SIGRAD 2014
M. Obaid, D. Sjölie, E. Sintorn and M. Fjeld (Editors)

IPFViewer: Incremental, approximate analysis of steel samples

M. Thurau[1] & C. Buck[1] & W. Luther[1]

[1]University of Duisburg-Essen, Germany

Abstract

This paper describes the design of a visual analysis tool for our Inclusion Processing Framework, called IPFViewer. The tool has been designed in cooperation with a large German steel production facility in order to acquire knowledge from data collected about nonmetallic inclusions and other defects in steel samples. We have highlighted parts of the framework in previous publications in interdisciplinary journals. Here, we describe our contribution in the area of grouped or clustered data items. These data groups are visualized by techniques known from uncertainty visualization to make visible fluctuations and corresponding variations in steel samples. However, our results are also transferable to other ensemble data. To find an optimal way to design algorithms and visualization methods to process the huge data set, we discuss the project-specific requirements regarding memory usage, execution behavior and precision. By utilizing approximate, incremental analysis techniques when needed, the responsiveness of the application is ensured while high precision is guaranteed for queries with fast response times. The design allows workers at the steel production facility to analyze correlations and trends in billions of data rows very quickly and to detect outliers in routine quality control.

Categories and Subject Descriptors (according to ACM CCS): Computing Methodologies [I.3.6]: Computer Graphics—Methodology and Techniques

1. Introduction

In [HBT*12] we presented a new approach to measuring nonmetallic inclusions and other defects in steel samples at a real-world steel production facility. These samples are continuously sliced by a milling machine to accumulate volumetric data about any defects they might contain through repeated image capturing and processing. For each defect various attributes, including volumetric statistics, such as volume and sphericity, are calculated and stored. In [BBPL14] we continued our work by classifying the defects with the help of machine learning algorithms. As a result, three types of defects can be identified: pores, such as those containing trapped gases; solid inclusions containing non-metallic elements, like aluminum oxide; or longish cracks. In [TBL14] we reported on some features and the visual capabilities of the visual analysis tool IPFViewer, which was built explicitly for visually analyzing the huge data set. We also responded to objectives and tasks defined by the steel experts in greater detail and presented applicable visualizations and interaction techniques. In this paper, we report on ongoing research in the area of data groups — in particular, groups of samples. Samples using the same production parameters are of special interest. Through natural fluctuations during the steel-making process, each sample is different. These differences lead to uncertainties in all kinds of measurements, including the amount, size and position of defects in the sample. Uncertainties are visualized with the help of uncertainty visualization techniques to allow the analysis of variances. These large data groups, which contain billions of data items, yield performance problems that we have not previously discussed in detail.

Visual analytics of the data set provides highly valuable information. Steel experts want to know what influence various process parameters have on the outcome of the steel product (sensitivity analysis). Their interest lies in a better understanding of the steel making process, which has been a significant focus of research for over 80 years. Defects have a major impact on the mechanical properties of steel and therefore on its steel. The goal is to improve the quality of steel while, at the same time, lowering the costs of producing

it. Additionally, quality control in daily usage is of interest. By comparing the measured production outcomes with reference data that show the intended outcomes, manufacturers can monitor production and detect outliers.

Our scientific contribution is a design study that solves a specific real-world problem in the domain of steel production. The targeted audience is steel experts who want to analyze their steel quality. First, we define a new task which was not discussed in our previous papers: the grouping of multiple objects, mostly the grouping of samples of similar production parameters. After specifying the design requirements from a visual analytics perspective, we justify our general design choices based on related work in the literature. The design itself is then presented along with a use case. These are analyzed in order to validate the design choices and point out possible improvements. In short, the design choices are

- uncertainty visualization to show fluctuations.
- data management based on expected query cost.
- incremental, approximate analysis for large queries.

2. Task Analysis and Design Requirements

We begin by defining the tasks necessary for an analysis of milled steel samples based on data collected by a large German steel facility. Each sample is an ensemble member in an ensemble data set [WP09]. There are multiple steel samples containing multiple defects. Therefore, the data set is a tree of ensemble members, or hierarchical ensemble data. The data is also large, multidimensional and multimodal. In [TBL14], we presented a new way of dealing with that data, which came down to three tasks:

- **Task #1: Single node analysis**
  Analyzing a single node (a subtree) in the data tree (e.g., a sample and its defects) with the help of a multiple view system to generate hypotheses and to find relationships and anomalies.
- **Task #2: Repeatability analysis**
  Knowledge from task #1 is reexamined in neighboring objects by utilizing *small multiples of multiple views*.
- **Task #3: Trend analysis**
  How does a correlation between two or more attributes vary with respect to other attributes? For example, one hypothesis is that, the larger a defect, the more spherical it will be. Such correlations may only be true for some of the smoother steels. This is shown by sorting the small multiples.

