section we review existing methods for discovering operational sequences that affect product quality. We present the main limitations of each existing method. On the proceeding sections, we demonstrate how the proposed method overcomes these limitations.

The n-gram approach

Da Cunha et al. (2006) concluded that in order to identify the production sequences causing faulty products, the challenge was to find a method for restructuring the input data. They suggest utilizing the bi-gram representation method in order to represent the production sequences. The bi-gram is a subgroup of the *n*-gram representation method, and a common and proven method used for text classification problems (Cavner and Trenkle 1994). A character *n*-gram is simply any sequence of *n* consecutive letters taken from a text. As an example of bi-grams, consider a two letter wide sliding window moving from the beginning to the end of a text, one letter at a time. The content of the window at each step is a bi-gram.

Consider the production route illustrated in Fig. 1. It comprises five operations; afterwards an inspection test is carried out in order to determine the quality of the product. In this case the product was identified faulty; hence rework of operation 1 needs to be done. Afterwards another inspection test takes place, in order to examine the end product quality. The production route 31452 transformed by bi-gram representation is therefore the set of characters: {B_3, 3_1, 1_4, 4_5, 5_2, 2_F}. Character B_3 indicates that operation 3 is the first operation, and character 2_F indicates that operation 2 is the last operation of the manufacturing process. After utilizing the bi-gram approach for representation, Da Cunha et al. 2006 utilized an association rules algorithm (Agrawal and Srikant 1994) in order to extract classification rules, analyzing the effect of production sequence on the quality of the product.

The bi-gram representation method has several drawbacks:

A. It is restricted to finding two length sequence patterns. More generally, the *n*-gram approach predefines a specific sequence pattern length.

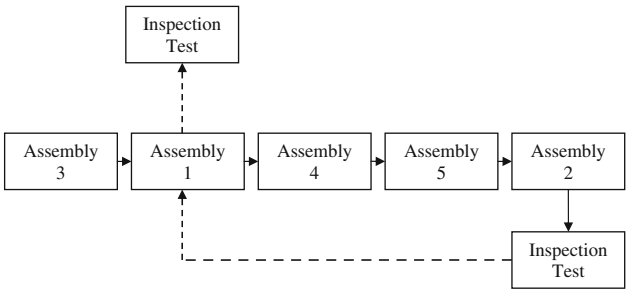

Fig. 1 Illustration of a product route

B. It assumes that a normal operation can be performed at most once per product, i.e. a manufacturing process such as 9-2-7-1-5-3-6-9 is not allowed. In case an operation is performed more than once in a manufacturing process, the bi-gram method is incapable of distinguishing between the different occurrences of the same operation. For example, if the following two unrelated operational sequence patterns: 3-6-9 and 9-2-7-1 are responsible for product defects, the bi-gram representation method detects the faulty operational sequence patterns 3-6, 6-9, 9-2, 2-7, 7-1, with no additional information regarding the fact that these are two unrelated operational sequence patterns, each consisting of a different occurrence of operation 9. In order for the bi-gram method to reveal the full operational sequence patterns, another step is required, examining all permutations of 3-6-9-2-7-1 using association rules.
C. Rokach et al. (2008) provide another limitation of the bi-gram method. They state that although bi-gram can be extended to detect patterns whose length is greater than two operations (i.e. *n*-gram), it is not efficient for long sequences, because it requires a much higher dimensionality. Increasing the dimensionality may cause some problems to data mining algorithms – a phenomenon known as the "curse of dimensionality".

The method we propose in this work overcomes these disadvantages, while providing comparable results.

Hidden Markov model

The Hidden Markov Model (HMM) is a common and proven machine learning method in which the system being modeled is assumed to be a Markov process with unknown parameters. The challenge is to determine the hidden parameters from the observable ones.

HMM models are especially known for their applications in domains such as speech recognition and biological sequence analysis. We did not find any work that implements HMM for the detection of faulty operational sequence patterns in manufacturing processes, although the HMM method could be utilized to perform this task.

The HMM method has several drawbacks:

A. Adapting this method for the detection of faulty operational sequence patterns in manufacturing processes requires a complex task of selecting the model structure (i.e. defining the states and the available transitions). Rokach et al. (2008) state that this is especially true in cases where the model aims to discover long sequence patterns.
B. The model is considered a "black box" in the sense that its meaning is inexplicable. Hence this model does not provide a production engineer tools for comprehending the causes for product defects.

The method proposed in this paper enables production engineers to visually identify and therefore comprehend the production sequences causing product defects.

Iterated function system (IFS)

IFS was originally developed as a method for constructing fractals as discussed in detail in Barnsley (1988). Other main applications of IFS are image compression (Barnsley and Hurd 1993) and analysis of genomic sequences (Jeffrey 1990).

We utilize IFS in order to represent manufacturing processes as fractals. For this purpose, IFS is used as an iterative contractive mapping technique that represents a sequence as vectors in $\Re$. This type of transformation of a sequence, also known as the 'Chaos Game Representation', produces a self similar fractal formed graph, and has the following significant properties:

1. An IFS of a sequence provides a unique representation of it. It can be seen as the 'fingerprint' of a sequence. Every point on the graph achieved via IFS represents uniquely all sequence history up to this point. Consequently, an IFS representation comprises all information regarding all subsequences existing in a sequence.
2. The source of the sequence can be inversed from the graph.

Barnsley (1988) provides the following definition for IFS:
An Iterated Function System consists of a complete metric space *(X, d)* together with a finite set of contraction mappings $w_n : X \to X$, with respective contractivity factors s_n, for $n = 1, 2, \ldots, m$.
A mapping $w_i(x)$ is contractive in *(X, d)*, if $d(w_i(y), w_i(z)) \leq s_i \cdot d(y, z) \forall y, z \in X$ for some contractivity factor $0 < s_i < 1$.

Barnsley (1988) uses the following general notation for IFS transformation in $\Re^2$:

$$w_i(x) = w_i\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix}$$

Varying the IFS parameters produces a variety of IFS transformations.

As an example, IFS transformation is utilized for visual analysis of genomic sequences since Jeffrey's foundation essay in 1990. This can be accomplished by using the following IFS:

$$w_1(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$w_2(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -0.5 \\ 0.5 \end{bmatrix}$$

$$w_3(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}$$

$$w_4(x) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}$$

In order to apply this IFS transformation for visual analysis of genomic sequences, one follows the following procedure:

1. Associate each nucleotide (A,C,G,T) with one of the contractive mappings $w_i(x), i \in \{1, 2, 3, 4\}$. For example: $A = w_1(x), C = w_2(x), G = w_3(x), T = w_4(x)$.
2. Accordingly, represent the genomic sequence of length N as a sequence of N corresponding contractive mappings $\{w_{i(n)}(x) : i \in \{1, 2, 3, 4\}$ and $n = 1, 2, \ldots, N\}$.
3. Plot x_0, an arbitrary point in $\Re^2$.
4. Recursively apply each of the N contractive mappings $w_{i(1)}(x), w_{i(2)}(x), \ldots, w_{i(N)}(x)$ by their sequence order as follows: apply contractive mapping $w_{i(1)}(x)$to point x_0 in order to achieve point x_1, then apply contractive mapping $w_{i(2)}(x)$ to point x_1 in order to achieve point x_2, etc. More generally, $x_n = w_{i(n)}(x_{n-1})$ for $n = 1, 2, 3, \ldots, N$ and $i \in \{1, 2, 3, 4\}$. This results in a sequence of N points in $\Re^2$ $\{x_n : n = 1, 2, 3, \ldots, N\}$.

