International Workshop on Visual Analytics (2012)
K. Matkovic and G. Santucci (Editors)

PCDC - On the Highway to Data
A Tool for the Fast Generation of Large Synthetic Data Sets

S. Bremm[1] and M. Heß[1] and T. von Landesberger[1] and D. W. Fellner[1,2]

[1]Technische Universität Darmstadt, Germany
[2]Fraunhofer IGD, Darmstadt, Germany

Abstract

In this paper, we present Parallel Coordinates for Data Creation (PCDC), a new visual-interactive method for the fast generation of labeled multidimensional data sets. Multivariate data need to be analyzed in various domains such as finance, biology or medicine using complex data mining techniques. For the evaluation or presentation of the techniques, e.g., for assessing their sensitivity to specific data properties, test data need to be generated. PCDC allows for a fast and intuitive creation of multivariate data with several classes. It is based on interactive definition of data regions and data distributions in a parallel coordinates view. It offers a quick definition of data regions over several dimensions in one interface. Moreover, the users can directly see the outcome of their settings in the same view without the need for switching between data generation and output visualization. Our tool enables also an easy adjustment of the data generation parameters for creating additional similar datasets.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.3]: Viewing Algorithms—Computer Graphics [I.3.6]: Methodology and Techniques—Graphics data structures and data types

1. Introduction

Multivariate data need to be examined in various domains such as finance, biology or medicine. The data are often analyzed using modern data mining algorithms. Main tasks include finding clusters of similar data objects (e.g., grouping patients according to their blood tests), finding data properties for differentiating various types of data objects (e.g., differentiating healthy from ill tissues), or generating lower dimensional representations (e.g., visualization of multidimensional measurements).

Creating robust and high quality algorithms for data analysis requires their controlled evaluation, which is commonly done on a basis of test data sets. This data basis should cover a wide variability in data properties. Moreover, the data repository should include data sets with predefined properties that need to be found by the algorithms. However, often sufficiently large and variable real test data are not available. This is because real data are difficult and time-costly to obtain or they are not available due to privacy concerns (esp. in medical applications). Additionally, real datasets may not include data with specific properties needed for testing the robustness of the algorithms. Therefore, researchers often rely on synthetic data sets [PHL04].

Synthetic data sets are created by data generation tools, which are usually provided by statistics or software testing environments. They are specific solutions with limited options. A user-friendly way of creating data sets is by visual interfaces [ALM11], which allow for drawing of data regions and data properties. However, creating multidimensional data sets with several classes (also denoted as labels, or clusters) is difficult and time consuming, as the users need to define data properties for many dimensions.

In this paper, we present a new approach for the fast creation of multivariate floating-point data with several classes. The user can interactively define data ranges, data distributions and sampling rates for all or only some specified dimensions. We employ a parallel coordinates view as it provides fast and intuitive way to define and represent the model in several dimensions simultaneously. An advantage of our system is the direct visualization of the data outcome in the same view. For testing the sensitivity of algorithms, our tool also provides easy adjustment of data generation parameters

DOI: 10.2312/PE/EuroVAST/EuroVA12/007-011

as well as their export and import.
PCDC is not only well suited for testing of various data mining algorithms, but can also be employed to teach their strengths and weaknesses to students.

This paper is organized as follows. Section 2 gives an overview of related work in this area. Section 3 presents details of our approach. Section 4 shows an example use case and Section 5 concludes and outlines future work.

2. Related Work

Test data sets are needed for evaluating data mining algorithms [PHL04] or software systems. Benchmark datasets from public repositories such as UCI [FA10] or WEKA datasets [HFH*09] are popular in this respect. These data sets are widely used for comparing speed or quality of algorithms. However, they may not include data sets with specific properties needed for testing new types of algorithms or for testing algorithm sensitivity to special data characteristics. In these cases, synthetic data sets are generated by domain experts. They usually develop one-off programs specifically for the particular problem at hand [PHL04].

Synthetic data can be also created using statistical software tools or programming libraries, that include data generation methods from various distributions. For example, R tool [Tea], MATLAB [Mat] or Apache Commons Math library in Java [Fou] provide this functionality. These tools, however, require a sound programming knowledge and expertise in data generation methodology [How75, Edv99, PSVS05]. Moreover, the experts need to specify data properties "blindly", i.e., without the possibility to directly see the impact of program settings on the data output.

A user-friendly way of producing synthetic data sets is provided by the GenerateData Tool [Kee]. It offers a simple interface, where the user defines each dimension of the dataset separately. It allows for producing datasets with various types of data (numbers, address, names, postal codes, etc.). However, defining each dimension individually using drop-down boxes is very time consuming. Moreover, the interface does not include any visual feedback on the outcome.