Next, we define a new task, which is the aggregation or grouping of multiple samples and defects:

- **Task #4: Aggregation of objects**
  Repeat task #1 to #3 with user-defined groups of probations or defects.

The outcomes of the steel making process have certain natural fluctuations even when the same production parameters have been used. Therefore, analysis of a single sample may lead to misconceptions if that sample is not representative or is an outlier. By grouping multiple samples, there is a high probability that the resulting statistics will yield a general truth that is less susceptible to outliers and will predict the features of future production. This prediction depends on breadth of the range of fluctuations. These variances may depend on some process parameters and are important factors. This kind of analysis is known as uncertainty or variance analysis. Steel experts want to know the boundaries of the range of possible outcomes. They want to be able to identify best- and worst-case samples as well as the range of most probable outcomes.

Aggregating samples with similar production parameters helps in the analysis of the range of possible outcomes, while aggregating samples with similar outcomes helps in the analysis of the range of process parameters suitable to achieve particular outcomes. For example:

- **To guarantee similar production conditions, the following parameters are important:** steel type, temperatures, amount of oxygen and timings.
- **Similar outcomes are grouped based on:** amount and types of defects and total cleanliness.

After identifying the tasks for the steel analysis, we imposed additional requirements on the system in order to cope with the challenges familiar in visual analytics:

Requirement #1: Visualization of uncertainty. As shown in previous publications, each sample is visualized with the help of different statistical visualizations. For instance, a trend graph showing the influence of the defect diameter on the defect eccentricity in a particular sample. Larger defects may be more spherical. When dealing with a whole group of samples, we get a family of curves (one curve for each sample) [KLM*12]. These multiple curves can be aggregated and then visualized as a single trend chart with uncertainty bars or box plots.

Requirement #2: Flexible parameter set. The concrete set of data attributes for samples and defects may change from time to time. There are thousands of process parameters and measurements stored for each sample and defect on different independent database systems. For security reasons, however, the database for the analysis will be separated and process parameters will be added and removed regularly based on current analysis goals.

Requirement #3: Fast response times. There are expected to be more than 10 billion defects over the next 10 years. Statistics over such huge amounts of data items cannot be calculated quickly enough to allow interactive analysis with continuously changing user demands. Scalability to huge datasizes is one of the current challenges in visual analytics in the research agenda [TC06].

3. Related Work

The visual analysis of ensemble data sets is relatively new [WP09]. Wilson et al. published a general overview of challenges of ensemble data sets. We compared our general approach with their paper about Ensemble-Vis [PWB*09] in a previous paper [TBL14].

To enable fast querying in large databases, a great deal of research is being conducted in the area of database optimization [KIML11]. There is a great demand on standardized implementations at the database level, because optimizations at this level are beneficial for all visual analytics tools [TC06]. However, such techniques are not yet mature or available for standardized databases.

A recent publication in this area is BlinkDB [AMP*13]. Instead of querying every single data item, it uses a small sample of the database to get an approximate answer. There are a lot of sampling based solutions available. The challenge is to find an optimal sample that best approximates the precise answer. BlinkDB manages multiple sets of uniformed and stratified samples for that purpose. The precalculated sample selection is based on predictions of the most likely future queries. In their case, they predict the columns involved in the query. Their work is very impressive, as they developed an approximate query engine with both error and response time constraints. The disadvantage of systems based on prediction and pre-calculation is a certain degree of inflexibility when compared to incremental, approximate analysis. Some answers can be given very quickly, while others cannot. This is a problem in explorative analysis, where all answers should be provided interactively. In our case, the flexible parameter set is an additional burden to pre-aggregation. In BlinkDB's taxonomy of four different workload models, the most flexible one is the online aggregation model. On the negative side, online aggregation does not offer solutions to increase efficiency. It is only supposed to return intermediate results so that users are involved in the calculation process. They can monitor interim results and intervene accordingly. An optimal way to gain efficiency and flexibility at the same time is to use hybrid systems, which may arise when mature implementations of each technique are available. Systems can evaluate query costs and choose the optimal corresponding technique.