The sequence of points produces a square-formed fractal-like graph that enables visual analysis of genomic sequences. If the original sequence of four category types would be uniformly random and long enough, this IFS transformation would produce an equally filled in square. Since the transformed sequence is not uniformly random, the graph reveals its underlying correlations by varying densities of points in different zones.

For illustration purposes, Fig. 2 shows the results of implementing the above IFS transformation on amylase enzyme.

This program was coded in *MATLAB*^{TM} language.

The interpretation of the graph is done by the addresses of points on a fractal (Barnsley 1988). Figure 3 shows the addresses for this specific IFS. The addresses are utilized for the purpose of inversing and analyzing the subsequences that led to a specific point. The length of the inversed subsequence depends on the address resolution. In theory, we can display infinite address resolutions or subsequence lengths. In practice this is limited by display resolution.

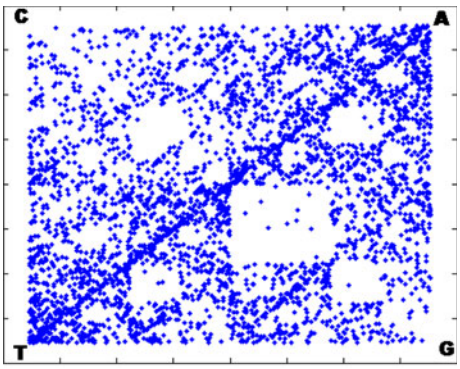

Fig. 2 IFS transformation of amylase enzyme

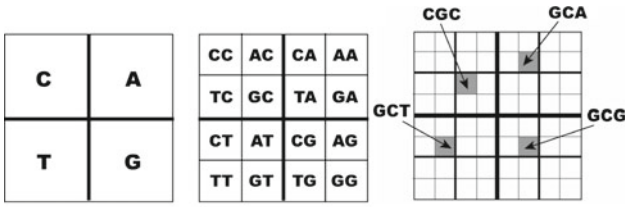

Fig. 3 IFS addresses

Based on the IFS addresses, one is able to provide the following analysis of the amylase enzyme IFS transformation presented in Fig. 2: The largest empty square formed area right and below the center diagonal line and other empty areas indicate that the subsequence 'CG' is rare in amylase enzyme. Moreover, the A to T diagonal indicates that various combinations of A and T are frequent in amylase enzyme.

This approach is restricted to visual analysis of sequences consisting of four category types. Nevertheless, the number of category types in production sequences is determined by the number of operation types and cannot be predefined. Consequently, the proposed algorithmic framework enables us to visually analyze sequences consisting of any number of categories, yet to keep the representation in $\Re^2$ for computer-based visual analysis.

Algorithmic framework

Overview

In order to enable visual analysis of the effect of production sequence on the product quality measure, we suggest using the framework presented in Fig. 4. The process consists of four phases:

1. *Production sequence representation:* A domain-specific task designed to represent each set of manufacturing processes that share a common quality measure as one long string of tokens.
2. *Production sequence transformation:* Applying an IFS scheme with circle transformation to each established string of tokens in order to receive a fractal-like graph that enables visual analysis of production sequences.
3. *Production sequence patterns detection:* Visual detection of operational sequence patterns for each group of manufacturing processes that share a common product quality measure. The process is based on interpretation of the graph via addresses of points on fractals.
4. *Classifiers selection:* A heuristic for filtering the detected operational sequence patterns in order to select the ones that cause product defects. The operational sequence patterns that influence product quality are selected as classifiers.

The following subsections describe in detail each of the above phases.

Production sequence representation

The manufacturing process of each product is represented as a string of tokens, each token representing a different operation activity. For instance, the production sequence 3-1-4-5-2 illustrated in Fig. 1 is represented as the string "3 1 4 5 2".

Next, we group manufacturing processes by their quality measure (for example passed /failed quality control) or by the required rework operation. We concatenate the manufacturing process representations of each group to form one string, while adding a delimiter between manufacturing processes in order to differentiate between them.

Production sequence transformation

We utilize an IFS with circle transformation, based on the IFS developed by Weiss (2008). This unique IFS transformation provides the flexibility of analyzing sequences consisting of any number of category types, yet keeping representation in $\Re^2$ in order to enable visual analysis (Weiss 2008).

Following is a description of the suggested IFS transformation for a sequence consisting of m types of categories:

$$w_i(x) = \begin{bmatrix} \alpha_i & 0 \\ 0 & \alpha_i \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \beta_i \\ \delta_i \end{bmatrix} \quad \text{for } i = 1, 2, \ldots, m$$

When

$$\beta_i = \cos\left(i \cdot \frac{2\pi}{m}\right) \qquad \text{for } i = 1, 2, \ldots, m$$

$$\delta_i = \sin\left(i \cdot \frac{2\pi}{m}\right) \qquad \text{for } i = 1, 2, \ldots, m$$

$$\alpha_i = \alpha \quad \forall i \qquad \text{for } i = 1, 2, \ldots, m$$

It is also required that α fulfills: $\frac{\alpha}{1-\alpha} < \sin\left(\frac{\pi}{m}\right)$. For proof see Appendix A.

We introduce an explanation of the steps for applying IFS with circle transformation to a sequence of length N consisting of m category types $C = c_1, c_2, \ldots, c_m$.

1. Associate each category type $c_1, c_2, \ldots, c_m$ with one of the contractive mappings $w_i(x), i \in \{1, 2, \ldots, m\}$.
2. Accordingly, represent the sequence of length N consisting of m category types as a sequence of N corresponding contractive mappings $\{w_{i(n)}(x) : i \in \{1, 2, \ldots, m\}$ and $n = 1, 2, \ldots, N\}$.
3. Plot x_0, an arbitrary point in $\Re^2$.
4. Recursively apply each of the N contractive mappings $w_{i(1)}(x), w_{i(2)}(x), \ldots, w_{i(N)}(x)$ by their sequence order as follows: apply contractive mapping $w_{i(1)}(x)$to point x_0 in order to achieve point x_1, then apply contractive mapping $w_{i(2)}(x)$ to point x_1 in order to achieve point x_2, etc. More generally, $x_n = w_{i(n)}(x_{n-1})$ for $n = 1, 2, 3, \ldots, N$ and $i \in \{1, 2, 3, \ldots, m\}$. This results in a sequence of N points in $\Re^2$ $\{x_n : n = 1, 2, 3, \ldots, N\}$.

The sequence of points produces a circle-formed fractal-like graph that enables visual analysis of sequences. If the original sequence of m category types is uniformly random and long enough, this IFS transformation results in a graph of a self-similar fractal consisting of m equally filled and disconnected circles, each circle at each resolution also comprising m circles. If the sequence is not uniformly random, the graph will reveal its underlying correlations by varying densities of points in different zones.

