Vision, Modeling, and Visualization (2017)
M. Hullin, R. Klein, T. Schultz, A. Yao (Eds.)

User-study based optimization of fast and accurate Mahalanobis brushing in scatterplots

Chaoran Fan and Helwig Hauser

University of Bergen, Norway, ii.UiB.no/vis

Abstract

Brushing is at the heart of most modern visual analytics solutions with coordinated, multiple views and effective brushing is crucial for swift and efficient processes in data exploration and analysis. Given a certain data subset that the user wishes to brush in a data visualization, traditional brushes are usually either accurate (like the lasso) or fast (e.g., a simple geometry like a rectangle or circle). In this paper, we now present a new, fast and accurate brushing technique for scatterplots, based on the Mahalanobis brush, which we have extended and then optimized using data from a user study. We explain the principal, sketch-based model of our new brushing technique (based on a simple click-and-drag interaction), the details of the user study and the related parameter optimization, as well as a quantitative evaluation, considering efficiency, accuracy, and also a comparison with the original Mahalanobis brush.

CCS Concepts

• ***Human-centered computing*** → *Interaction techniques;* • ***Computing methodologies*** → *Optimization algorithms;*

1. Introduction

In interactive visual data exploration and analysis, linking and brushing is a central and well-established interaction technique for relating data aspects across coordinated multiple views [Mun14, Rob07]. The principles of brushing were first described by Becker and Cleveland [BC87], who defined brushing as an interactive method for painting a group or subsets of points with a square, circle, or a polygon, i.e., the brush. A key functionality in standard instances of coordinated multiple views is that brushing leads to a consistent highlighting of the selected data in all linked views, for example, by coloring them consistently. This amounts to one important form of focus+context visualization [Hau05], enabling the fast and effective exploration of data relations, which are too challenging to show in just one view.

As popular and common as linking & brushing has become in modern visual analytics solutions, already many different techniques for brushing have been realized, including many variants from the following categories:

- *brushing using simple geometries*—examples of this most common approach include the rectangular or circular brushing on scatterplots, line-brushing on data graphs [KMG*06], etc.
- *lassoing*—the user selects data subsets by drawing a geometrically detailed lasso around a target group of item representations
- *logical combinations of simple brushes*—the user refines the data selection by using multiple brushes and combining them using logical operators [MW95, DGH03]
- *sketch-based brushing*—the user sketches a shape onto a visualization and some selection heuristic is used, usually exploiting a related similarity function, to determine which data are actually selected [MKO*08]

Each brushing technique can be discussed in terms of its advantages and disadvantages and two criteria are particularly important:

- *efficiency*—how fast is the brushing interaction; does it enable a fluid data exploration/analysis [EVMJ*11, TKBH17]?
- *accuracy*—does the brushing interaction lead to a selection of exactly the data subset, which the user wished to select?

In many cases, there is an unfortunate competition between these criteria: Many brushing techniques are indeed fast—we think of a brushing technique to be fast, if only one click (or only very few atomic interactions of that kind) are needed to actually specify the brush, leading to a swift user–computer dialogue during the data exploration/analysis [CRM91]. Classical examples include the use of simple brushing geometries (rectangles, circles, etc.) as well as sketched brushes, where only a quick gesture is used for brushing. A common disadvantage of all these fast techniques is that it can be difficult to accurately brush a particular data subset.

On the other hand, we certainly find brushing techniques that are fully accurate—likely with lassoing being the most prominent example besides others such as the logical combination of simple brushes. With these techniques, it is straight-forward to select subsets of interest accurately. This benefit, however, comes at the price of being slower, in general—specifying a lasso, point by point, for

DOI: 10.2312/vmv.20171262

example, easily becomes a unit task by itself [CRM91], potentially interrupting the exploration/analysis process.

In our research, we have studied the question of how close one can get to successfully integrating both criteria in one technique and in the following we present a successful solution (we chose the example of brushing in scatterplots as our study case—we think, however, that our principle approach is extensible to other views and according brushes). Our solution is based on an extended version of the previously published Mahalanobis brush [RSM*16], which we have extended and further optimized using data from a user study with 50 participants. Our quantitative evaluation shows that we significantly improved the brushing accuracy from ≈65% (original Mahalanobis brush) to ≈92% (our new technique). Our technique is as fast as a simple click-and-drag interaction—the original Mahalanobis brush required only one location (one click), but depended, in addition, on an off-screen size parameter (overall brush size).

Since brushing is central in most modern visual analytics systems, we see our research very relevant—optimizing at the heart of a common procedure has the strong potential of significant impact.

2. Related work

In the following, we review a few pieces of important related work, before going into detail with respect to our new brushing technique. We first review, in short, some critical works concerning brushing for visual analytics, before we then discuss related work concerning the optimization of interaction techniques.

2.1. Brushing techniques

Many variations of brushing have been proposed over the years, each with its own strengths and weaknesses—for example, in terms of their ease of use and the degree of control which the user has. Brushing is intrinsically based on the interaction between the user and the system, often a combination of mouse/cursor motions and button clicks. Less usual methods, based on eye/head tracking, for example, or gestures in a virtual reality environment, have also been proposed [YA01].

Brushing in scatterplots is often based on the use of simple geometric shapes such as a rectangle or circle to select the data items, or using a lasso to specify the brush region more accurately.

Several extensions to simple brushing have been published, including techniques to formulate more complex brushes by combining multiple brushes using logical operators. Martin and Ward [MW95], for example, allow the user to configure composite brushes by applying logical combinations of brushes, including unions, intersections, negations, and exclusive or operations.