Until then, we have chosen to implement the most flexible technique, online aggregation, which was called **incremental, approximate analysis** in a recent paper in visual analytics [FPDs12]. The same technique is also used by the CONTROL project [HHW97]. Incremental, approximate analysis or "online incremental analysis" has many advantages over other techniques. The idea is to calculate answers on a small sample of the data first. The approximate answer is visualized, while another sampling of the data is obtained that does not include the data items of the previous sample (sampling without replacement). After that, the two approximate answers are merged to obtain a better approximation. At the last data iteration, all available data has been taken into account, which leads to either precisely or closely approximated results. What was missing in previous publications was a discussion of the algorithms used. While [FPDs12] only named *average*, *sum* and *count*, more sophisticated statistical functions, like *quantile estimation*, are needed for a versatile analysis tool. Below, we will discuss and categorize some requirements of statistical algorithms.

Visualizing uncertainty when dealing with summary statistics is commonly done with box plots or similar techniques [PKRJ10]. Instead of analyzing individual values, the average value can be used. Depending on the desired precision of the information, more precise summaries for data distributions are available. One of these is the five-number summary, which consists of the min and max values, the upper and lower quartiles and the median. The family of curves provides a good example. When a trend graph visualization has multiple data points per bin (one for each curve), the bin can be summarized with a corresponding summary. The trend graph shows a trend including its possible variations just as we need it for the analysis of groups of steel samples and defects (Fig. 3).

4. The Data Model

Generating groups of objects can be understood differently. When dealing with more complex algorithms that try to group objects on nontrivial aspects, literature refers to clustering. We cluster data offline only and make results available as attributes in the database. The IPFViewer groups objects based on intervals of their attributes in real time. This means, for instance, that we group the samples based on melting temperature intervals. This is also known as data binning. On the other hand, we might group samples according to categorical data dimensions, like the steel type. We allow the definition of an unbound number of dimensions for clustering. That means that users can first create clusters based on attributes such as defect diameter, for example, small and large. The resulting clusters can be further divided by defect type, such as cracks and pores. In this way, we obtain four clusters — small cracks and small pores, large cracks and large pores — which then are visualized.

4.1. Algorithm Requirements

First of all, we require **limited memory usage**. Depending on the system's memory capacities, the amount of data items and distinct data dimensions required for one or multiple visualizations, we can run into a memory overflow quite fast. As a result, we want to keep only the current iteration in memory. Data items of older iterations have to be summarized in such a way that their meaning distributes to the overall calculation without saving each item separately. An example is the calculation of the average. We can calculate and store the average of the complete first iteration without saving each individual item. In the next iteration, we can merge

the results to get a more precise average. This requirement makes quantile calculation difficult. To calculate quantiles with a naive algorithm, we need all data items to be sorted in a single list. A lot of research regarding quantile estimation is available [GK04].

The second requirement concerns **execution behavior**. Algorithms can only work batch-wise, iteratively or completely. Batch-wise means that the data is divided into parts, or batches, which are handled one after the other. When working with batch-wise algorithms, we assume the ability to merge results (Mergeable Summaries [ACH*12]). For instance, we could calculate the quantiles of several batches. However, these batch quantiles would not reveal the quantiles of the overall data. While it is possible to use the median of medians, which fulfills the batch-wise behavior, this method is not as precise as needed. On the other hand, the count function can be executed batch-wise, and simply summarizing the results of the batches results in an outstanding ability to merge them. Other examples are *sum, count, min, max* and *avg*. In contrast, iterative algorithms handle data items one after the other. Here, we also assume the single pass criterion, which means that each data item is viewed only once. If that were not the case, we would overburden the limited memory available in order to store the data items for subsequent passes. While batch-wise and iterative algorithms are appropriate for very large data sets, algorithms that require access to the complete data set without the ability to provide interim results are completely unsuitable, as discussed above.

Data handling is divided into multiple steps. At each step, execution behavior can vary. From a database we normally get multiple data items batch-wise to reduce overhead. After that, a subsequent algorithm with an independent execution behavior may run within the application. Sometimes, batch results can be calculated directly within the database system. We assume that this was the way CONTROL [HHW97] and [FPDs12] worked. When dealing with more complex algorithms, especially quantile estimation and data mining, we receive all data items batch-wise and calculate results within the application.