For example, Fig. 5 shows the result of transforming a uniformly random sequence of length $N = 20{,}000$ consisting of m =26 category types. In this example, we chose $\alpha = 0.08$. Note that α fulfills the requirement mentioned above. The Figure presents the 1st and 2nd resolutions of the graph. By

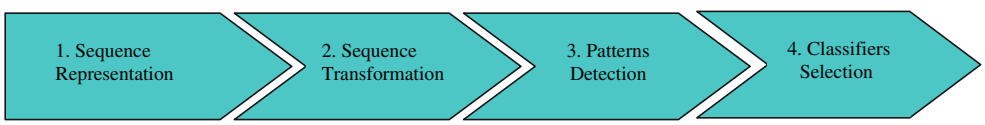

Fig. 4 The process overview

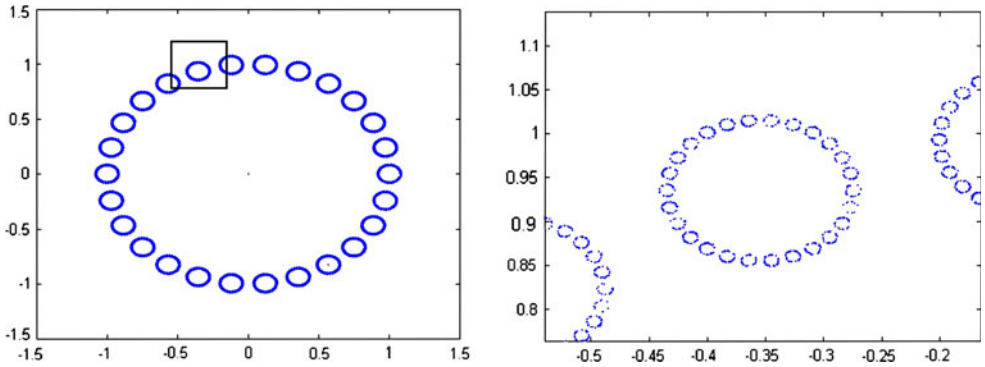

Fig. 5 IFS circle transformation of a random sequence of m = 26 categories, 1st and 2nd resolutions

examining the 2nd resolution (zooming into the upper left area) we can conclude that the sequence was only pseudo-random, since the circles are not equally filled. This program was coded in *MATLAB*^{TM} language.

IFS of circle transformation is separately applied to each of the concatenated strings of tokens established in the first stage of the process.

Production sequence pattern detection

We refer to the term sequence pattern as a frequent subsequence existing in a set of sequences sharing the same target class. The frequency level is defined by a threshold. More specifically, we aim to detect operational sequence patterns in every set of manufacturing processes sharing the same product quality measure, in order to reveal the operational sequences causing each type of product defect.

The fractal graph is interpreted by utilizing the address of points on a fractal (Barnsley 1988). The location of every point on the graph holds the information of the whole sequence up to this point. This allows us to translate areas on the graph such as empty areas, areas of relatively low density, and areas of relatively high density, into missing subsequences, rare subsequences, and frequent subsequences, respectively.

As mentioned in Section "Production Sequence Transformation", the IFS of circle transformation results in a graph of a self-similar fractal consisting of m disconnected circles, each circle at each resolution also comprising m circles. Since we associated every category, i.e. every operation type, to a certain contractive mapping, the address of every circle represents a category type, i.e. an operation activity. More specifically, category c_i, which was associated with mapping $w_i(x)$, is the address of the circle centered at $\begin{bmatrix} \beta_i \\ \delta_i \end{bmatrix}$. The address length is determined by the graph resolution. Figure 6 demonstrates addresses of 1st, 2nd and 3rd resolutions for IFS of circle transformation. In the example, we show the result of transforming a uniformly random sequence of length N=15,000 consisting of m=9 categories ($c_1 = 1$, $c_2 = 2, \ldots, c_8 = 8$, $c_9 = 0$). The IFS parameter $\alpha = 0.08$ was chosen.

This program was coded in *MATLAB*^{TM} language.

Defining the addresses of points on the graph enables us to suggest the following algorithm for the detection of operational sequence patterns affecting product quality:

1. Detect an area of relatively high density on the 1st resolution of the fractal graph, i.e. one of the m circles comprising a high percentage of points. A circle of relatively high density is defined by a certain percentage threshold.
2. Drill into the relevant circle.
3. Detect an area of relatively high density on the next resolution, i.e. one of the m circles comprising a high percentage of points.
4. Drill into the relevant circle.
5. Repeat steps 3 and 4 until the relevant circle contains points almost uniformly distributed between approximately m circles, with no area of relatively high density. This is where the sequence pattern ends.
6. Compute the address of the relevant circle location in order to recover the operational sequence pattern. For example, if the sequence pattern ends at the 4th resolution, we conclude that the sequence pattern's length is 3. We therefore recover the three length address of the relevant circle from the 3rd resolution.
7. Repeat steps 1–6 for a different area of relatively high density in order to reveal another operational sequence pattern.
8. End after exploring all areas of relatively high density, i.e. after all operational sequence patterns have been revealed.

During this stage we therefore apply the Operational sequence Pattern Detection Algorithm separately to each of the fractal graphs established in the prior stage of the process. It should be noted that we could also use the same algorithm

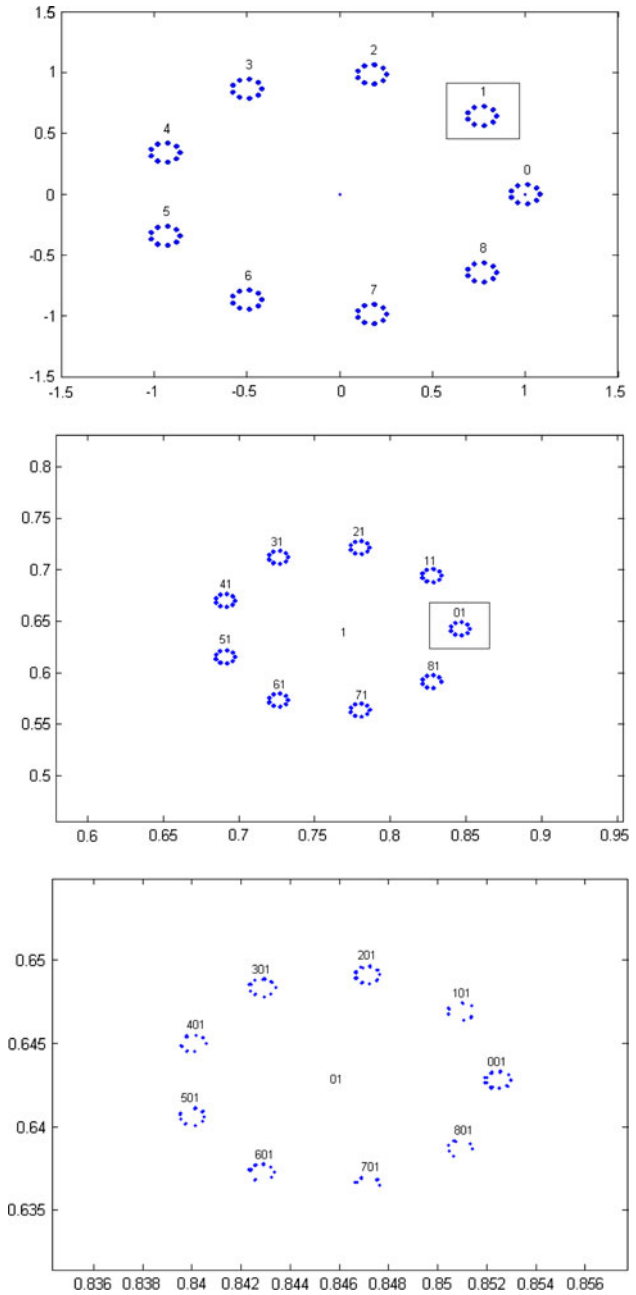

Fig. 6 Addresses of points for 1st resolution, 2nd resolution (zooming into circle address 1) and 3rd resolution (zooming into circle address 01) resulting from IFS of circle transformation

to explore *empty areas* and *areas of relatively low density* for detecting missing and rare operational sequences per product quality measure, respectively.