Recently, a visual-interactive way of defining data generation properties was presented by Albuquerque et al. [ALM11]. It provides a user interface for drawing data properties and a set of data distributions for data generation. Although it offers the possibility to create multidimensional datasets, the data definition process is constrained to 2D and 3D interfaces (Fig. 1 left). Therefore, the user has to define each dimension combination separately, which is time consuming (quadratic time with respect to the number of dimensions) and does not allow the easy definition of relationships among several dimensions. Moreover, the data creation view differs significantly from the output view. So the user needs to make a mental correspondence between data parameters and outcome, which is often difficult (Fig. 1). Moreover, adjustments of previously created datasets in the visual interface are not possible. The user has to start a new data creation process. Therefore, we propose a new approach which overcomes these limitations.

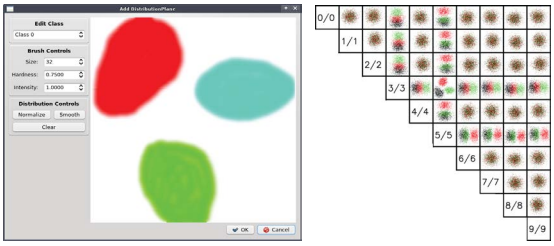

Figure 1: *Data generator by Albuquerque et al. [ALM11]. It provides only 2D and 3D interfaces for defining data properties (left). The output is presented in a different view (right). Creating multidimensional data is cumbersome and unintuitive.*

3. Approach

We introduce a visual user interface, where the users can interactively define data properties for generating synthetic multivariate data sets (Fig. 2). It additionally directly displays the outcome, which enhances the intuitiveness of the data generation. The view is composed of two parts:
1. The main part (on the right) employs a parallel coordinates plot for drawing and displaying the data properties. Each data dimension is shown as a vertical axis, data properties (value ranges) are displayed as colored bars on the axes. Colors encode data classes. Value ranges on the axes are connected with transparent bands.
2. The left part displays an overview of the data classes and provides the possibility to define data distributions.

For data generation, the user interactively defines data dimensions, classes, value ranges and type of data distribution. This process can be seen in the video [BHvLF].
Data dimensions and classes are interactively added or removed on demand as all settings (such as axis ordering) can be changed later in the process.
Value ranges in data dimensions are initialized by clicking on an axis and can be moved or resized. As one data class may consist of several value ranges, we offer the possibility to "split" value ranges (Fig. 2 green class). For each range, the number of samples and their distribution can be defined separately.

For fast value range definition on several axes, the user can move the mouse over the axes where new value ranges are created. Moreover, the user does not need to define value ranges for all classes in all dimensions. If no range for a class in a dimension is specified, its data samples are distributed either throughout the whole range or within the free

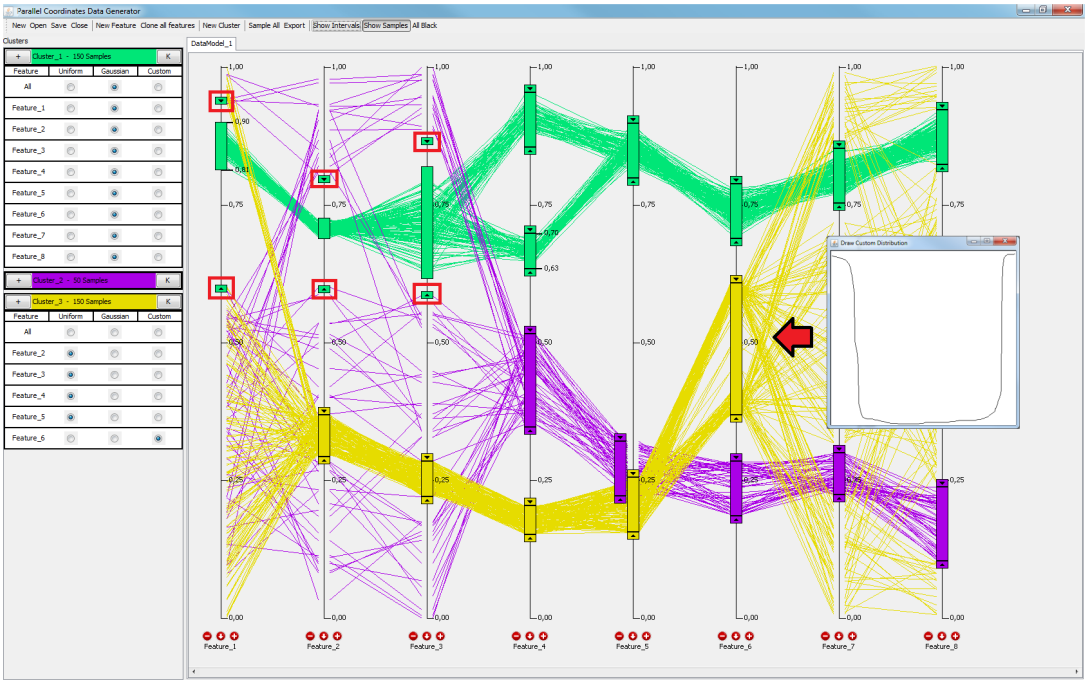

Figure 3: *The sampled data according to the model defined in Fig. 2. The different distributions are easy to spot (green: Gaussian, purple: uniform, yellow: uniform with a custom drawn distribution in the 8th dimension)*