Another requirement deals with the **precision of results**. A statistical calculation can be precise or it can yield an estimation with precise bounds or an estimation with probabilistic bounds. While trying to achieve as much precision as possible, we work with all these precisions and make sure to tell the user the degree of precision currently available. As stated in the introduction, a high degree of precision is not necessarily needed for analyzing general trends or creating hypotheses. Related to precision is the need for a priori knowledge. Some algorithms achieve good results only for certain data distributions. Due to the nature of explorative analysis, we cannot foresee such data distributions. Other a priori knowledge often needed is the number of data items. When dealing with many data items that are highly filtered, we would have to use a previous pass to count them. Therefore, we require algorithms without any a priori knowledge.

4.2. Algorithms based on query costs

	Itemcount: multiple billion	Itemcount: multiple thousand
Standard func.: count/ max/min/ sum/avg/ std. binning (e.g. histogram)	precise Batch-wise calculation by DB, merging by application	precise One-step by DB
Statistical func.: quantiles variance skewness kutosis mode stat. binning (e.g. box plot)	approximation in some cases Batch-wise retrieval of values from DB, iterative calculation by application	precise One-step by DB

Table 1: *The data processing steps depend on table sizes and desired statistics*

We differentiate between low and high numbers of data items in the expected results. As mentioned above, the exact number of data items is not known a priori when filters are involved. However, the typical item count for tables is familiar; thus, we know the limits for some types of queries:

Query all samples: max. 100 thousand items
Query defects from one sample: max. 100 thousand items
Query defects from multiple samples: up to 10 billion items

For a small number of items, we calculate the statistics completely in one step within the database system and achieve a precise result in an acceptable amount of time (Tab. 1). For large numbers of items, we differentiate between standard functions, which are very well suited for batch-wise merging [ACH*12], and more complex functions, which are calculated by the application iteratively. For the latter, we receive all data items in batches from the database and then iteratively calculate the desired statistics using the Boost Accumulator library †. These calculations, however, are not necessarily precise, but may be approximations in some cases. For instance, the quantile estimation uses the extended P-Square algorithm [JC85]. When binning is involved, which means a statistical calculation for each

† Boost Accumulator: www.boost.org

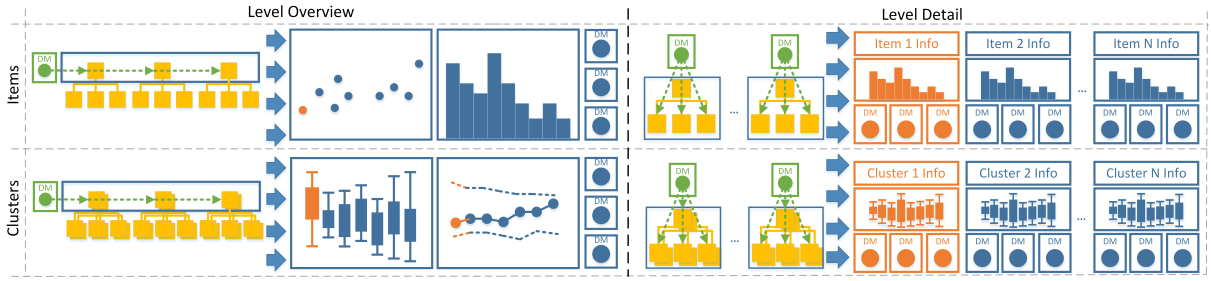

Figure 1: *The data model is divided into two parts. On the left is a level overview showing an overview of the level of interest in the data tree, which is selected by the user. Each data point in the overview corresponds to a sub-tree shown on the right. The sub-trees are visualized side by side in detail in the level detail view, using a multiple view system. The color orange indicates connections between the nodes. Grouped data is visualized with uncertainty to represent fluctuations of group members.*

bin is needed, we also differentiate between standard functions (histogram based on *count*) and statistical binning (box plots). Our implementation regarding incremental data mining is not mature yet. We mine and recommend data items based on most occurrences (*mode*) and extremes (*min* and *max* of attributes) of pre-calculated item descriptors. Once we have found suitable items, more corresponding data dimensions are received and presented as recommendations, such as the most dangerous defect.

	small table	large table
small workload per item (count, avg, etc.)	no limit	limit scanned items
large workload per item (aggregates, joins)	limit returned items	limit returned and scanned items

Table 2: *Query limit depends on table size and expected workload*

One of the problems we ran into was **paging within the database**. Paging is the operation of limiting the data to small portions, or pages, and receiving them one after the other using multiple queries utilizing LIMIT and OFFSET. As the database does not necessarily cache previous operations, a large offset means rearranging all items below the offset in the same order as before (the order must be unique), which can be a long operation. This was solved using the indexed ID as order and filter queries to the first ID that is a potential match. In this way, previous entries do not have to be scanned again. On the negative side, this prevents an optimization of the intermediate results based on sampling order, which is one of the most common optimizations. However, this problem can be solved by creating additional indices. Our solution is shown in Tab. 2. Using these limitations, the workload is distributed relatively equally among the queries, independent of the number of items that match the filters.