Classifiers selection

Applying stages 1–3 of the algorithm results in a set of operational sequence patterns detected for each product quality measure. Next, we filter the operational sequence patterns that have been detected by using a simple heuristic comprising the following steps:

1. Select one of the operational sequence patterns detected for a certain product quality measure.
2. Check whether the operational sequence pattern was detected for other product quality measures. If the sequence pattern was detected exclusively for the certain product quality measure, go to step 3. Otherwise go to step 4.
3. Conclude that the selected operational sequence pattern affects the specific product quality measure and therefore it can be selected as a classifier.
4. Conclude that the selected operational sequence pattern does not affect the product quality measure and therefore it cannot be used as a classifier.
5. Repeat steps 1-4 for another detected operational sequence pattern.
6. End after exploring all operational sequence patterns detected for all product quality measures.

It should be emphasized, that a more accurate approach would be to apply any data mining classification algorithm at this stage. For this purpose, each detected operational sequence pattern represents a different feature. The feature value is 1 if the corresponding operational sequence pattern exists in a manufacturing process; otherwise it is 0. The target class is the product quality measure. A classification algorithm such as C4.5, which creates a classification decision tree (Quinlan 1993), could be utilized at this stage. Nevertheless, since we aim to provide manufacturing engineers a practical tool that requires no background in data mining or statistics, our simple heuristic is sufficient.

Faulty operational sequence detection application

We propose a software application for the purpose of improving the process of visual data exploration. The software application consists of the following main features:

1. *Color code*: The detection of areas of relatively high density should be immediately visible to the production engineer. We therefore provide a color code for relatively high density areas.
2. *Zoom function*: In order to explore areas of relatively high density until the full operational sequence pattern is revealed, it is necessary to zoom into the relevant circles. The application offers this functionality.
3. *Method for recovering the sequence pattern address*: The application presents the corresponding addresses (i.e. the corresponding operation activities) near each circle as a part of the graph.

The software application was developed in *MATLAB*^{TM} language

Experimental study

Overview

This study demonstrates that the proposed method matches the performance of the previous work of Da Cunha et al. (2006) given the same conditions and outperforms it when the complexity of the problem increases, such as what might occur in more realistic conditions. We present cases in which our method was exclusively able to reveal the full operational sequences that affect product quality.

Experiment setup

Quality can be measured in many different ways. Usually the quality of product batches is measured and not that of a single product. The quality measure can either have nominal values (such as "passed" or "not passed") or continuously numeric values. Even if the measure is numeric, it can still be reduced to a sufficiently discrete set of interesting ranges (Rokach 2008).

The experiment assumes that the quality measure has nominal values. The goal is to detect patterns in operational sequences that affect the product quality measure. More specifically, we wish to detect operational sequence patterns of any length that cause product defects requiring a specific rework operation.

For this purpose, we represent each manufacturing process of each product as a string of tokens, each token representing a different operation activity.

We generate three manufacturing datasets in order to examine if the suggested method is capable of visually detecting operational sequence patterns of different lengths that affect product quality. In order to prove that the method is effective in identifying the causes of product quality issues in the presence of noise, deriving in part from unexpected events, the data includes random events that occur in production systems.

Following are the assumptions of the generated datasets:

All three generated datasets share the following assumptions:

1. There are 8 types of operation activities. We represent operation activity types by tokens 0,1,2,…,7.
2. An operation activity can be performed more than once in a product route.
3. 1,000 product routes were randomly generated per dataset.

Dataset 1: Faulty manufacturing processes requiring rework of operation 4

1. We determined that all manufacturing processes of the generated dataset resulted in faulty products requiring the rework of operation 4.
2. The following systematic faulty operational sequence patterns were planted in the dataset:
   a. Faulty operational sequence 4-7-2 (i.e. operation 4 followed by operation 7, then followed by operation 2) was randomly distributed in 10% of the faulty manufacturing processes requiring rework of operation 4. The sequence pattern's location in every manufacturing process was randomly chosen with one exception: the pattern was never planted at the beginning of a manufacturing process.
   b. Faulty operational sequence 5-0-4 was randomly distributed in 10% of the manufacturing processes requiring rework of operation 4. In these processes, the sequence pattern was located at the beginning of the manufacturing process.

Note that operation 4 was deliberately chosen to appear in both faulty operational sequence patterns. This was done in order to examine whether the suggested method could still distinguish between the two different operational sequence patterns and not mistakenly detect one faulty operational sequence pattern of 5-0-4-7-2.

Dataset 2: Faulty manufacturing processes requiring rework of operation 1

1. We determined that all manufacturing processes of the generated dataset resulted in faulty products requiring the rework of operation 1.
2. The following systematic faulty operational sequence pattern was planted in the dataset: faulty operational sequence 2-2-1 (i.e. operation 2 followed by a second operation 2, followed by operation 1) was randomly distributed in 10% of the faulty manufacturing processes requiring rework of operation 1. The sequence pattern's location in every manufacturing process was randomly chosen with one exception: the sequence pattern was never located at the beginning of a manufacturing process.

Dataset 3: Manufacturing processes of products that passed quality control

1. We determined that all manufacturing processes of the generated dataset resulted in products that passed quality control.
2. No operational sequence pattern was planted in the dataset

Algorithmic framework implementation

Production sequence representation

The manufacturing processes generated are represented as strings of tokens, each token representing a different operation activity (tokens 0,1,2,…,7 represent all eight types of operation activities). Since each dataset consists of manufacturing processes sharing a common product quality measure, we concatenate all manufacturing processes per each dataset to form one string. We choose token '8' as the delimiter. We denote the concatenated strings as $S_i, i = 1, 2, 3$ corresponding to datasets 1, 2 and 3 respectively.

Production sequence transformation

We transform each string $S_i, i = 1, 2, 3$ using an IFS of circle transformation. This transformation results in three fractal-like graphs, i.e. a graph for each dataset.

Faulty production sequence pattern detection

We explore each generated fractal graph in order to detect operational sequence patterns per product quality measure.

This phase results in a set of operational sequence patterns for each dataset, i.e. for each product quality measure.

For illustration purposes, we demonstrate a full visual pattern detection process for dataset 1.