Figure 2: *Data generation interface. Left: Overview of data classes and dimensions including data distribution options. Right: Interactive definition of data dimensions and data ranges in parallel coordinates view. Classes are color coded. Value ranges are shown as bars in the respective axis.*

areas so they do not overlap with other classes. Moreover, we provide the possibility to define "prohibited" value ranges, where no values will be created (Fig. 3, highlighted with red boxes). This can e.g. be used to simplify a problem for a cluster algorithm. Additionally, single or all dimensions at once can be duplicated including all class definitions and already sampled data. By this, a higher dimensional image of a pattern can be created quickly to evaluate the dimensional dependency of a target algorithm as nearest neighbor information become less meaningful in high dimensional spaces [BGRS99].

Data distribution properties need to be defined for each value range. For a faster definition, they can be defined at once for whole classes (Fig. 3). The user can choose from a given, but easily extendable set of distributions. We currently provide uniform, Gaussian [Ora] or custom user-drawn [Bis06] distributions.

After the user defined all data properties, data is generated by drawing from the user-defined data distributions in each data range. The final result can be immediately seen by showing data values as lines in parallel coordinates view (Fig. 3). Data values can be exported as various ASCII file formats (e.g., CSV or ARFF).

An important feature of data generators is their reusability. We therefore provide export and import of data generator settings. They allow for creating new datasets with the same or similar properties. Data properties can be easily changed by interacting with the data view.

4. Example Use Case

In this section, we show how our system can be employed for educational purposes in Visual Analytics lectures. We base the use case on our own teaching experience, when we explain the students strengths and weaknesses of selected data mining and transformation algorithms, such as principal component analysis (PCA). Having a striking example at hand makes this task much easier. However, finding good examples or creating them manually has always been very time consuming for us.

The first example shows the strength of PCA. During the lecture, we can quickly draw a 13-dimensional dataset with 14 classes and 150 samples each. Showing only the result in parallel coordinates makes it very hard for the students to identify the number of classes in the visualization (Fig. 4 top). After projecting the samples to two dimensions using PCA in WEKA [HFH*09], all clusters can easily be discriminated (Fig. 4 bottom).

The second example demonstrates the weakness of the PCA in contrast to the capability of human pattern recognition. We again create a 13-dimensional dataset. However, this time we use only two classes which are clustered in one dimension and uniformly distributed in the other 12 dimensions (Fig. 5 top). The students can easily spot this pattern in the parallel coordinates but not in the PCA projection (Fig. 5 bottom).

The education message following from these two synthetic examples is: Depending on the use case, both data mining and visualization have their strengths, so a combination of both is desirable for proper data analysis.

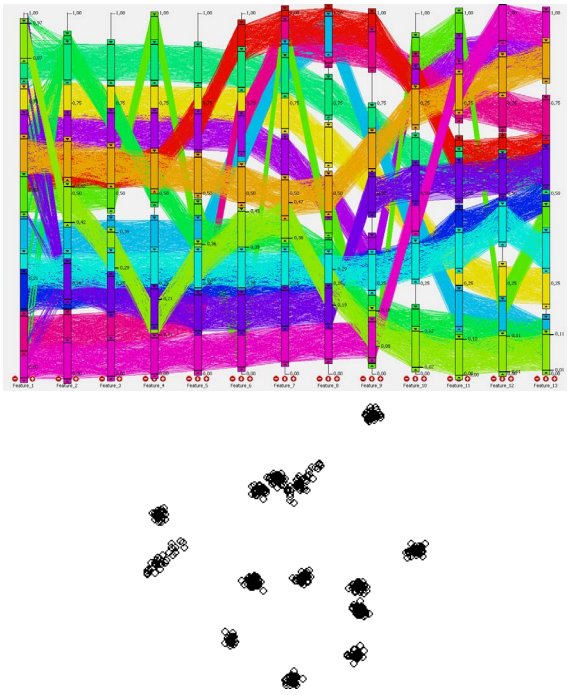

Figure 4: *Example showing strength of PCA. Top: The created labeled dataset. Bottom: Unlabeled data set. Bottom: A very good PCA result.*