5. The Visualization Model

We now present the visualization model, which is shown in Fig. 1. First, a level of the main data tree is chosen as the level of interest. For instance, one might want to analyze the samples by conducting a comparison and a trend analysis. Another level of interest would be the hierarchical level containing the defects. We present all the data items on that level of the data tree in a visualization, the *level overview*, and also visualize each data item in detail side by side in the *level detail*. Each data item or cluster in the level overview is represented by a single visual item within a visualization, like a point or a bar. Multiple visualizations can be used as the level overview. In the level detail, each data item or cluster is represented by a comprehensive multiple-view system [WBWK00], showing various data dimensions, modalities and visualization forms simultaneously. Users can specify the data and visualization forms, like histograms, bar charts, graphs, scatter plots, pie charts and textual information. Since each visual item in the level overview relates to one comprehensive visualization layout in the level detail (Fig. 1, orange), sorting, filtering and clustering the data level affect both visualization methods. This is very effective for trend analysis [TBL14]. To sum up, we provide a small multiples visualization [Tuf90] (actually small multiples of multiple views) with an additional overview, as shown in Fig. 2.

The upper part of Fig. 1 deals with non-clustered data. We focus on the lower part regarding visualization of clusters. A cluster is a group of nodes on the data tree. The **precision of the summary statistic** can be selected. Instead of showing a single value to represent a large cluster, such as an average, multiple values representing boundaries have advantages when examining uncertainty [MTL78]. Here are some examples:

1 value: avg, median, count, etc.
2 values: min+max, upper/lower quartiles, etc.
3 values: min+avg+max, upper/med/lower quart., etc.
5 values: five-number summary (boxplots), etc.

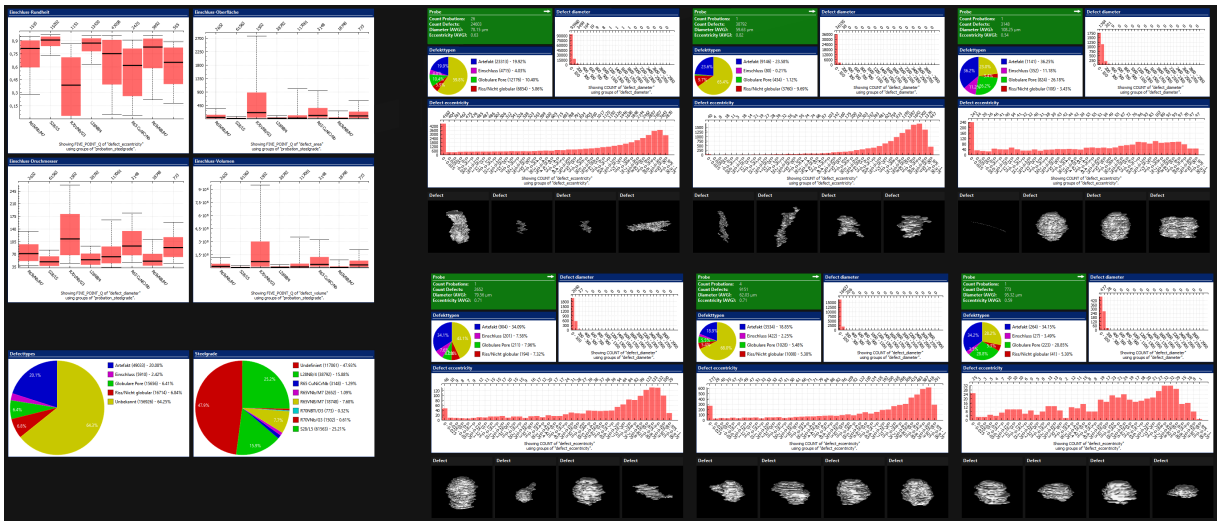

Figure 2: *A screenshot of the main window of the tool. In the overview on the left, uncertainties of four different outcome measurements are shown for different steel grades (bar charts). Two pie charts show the distribution of two categorical attributes. On the right, each steel grade is visualized in greater detail. Each steel grade uses the same layout, which enables easier comparison. Additionally, for every steel grade, four of the most significant defects are shown.*

For categorical data dimensions, like the steel type, we cannot use these numerical statistics. Instead, statements about most occurrence are given as textural information, utilizing the statistical function *mode*.