Figure 7 shows the 1st resolution of the graph derived from transforming dataset 1 using an IFS of circle transformation. Every circle represents an operation activity. The mid-right circle represents operation activity 0. Then, moving counter-clockwise to the following circles, operations activities 1-7 and the delimiter 8 are represented respectively.

The color code we chose for this experiment was red for points located in relatively high density areas, and blue for the rest of the points. We define areas of relatively high density in this experiment as circles of 3rd resolution that consist of more than 0.5% of the points. We can easily visually detect three areas of relatively high density located in circles 0, 2 and 4 (See Fig. 7).

Table 1 summarizes the steps for exploring a complete faulty operational sequence pattern for dataset 1.

Similarly, we visually detect the remaining operational sequence patterns of all three datasets. A full demonstration can be found in Appendix B.

Classifiers selection

We filter the operational sequence patterns detected in the three datasets by implementing the suggested heuristic.

All production sequence patterns detected (presented in Table 1 and in Appendix B) were unique for each product quality measure. We therefore conclude that all detected operational sequence patterns affect product quality. Explicitly, all detected operational sequence patterns are selected as classifiers.

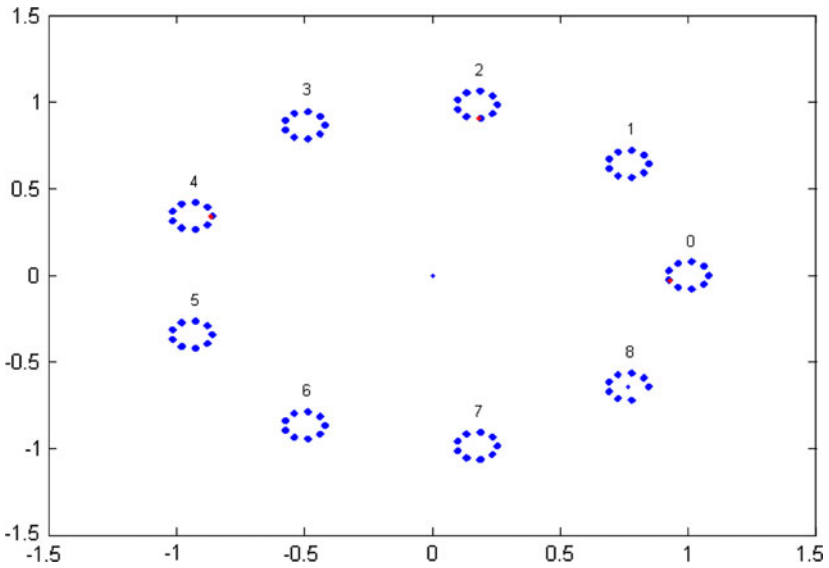

Fig. 7 IFS graph, 1st resolution. Note that three areas of relatively high density marked by *color code red* are detected in circles 0, 2 and 4. (Color figure online)

Table 1 Exploring an operational sequence pattern from IFS graph of dataset 1

Process step	Action	Result	Conclusion	Corresponding figure
1	Explore 1st resolution of dataset 1 IFS graph	One relatively high density area is detected in circle address 2.	Partial sequence pattern detected: 2	Fig. 8
2	Zoom into circle address 2	An area of relatively high density is detected in circle address 72.	Partial sequence pattern detected: 7-2	Fig. 9
3	Zoom into circle address 72	An area of relatively high density is detected in circle address 472.	Partial sequence pattern detected: 4-7-2	Fig. 10
4	Zoom into circle address 472	1. No area of relatively high density is detected 2. Circle address 8472 is nearly empty	1. Full sequence pattern detected: 4-7-2 2. Sequence 8-4-7-2 is rare	Fig. 11

Interpretation: The first faulty operational sequence pattern we discover for dataset 1 is therefore 4-7-2. Note that circle address 8472 is nearly empty (See Fig. 11), i.e. sequence 8-4-7-2 is rare. Since token 8 represents the delimiter, it represents the beginning of a manufacturing process in this case. We therefore conclude that operational sequence pattern 4-7-2 may be the cause of product defects requiring rework of operation 4 in cases where the sequence pattern is not performed at the beginning of a manufacturing process

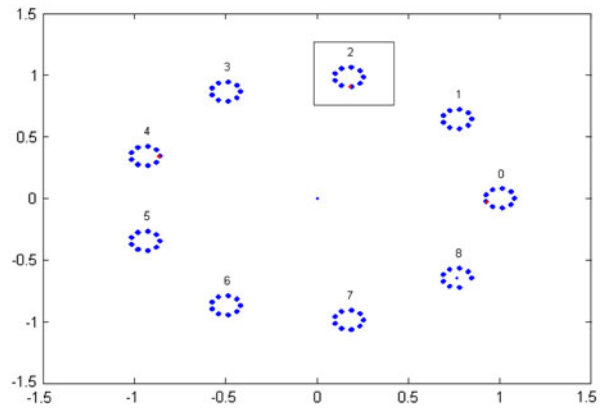

Fig. 8 IFS graph, 1st resolution. One area of relatively high density is detected in circle address 2

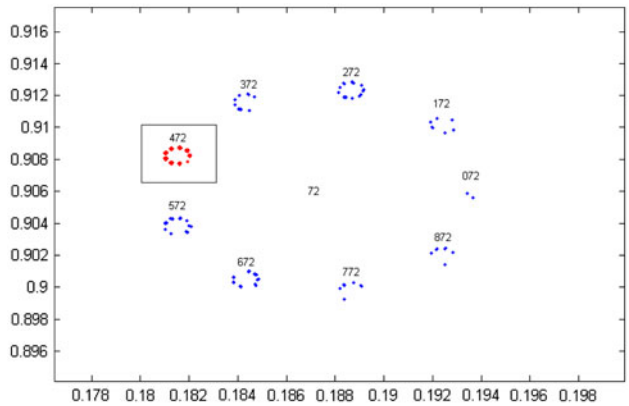

Fig. 10 IFS graph, zooming into circle address 72. An area of relatively high density is detected in circle address 472

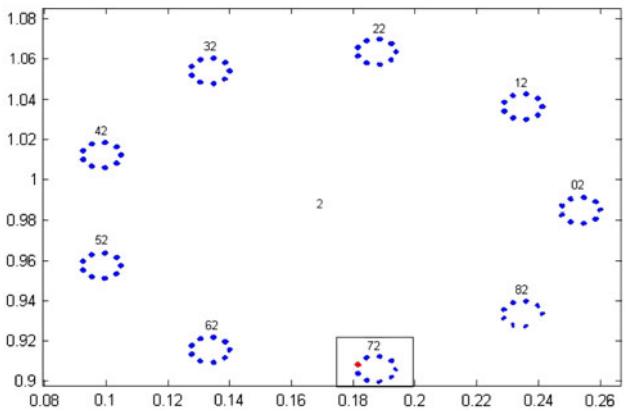

Fig. 9 IFS graph, zooming into circle address 2. An area of relatively high density is detected in circle address 72

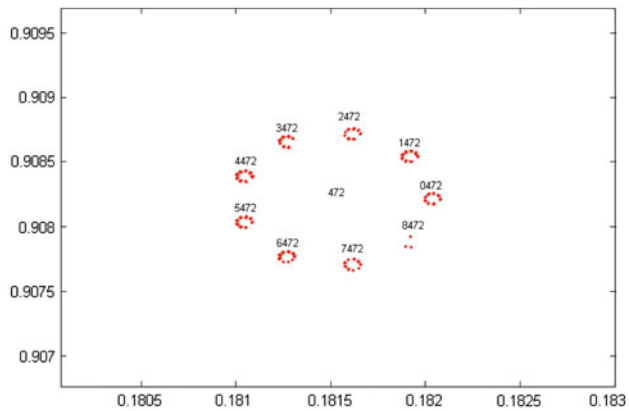

Fig. 11 IFS graph, zooming into circle address 472. Points seem to be almost uniformly distributed between circles of addresses 0472-7472. No area of relatively high density is detected

Experimental results

We present a summary of the faulty operational sequence patterns detected by the suggested algorithm, compared to those detected by the bi-gram approach suggested by Da Cunha et al. (2006).