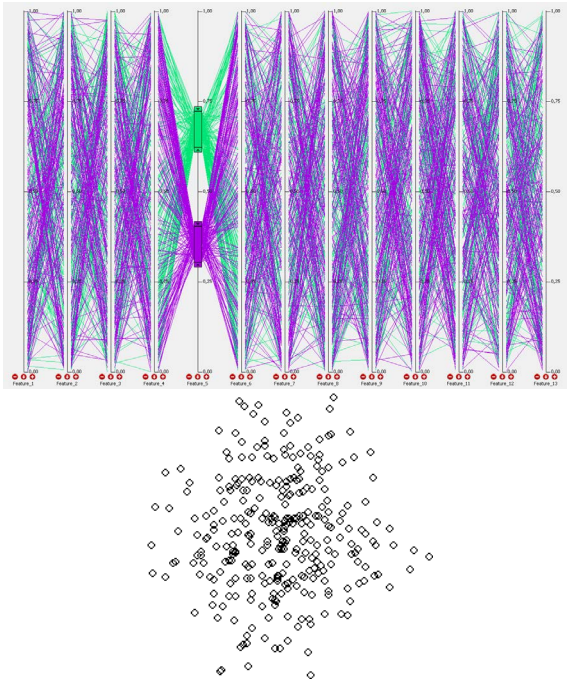

Figure 5: *Example showing weakness of PCA. Top: The created dataset. Bottom: A PCA result with bad clustering.*

5. Conclusion and Future Work

We have presented a novel way of generating synthetic multidimensional floating-point data sets using a visual interactive interface based on parallel coordinates. The users can intuitively and quickly define labeled data sets with relationships in several dimensions. They can inspect the created data immediately in the same single view. The system also allows for adjustment of previous data generation parameters needed, e.g., for testing sensitivity of algorithms or revising generated data properties.

As this tool is work in progress, many more features are planed like user defined normalization, different types of values (e.g. categorical data) or a visual feedback of the used distributions. Right now, system is suitable for data up to dozens of dimensions. We want to extended it for larger data sets (w.r.t., number of dimensions and number of classes). We have introduced a concept for splitting of classes, e.g., for subspace clustering. It however does not yet allow for control of data generation at object level (which object belongs to which split).

References

[ALM11] ALBUQUERQUE G., LÖWE T., MAGNOR M.: Synthetic generation of high-dimensional datasets. *IEEE Transactions on Visualization and Computer Graphics (TVCG, Proc. Visualization / InfoVis) 17*, 12 (Dec. 2011), 2317–2324. doi: http://dx.doi.org/10.1109/TVCG.2011.237. 1, 2

[BGRS99] BEYER K., GOLDSTEIN J., RAMAKRISHNAN R., SHAFT U.: When is nearest neighbor meaningful? *Database Theory ICDT 99* (1999), 217–235. 3

[BHvLF] BREMM S., HESSM., VON LANDESBERGER T., FELLNER D.: PCDC - On the Highway to Data, A Tool for the Fast Generation of Large Synthetic Data Sets. http://www.gris.tu-darmstadt.de/research/vissearch/projects/pcdc/. video & sourcecode. 2

[Bis06] BISHOP C.: *Pattern recognition and machine learning*, vol. 4. springer New York, 2006. 3

[Edv99] EDVARDSSON J.: A survey on automatic test data generation. In *Proceedings of the 2nd Conference on Computer Science and Engineering* (1999), pp. 21–28. 2

[FA10] FRANK A., ASUNCION A.: UCI machine learning repository, 2010. 2

[Fou] FOUNDATION A. S.: Apache Commons Math library. http://commons.apache.org/math/. Online; accessed April 2012. 2

[HFH*09] HALL M., FRANK E., HOLMES G., PFAHRINGER B., REUTEMANN P., WITTEN I. H.: The WEKA data mining software: an update. *SIGKDD Explor. Newsl. 11* (November 2009), 10–18. 2, 4

[How75] HOWDEN W.: Methodology for the generation of program test data. *Computers, IEEE Transactions on C-24*, 5 (may 1975), 554 – 560. 2

[Kee] KEEN B.: GenerateData. http://www.generatedata.com/. Online; accessed April 2012. 2

[Mat] MATHWORKS: Matlab. http://www.mathworks.de/products/matlab/. Online; accessed April 2012. 2

[Ora] ORACLE: Java™ platform standard ed. 7 API. http://docs.oracle.com/javase/7/docs/api/. Online; accessed April 2012. 3

[PHL04] PARSONS L., HAQUE E., LIU H.: Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl. 6* (June 2004), 90–105. 1, 2

[PSVS05] PRASANNA M., SIVANANDAM S., VENKATESAN R., SUNDARRAJAN R.: A survey on automatic test case generation. *Academic Open Internet Journal 15* (2005), 1–5. 2

[Tea] TEAM R. D. C.: The R Project. http://www.r-project.org/. Online; accessed April 2012. 2