Besides pure processing of data by humans based on information visualization, we also added machine-based **Data Mining** [KMS*08] in the level overview via a tree traverse through the level of interest, thus horizontally. While data from lower and higher levels can be used during the mining process, the mining result will be an item or cluster in the level of interest (Fig. 1, green). The mining results are visualized as a recommendation to the user. This is very valuable in combination with level filters. For instance, the user may choose to analyze samples from the current week only. At first sight, exceptional samples from the current week are seen. That approach was also chosen for the level detail. Each sample or cluster has a large list of defects attached to it. The data mining process here is done vertically. Exceptional defects belonging to the current sample are found and recommended to the user, as are exceptional samples in the case of clustered data (Fig. 2, four exceptional defects recommended for each cluster).

6. Use Case

This paragraph describes the sequence of a typical analysis process:

Users choose one of three categories of objects of interest. These are *defects*, *samples* or *aggregates*. After choosing aggregates, they specify one or more data attributes that define

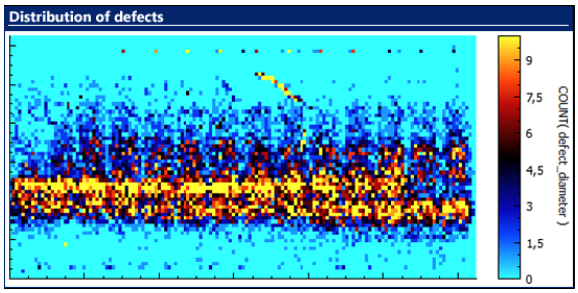

Figure 4: *Visualizing the defect bands for various steel types.*

the aggregates. They then select the categorical data attribute *steel type*, so that samples are grouped by their steel type. In this way, users are now comparing all the steel types available in the database.

Users may also choose to focus on a smaller set of steel types. To do so, they employ filtering on some of the process parameters, such as selecting steel types with a melting temperature of about 1500 °C. and additional oxygen blown into the melt. During all interaction with the GUIs the screen is constantly updating to address the user's requests. The overview and the detail view use some standard layouts that have been defined in a previous analysis session (Fig. 2).

In the detail view, users scroll through the different steel types and see how greatly they differ in the defect sizes they contain (histogram of diameter) and defect types that occur

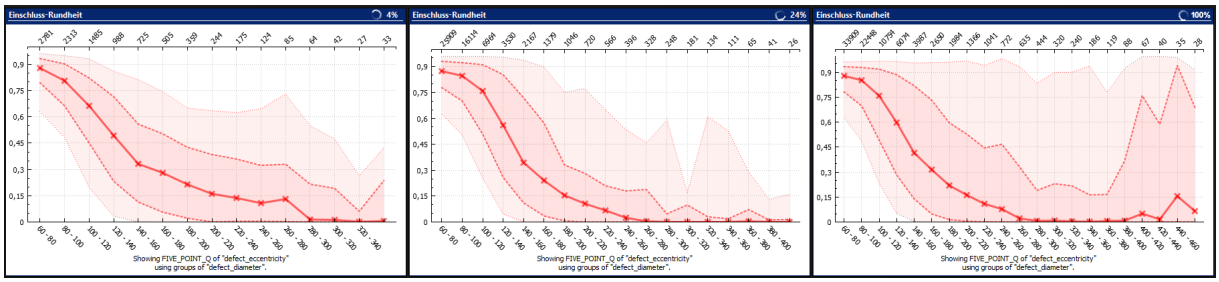

Figure 3: *Visualizing the iterative loading process of a statistical plot, showing the correlation of defect diameter and defect eccentricity: 4%, 28% and 100% of the data used.*

in them (pie chart of defect type). To determine whether there are also differences in the locational distribution of the defects within the sample, the user changes the layout to *distribution of defects*, as created in a previous session. A binned scatter plot, representing the rectangular sample format, displays a band where most of the defects are located (Fig. 4). The defect bands differ from steel type to steel type. Sometimes it is broader or wider; it may contain holes or be located differently within the sample. The user sorts the steel types according to different process parameters to see if there is a correlation between the defect band and certain process parameters.