Following are the detected faulty operational sequence patterns causing product defects:

1. Operational sequence pattern 4-7-2, in cases where it is not performed at the beginning of a manufacturing process, is the cause of faulty products requiring rework of operation 4.
2. Operational sequence pattern 5-0-4, in cases where it is performed at the beginning of a manufacturing process (i.e. the full sequence pattern is 8-5-0-4), is the cause of faulty products requiring rework of operation 4.
3. Operational sequence pattern 2-2-1, in cases where it is not performed at the beginning of a manufacturing process, is the cause of faulty products requiring rework of operation 1.

Table 2 Faulty sequence patterns requiring rework of operation 4: results summary

Method	Faulty sequence patterns detected
Bi-gram approach suggested by Da Cunha et al. (2006)	8-5
	5-0
	0-4
	4-7
	7-2
Fractal visualization method	4-7-2 rarely occurs after 8 (i.e. rarely occurs at the beginning of a manufacturing process)
	8-5-0-4 (5-0-4 mostly occurs at the beginning of a manufacturing process)
	Operation 4 appears on both patterns, but these are 2 different occurrences of it.

Table 3 Faulty sequence patterns requiring rework of operation 1: results summary

Method	Faulty sequence patterns detected
Bi-gram approach suggested by Da Cunha et al. (2006)	2-1
	2-2
Fractal visualization method	2-2-1 rarely occurs after 8 (i.e. rarely occurs at the beginning of a manufacturing process)

Tables 2 and 3 compare the results of the faulty operational sequences detected by the suggested fractal visualization algorithm to those detected by the approach suggested by Da Cunha et al. (2006).

Conclusions and further research

The assemble-to-order strategy is aimed at reducing product lead-time while offering a large portfolio of products. Since the delivery time is short, any product rework may violate the delivery time constraint. Rework of a faulty product reduces the cost of faults when its cost is lower than the cost of the lost material and labor. Nevertheless, when the product delivery time is a contractual requirement, overdue payment is added to the cost of rework. This type of fault should be particularly avoided (Da Cunha et al. 2006).

Assemble-to-order strategy enables to perform quality tests on the stocked modules without impacting the assembly schedule. Consequently, quality tests should focus on final assembly operations (Da Cunha et al. 2006).

The fractal visual analysis approach presented in this paper uses production data to determine the sequence of final assembly operations that minimizes the risk of producing faulty products.

The proposed method has the following advantages:

1. The proposed method enables visual data analysis. The main advantages of visual data exploration techniques over automatic data mining techniques from statistics or machine learning are as follows (Keim 2002):
   A. Visual data exploration can easily deal with noisy data.
   B. Visual data exploration is intuitive and requires no understanding of mathematical or statistical algorithms or parameters.

   Therefore we provide an analytical tool that can be easily utilized by the engineering staff in a manufacturing environment.
2. The proposed method can detect faulty production sequence patterns of any length without predefining the length of the pattern. It is only limited by display resolution.
3. In case an operation activity is performed more than once in a product route, such as might occur in realistic conditions (e.g., in the semi-conductor industry), the proposed method is able to distinguish between the different occurrences of the same operation activity. For example, as demonstrated, if the following two faulty operational sequence patterns, 8-5-0-4 and 4-7-2, cause a product defect, our method detects these two different

patterns and therefore clearly distinguishes between the different occurrences of operation 4.

4. The proposed method provides significant added value by enabling visual detection of rare and missing operational sequences per product quality measure.

In the next phase we plan to apply the method to actual data from a manufacturing assemble-to-order environment, e.g. an automobile plant. This step will help in studying the robustness of this approach.

Further research should consider additional manufacturing parameters that affect product quality such as operation settings and material data.

The algorithmic framework is applicable for many additional domains, for example, visual analysis of churned customers' action history, visual analysis of product defect codes history, and more.

The application is utilized by the General Motors research labs located in Bangalore, India, for visual analysis of vehicle warranty claims data.

Appendix A

The described IFS with circle transformation must be a 'totally disconnected' IFS (Barnsley 1988). In other words, the graph resulting from circle transformation must consist of m disconnected circles on all resolutions. **Theorem** *In order to fulfill this requirement, the relation between* α *and number of categories* m *needs to fulfill*:

$$\frac{\alpha}{1-\alpha} < \sin\left(\frac{\pi}{m}\right)$$

Proof The graph resulting from IFS of circle transformation is in the form of m circles, with each circle at each resolution comprising m circles.

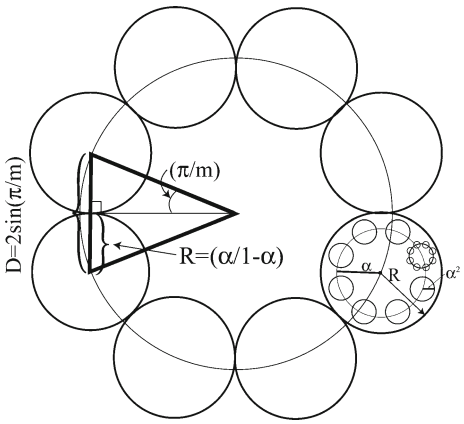

Fig. 12 Tangent circles of radius R on 1st resolution

We assume a case of circles of the 1st resolution being tangent to each other (See Fig. 12).

Due to contraction mapping by α, the radius of the m circles at the 1st resolution is α, and contracts at the factor of α respectively for every resolution. In other words, the radius of the circle in resolution j is given by α^j.

Note that the radius of the primary circle on the 1st resolution equals 1. (The primary circle is the main circle which comprises all centers of m circles of the 1st resolution. See Fig. 12).

In order for the IFS to be 'totally disconnected', the distance between two centers of two neighboring circles on the 1st resolution must be larger than twice the sum of *the radii of all circles in all resolutions.*

The sum of the radii of all circles in all resolutions is denoted as R and is given by:

$$R = \sum_{k=1}^{\infty} \alpha^k = \frac{\alpha}{1-\alpha}$$

The distance between two centers of two neighboring circles on the 1st resolution is denoted as D, and is given by: $D = 2sin(\pi/m)$, see Fig. 12.

Therefore, in order for the circles to be disconnected, they should obey:

$$2\mathrm{R} < \mathrm{D} \quad \Leftrightarrow \quad \frac{\alpha}{1-\alpha} < \sin\left(\frac{\pi}{m}\right) \qquad \square$$

Appendix B

Full demonstration of sequence pattern visual detection process—continuation of Section "Faulty production sequence pattern detection".