In another analysis, the hypothesis was examined that the largest defects are more spherical and thus have a high degree of eccentricity. To do so, in the overview area, the user creates a graph showing the defect eccentricity against the defect diameters. The graph shows the average defect eccentricity for defects diameters in steps of 10μm. It seems that the hypothesis was correct: On average, the largest defects are more spherical.

Wondering why the variance is very high for large defects, the user then decides to undertake a variance analysis and edits the graph to show not only the average, but also a five-point summary. Even in between the upper and lower quartiles, where 50% of the defects having an eccentricity value around the middle are located, there are defects with no eccentricity and full eccentricity (Fig. 3). The user decides to create groups based on the defect type. This visualization strengthens and refines the hypothesis. The large defects with low eccentricity are the longish cracks, while the non-metallic inclusions are, in fact, spherical with a low variance (Fig. 5).

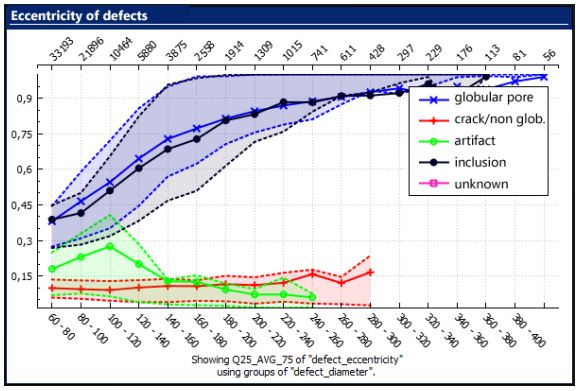

Figure 5: *Showing a correlation for different defect types.*

7. Discussion

The speed of the analysis tool is much better when compared to older versions of the software, which calculated everything in one step by database. Performance problems occurred when users changed input very quickly and ongoing calculations could not be completed. This was noticeable especially when users scrolled through a large number of clusters in the side-by-side views. By calculating the statistics incrementally, it is possible to cancel further unneeded calculation steps, greatly improving the degree of interactivity. By implementing the seek operation as described, there is only negligible overhead for paging. Calculating the quantiles with the naive algorithm in the database was significantly slower than the new method of retrieving the data batch-wise and approximating the answer. In general, by examining the data size and the statistical function of interest, we can adapt the way we process the data to meet the optimal requirements for execution behavior, speed and precision. There are algorithms which guarantee ε-approximate answers and fulfill our requirements. These algorithms guarantee an error in the approximation with an upper limit of ε. They will be implemented in future work. With these kinds of algorithms, as described, for instance, in [ZW07], we can more finely adjust the system with regard to the tradeoff between speed and precision in each iteration.

We collected **user feedback** from the steel experts who will use the tool and researchers involved in the project. Most wanted statistical function to be the quantile estimation. Compared to variance, user can see the distribution of

the values above and below the average independently. Variance only shows a general uncertainty in both directions. Feedback about performance was ambiguous. The steel facility employees work with systems that have very slow response times. Accidental user inputs are penalized by long waiting times before these inputs can be corrected. On the one hand, our new way of dealing with data was accepted. On the other, there was concern because its degree of reliability during the loading process was unknown. However, this was counterbalanced by the system's greater responsiveness. In many cases the approximations of visualizations hit the final visuals very fast due to high probability of similar data to come. We are exploring new input methods, so that all the available data is used only when explicitly wanted. This innovation would safeguard the infrastructure. Showing some data in comparison with reference data was adopted and desired for all kind of visualizations that show a single sample. For instance, when visualizing a sample, experts do not just see the facts about the collected data, but can evaluate and rate the sample compared to a reference. By defining the reference data dynamically — including rules to define similarity — great opportunities arise. Reference data is not as much of interest when clusters are visualized. Comparing multiple clusters side by side was preferred.

8. Acknowledgments

This work was funded by the European Regional Development Fund within the *Ziel 2 programme 2007-2013*.

References

[ACH*12] AGARWAL P. K., CORMODE G., HUANG Z., PHILLIPS J., WEI Z., YI K.: Mergeable summaries. In *Proceedings of the 31st Symposium on Principles of Database Systems* (New York, NY, USA, 2012), PODS '12, ACM, pp. 23–34. 4

[AMP*13] AGARWAL S., MOZAFARI B., PANDA A., MILNER H., MADDEN S., STOICA I.: Blinkdb: queries with bounded errors and bounded response times on very large data. In *Proceedings of the 8th ACM European Conference on Computer Systems* (2013), ACM, pp. 29–42. 3

[BBPL14] BÜRGER F., BUCK C., PAULI J., LUTHER W.: Image-based object classification of defects in steel using data-driven machine learning optimization. *Proceedings of VISAPP 2014 - International Conference on Computer Vision Theory and Applications* (1014). 1

[FPDs12] FISHER D., POPOV I., DRUCKER S., SCHRAEFEL M.: Trust me, i'm partially right: Incremental visualization lets analysts explore large datasets faster. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2012), CHI '12, ACM, pp. 1673–1682. 3, 4

[GK04] GREENWALD M. B., KHANNA S.: Power-conserving computation of order-statistics over sensor networks. In *Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (New York, NY, USA, 2004), PODS '04, ACM, pp. 275–285. 4

[HBT*12] HERWIG J., BUCK C., THURAU M., PAULI J., LUTHER W.: Real-time characterization of non-metallic inclusions by optical scanning and milling of steel samples. *Proceedings SPIE* (2012). 1

[HHW97] HELLERSTEIN J. M., HAAS P. J., WANG H. J.: Online aggregation. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 1997), SIGMOD '97, ACM, pp. 171–182. 3, 4

[JC85] JAIN R., CHLAMTAC I.: The p2 algorithm for dynamic calculation of quantiiles and histograms without storing observations. *Communications of the ACM (CACM) 28*, 10 (1985), 1076–1085. 4

[KIML11] KERSTEN M. L., IDREOS S., MANEGOLD S., LIAROU E.: The researcher's guide to the data deluge: Querying a scientific database in just a few seconds. *PVLDB Challenges and Visions* (2011). 3

[KLM*12] KONYHA Z., LEŽ A., MATKOVIĆ K., JELOVIĆ M., HAUSER H.: Interactive visual analysis of families of curves using data aggregation and derivation. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies* (New York, NY, USA, 2012), i-KNOW '12, ACM, pp. 24:1–24:8. 2

[KMS*08] KEIM D. A., MANSMANN F., SCHNEIDEWIND J., ZIEGLER H., THOMAS J.: Visual analytics: Scope and challenges. *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics, Springer, Lecture Notes In Computer Science (lncs)* (2008). 6

[MTL78] MCGILL R., TUKEY J. W., LARSEN W. A.: Variations of box plots. *The American Statistician 32*, 1 (1978), 12–16. 5

[PKRJ10] POTTER K., KNISS J., RIESENFELD R., JOHNSON C. R.: Visualizing summary statistics and uncertainty. *Computer Graphics Forum (Proceedings of Eurovis 2010) 29*, 3 (2010), 823–831. 3

[PWB*09] POTTER K., WILSON A., BREMER P.-T., WILLIAMS D., DOUTRIAUX C., PASCUCCI V., JOHNSON C.: Ensemble-vis: A framework for the statistical visualization of ensemble data. In *Data Mining Workshops, 2009. ICDMW '09. IEEE International Conference on* (2009), pp. 233–240. 3

[TBL14] THURAU M., BUCK C., LUTHER W.: IPFViewer - A Visual Analysis System for Hierarchical Ensemble Data. *International Conference on Information Visualization Theory and Applications, IVAPP 2014* (1014). 1, 2, 3, 5

[TC06] THOMAS J. J., COOK K. A.: A visual analytics agenda. *Computer Graphics and Applications, IEEE 26*, 1 (2006), 10–13. 2, 3

[Tuf90] TUFTE E.: *Envisioning information*. Graphics Press, Cheshire, CT, USA, 1990. 5

[WBWK00] WANG BALDONADO M. Q., WOODRUFF A., KUCHINSKY A.: Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (New York, NY, USA, 2000), AVI '00, ACM, pp. 110–119. 5

[WP09] WILSON A. T., POTTER K. C.: Toward visual analysis of ensemble data sets. In *Proceedings of the 2009 Workshop on Ultrascale Visualization* (New York, NY, USA, 2009), UltraVis '09, ACM, pp. 48–53. 2, 3

[ZW07] ZHANG Q., WANG W.: A fast algorithm for approximate quantiles in high speed data streams. In *Scientific and Statistical Database Management, 2007. SSBDM '07. 19th International Conference on* (July 2007), pp. 29–29. 7