See Tables 4, 5 and 6.

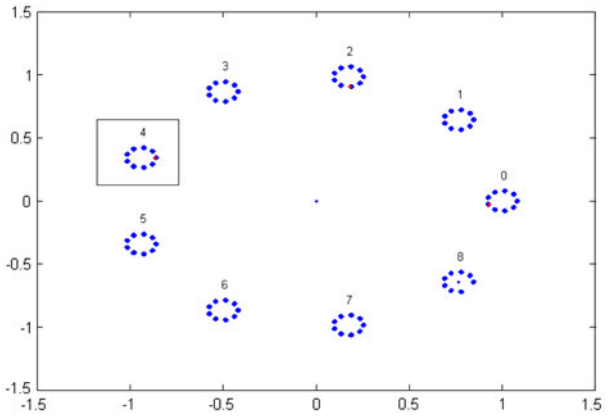

Fig. 13 IFS graph, 1st resolution. One relatively high density area is detected in circle address 4

Table 4 Exploring a second sequence pattern from IFS graph of dataset 1

Process step	Action	Result	Conclusion	Corresponding figure
1	Explore 1st resolution of dataset 1 IFS graph	One relatively high density area is detected in circle address 4.	Partial sequence pattern detected: 4	Fig. 13
2	Zoom into circle address 4	An area of relatively high density is detected in circle address 04.	Partial sequence pattern detected: 0-4	Fig. 14
3	Zoom into circle address 04	An area of relatively high density is detected in circle address 504.	Partial sequence pattern detected: 5-0-4	Fig. 15
4	Zoom into circle address 504	An area of relatively high density is detected in circle address 8504.	Partial sequence pattern detected: 8-5-0-4	Fig. 16
5	Zoom into circle address 8504	No area of relatively high density is detected	Full sequence pattern detected: 8-5-0-4	Fig. 17

Interpretation: The second sequence pattern discovered for dataset 1 is 8-5-0-4. The second sequence pattern consists of three operations and the delimiter 8. Hence 5-0-4 is the operational sequence pattern, while token 8 indicates that operational sequence pattern 5-0-4 is mostly located at the beginning of the manufacturing process. We therefore conclude that operational sequence pattern 5-0-4 may be the cause for product defects requiring rework of operation 4, in cases where operation 5 is the first assembly operation, operation 0 is the second operation, and operation 4 is the third operation in the manufacturing process

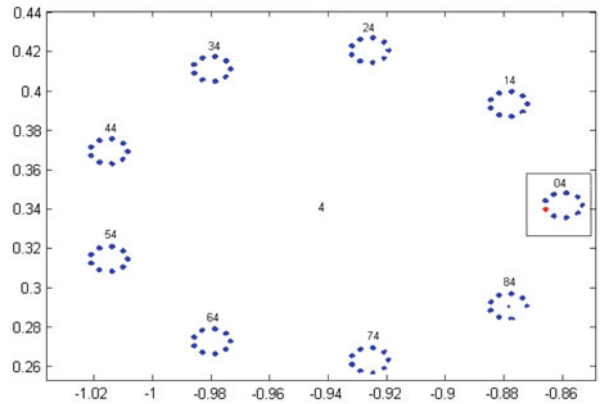

Fig. 14 IFS graph, zooming into circle address 4. An area of relatively high density is detected in circle address 04

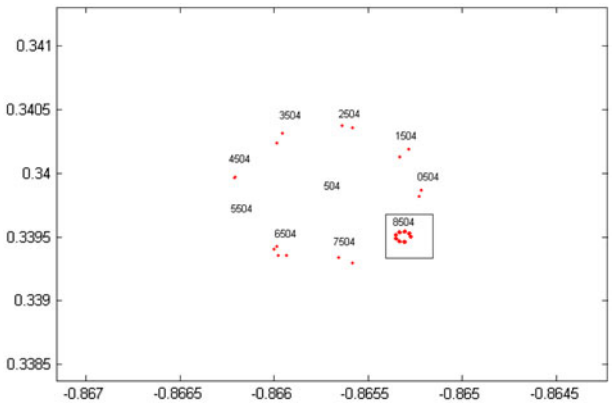

Fig. 16 IFS graph, zooming into circle address 504. An area of relatively high density is detected in circle address 8504

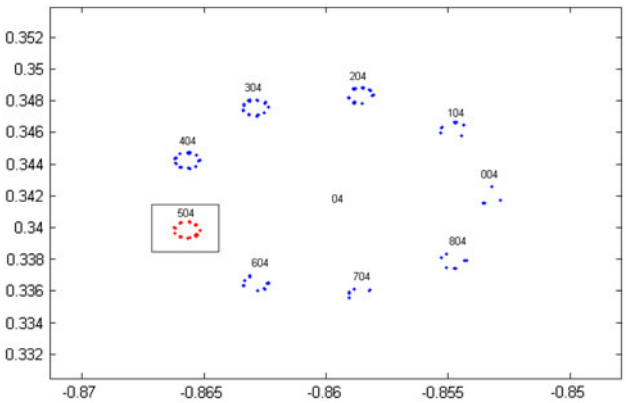

Fig. 15 IFS graph, zooming into circle address 04. An area of relatively high density is detected in circle address 504

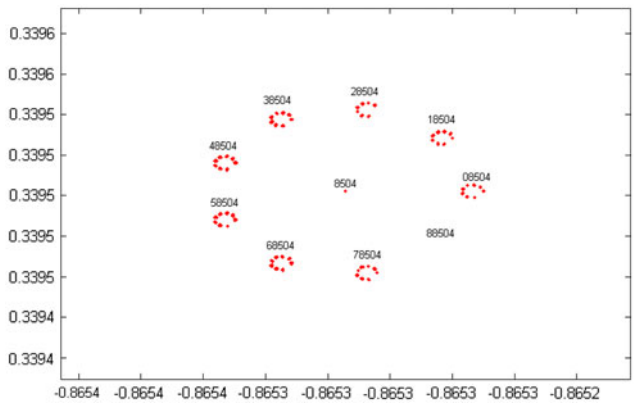

Fig. 17 IFS graph, zooming into circle address 8504. Points seem to be almost uniformly distributed between circles 08504-78504. No area of relatively high density is detected

Table 5 Exploring a sequence pattern from IFS graph of dataset 2

Process step	Action	Result	Conclusion	Corresponding figure
1	Explore 1st resolution of dataset 2 IFS graph	An area of relatively high density is detected in circle address 1.	Partial sequence pattern detected: 1	Fig. 18
2	Zoom into circle address 1	An area of relatively high density is detected in circle address 21.	Partial sequence pattern detected: 2-1	Fig. 19
3	Zoom into circle address 21	An area of relatively high density is detected in circle address 221.	Partial sequence pattern detected: 2-2-1	Fig. 20
4	Zoom into circle address 221	1. No area of relatively high density is detected	1. Full sequence pattern detected: 2-2-1	Fig. 21
		2. Circle address 8221 is nearly empty	2. Sequence 8-2-2-1 is rare	

Interpretation: The detected operational sequence pattern for dataset 2 is therefore 2-2-1. Note that circle address 8221 is nearly empty, i.e. sequence 8-2-2-1 is rare. Since token 8 represents the delimiter, it represents the beginning of a manufacturing process in this case. We therefore conclude operational sequence pattern 2-2-1 may be the cause for product defects requiring rework of operation 1 in cases where the sequence pattern is not performed at the beginning of a manufacturing process

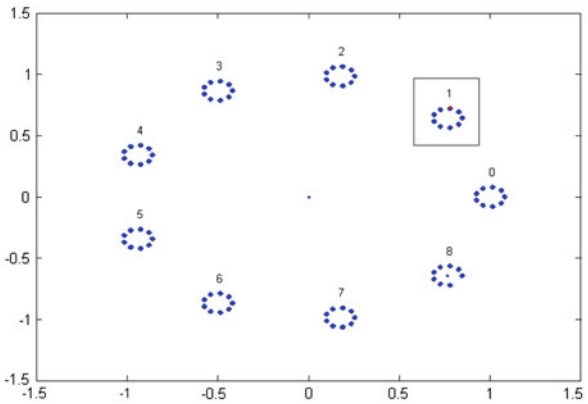

Fig. 18 IFS graph, 1st resolution. An area of relatively high density is detected in circle address 1

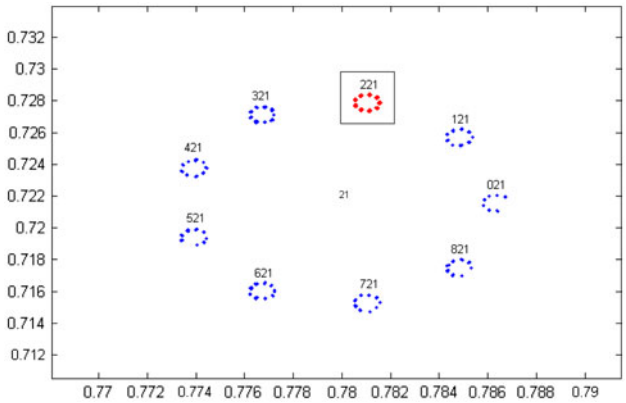

Fig. 20 IFS graph, zooming into circle address 21. An area of relatively high density is detected in circle address 221

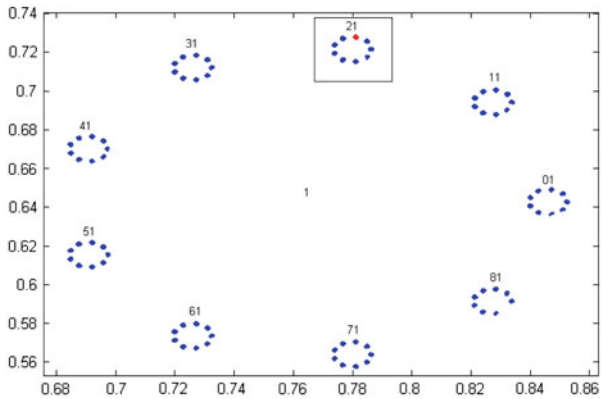

Fig. 19 IFS graph, zooming into circle address 1. An area of relatively high density is detected in circle address 21

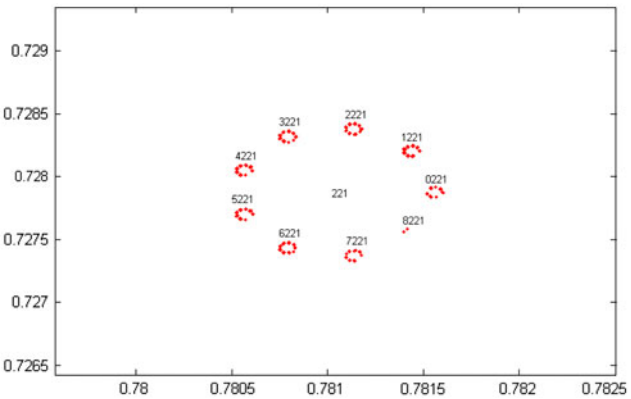

Fig. 21 IFS graph, zooming into circle address 221. Points seem to be almost uniformly distributed between circles of addresses 0221-7221. No area of relatively high density is detected

Table 6 Exploring sequence patterns from IFS graph of dataset 3

Process step	Action	Result	Conclusion	Corresponding figure
1	Explore 1st resolution of dataset 3 IFS graph	No area of relatively high density is detected	No sequence patterns exist in the dataset	Fig. 22

Interpretation: Dataset 3 represents all manufacturing processes that resulted in products that passed quality control. No production sequence pattern was detected for these processes

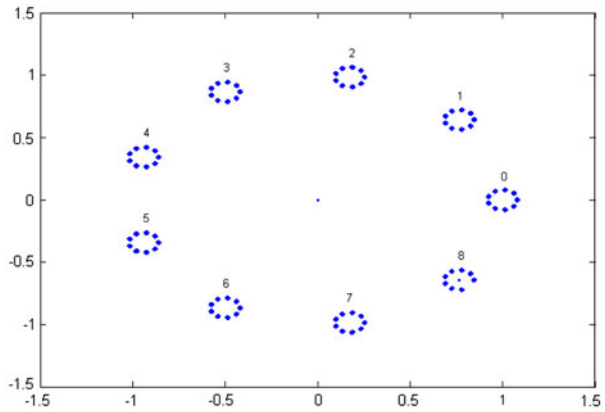

Fig. 22 IFS graph, 1st resolution. No area of relatively high density is detected

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the international conference on large databases*, pp. 478–499.

Barnsley, M. (1988). *Fractals everywhere*. Boston: Academic Press.

Barnsley, M., & Hurd, L. P. (1993). *Fractal image compression*. Boston: A. K. Peters.

Ben-Gal, I., Shmilovici, A., & Morag, G. (2003). Context-Based statistical process control: A monitoring procedure for state-dependant processes. *Technomatrix, 45*, 293–311.

Cavner, W. B., & Trenkle, J. M. (1994). *n*-gram based text categorization. In *Proceedings of the third annual symposium on document analysis and information retrieval*, pp. 261–169.

Da Cunha, C., Agard, B., & Kusiak, A. (2006). Data mining for improvement of product quality. *International Journal of Production Research, 44*(18–19), 4027–4041.

Hand, D. (1998). Data Mining—Reaching beyond statistics. *Research in Official Statistics, 1*(2), 5–17.

Jeffrey, H. J. (1990). Chaos game representation of genetic sequences. *Nucleic Acids Research, 18*, 2163–2170.

Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Transactions of Visualization and Computer Graphics, 7*(1), 100–107.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann.

Rokach, L., & Maimon, O. (2006). Data mining for improving the quality of manufacturing: A feature set decomposition approach. *Journal of Intelligent Manufacturing, 17*(3), 285–299.

Rokach, L. (2008). Mining manufacturing data using genetic algorithm based feature set decomposition. *IJISTA, 4*(1), 57–78.

Rokach, L., Romano, R. & Maimon, O. (2008). Mining manufacturing databases to discover the effect of operational sequence on the product quality. *Journal of Intelligent Manufacturing.*

Weiss, C. H. (2008). Visual analysis of categorical time series. *Statistical Methodology, 5*, 56–71.