Similarity brushing [NH06, MKO*08] is an interesting alternative in that it is based on a fast and simple sketching interaction—the user uses a swift and approximate gesture (for example, drawing an approximate shape that the data should follow) and then a similarity measure is used to identify, which data items actually are brushed by such an advanced brush. This way, the interaction is fast, but likely not 100% accurate.

Recently, the Mahalanobis brush was presented as an interesting alternative for brushing scatterplots [RSM*16]. The user uses a simple interaction (like a simple click into the center of some coherent data subset to be selected) for brushing the data. The link between the interaction and the actual selection is realized on the basis of a simple analysis of the underlying data (a local covariance matrix indicates the overall shape and orientation of the data to be brushed, forming then the basis for a local Mahalanobis metric, which is then used as a distance measure to select the data).

While this technique is already giving quite good results, it still has certain limitations, including: a non-optimized selection of the local context for the Mahalanobis computation (improved in our solution), at least one off-screen parameter for the brush size (no free parameter in our solution), and empirical parameters (we use the data from a user study to optimize the relevant parameters).

2.2. Optimization based on user data

Obviously, the user plays a key role during all sorts of interaction. Thus, efforts have been invested to take the user behavior into account, when improving the performance of interaction techniques.

The design of adaptive user interfaces, for example, is one of the most classical examples in this respect, enabling the interface to recognize user action plans by tracing and analyzing the user's action sequences [Lan97, LWH03].

Lieberman et al. [LVDV99] developed the "Let's Browse" application to assist the user browsing websites by tracking the user's behavior and predicting items of interest, accordingly.

In the mobile phone architecture design area, Shye et al. [SSM09] developed a logger application for mobile phones and released it "into the wild" to collect the traces of real users. Then, they used these traces for characterizing the power consumption on the mobile phones, eventually leading to the development of applications that optimize the consumption of battery power.

Considering visualization research, in particular, we cannot find a lot of related work (in terms of optimizing interaction techniques, based on user data). Instead, we see this as a highly interesting chance for interesting and relevant innovation.

3. The principal approach

The overall goal of our research was to devise a brushing technique, which is both fast and accurate. In order to get as close as possible to both goals, we used the following principal approach (also illustrated in Fig. 1):

In order to be fast, we excluded techniques that would require the user to do multiple basic interactions in order to define just one brush (like a lasso, for example). We also wished that the users would not have to adjust any off-screen parameters, potentially interrupting their explorative/analyical procedure.

In order to be accurate, we decided to raise our expectations over the use of simple geometries—mostly due to their limited abilities to accurately select specific data subsets, in particular in "crowded" regions of a data visualization. Accordingly, we concluded that a sketching interaction, combined with a carefully modeled selection heuristic, would be the right principal approach in our case.

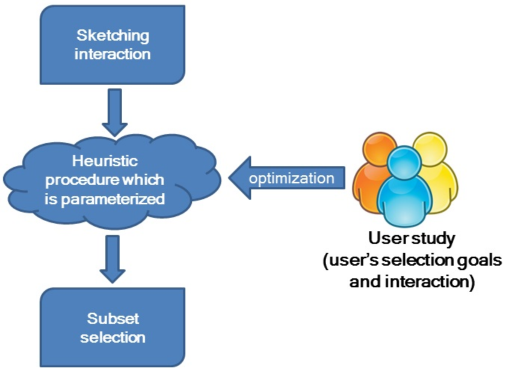
Sketching interaction
Heuristic procedure which is parameterized
optimization
User study (user's selection goals and interaction)
Subset selection

Figure 1: *Illustration of our principal approach: To be fast, we use sketching as interaction; to derive which data to actually brush, we use a heuristic with parameters that we optimize using data from a user study.*

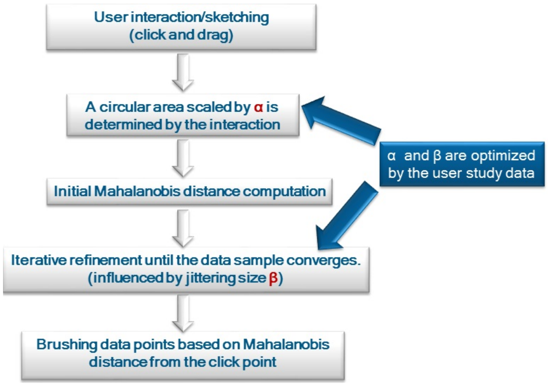
User interaction/sketching (click and drag)
A circular area scaled by α is determined by the interaction
α and β are optimized by the user study data
Initial Mahalanobis distance computation
Iterative refinement until the data sample converges. (influenced by jittering size β)
Brushing data points based on Mahalanobis distance from the click point

Figure 2: *Overview of our fast and accurate brushing technique: the user clicks into the middle of the data subset to be selected and drags the pointer to the border of the subset (sketching interaction); iteratively, a selection of points around the click-point is chosen, based on local covariance information, until convergence; a selection is made based on the Mahalanobis distance from the click-point. Two parameters, α and β, related to the sample size, before iterating, and to some jittering, stabilizing the technique, influence the performance and we optimize them using our user study.*

Typically, the heuristic, which determines the data subset to be brushed, based on a simple sketching interaction, is parameterized and different parameters will lead to different brushing results, even when the user interaction (the sketch) is exactly the same. One opportunity is, of course, to expose these parameters in the user interface, requiring the user to adjust parameters in order to achieve an expected result. Clearly, this is not, what we need. Instead, our goal was a technique, which does not require any adjustment of technique parameters by the user such that the user can concentrate on the fast and accurate interaction with the data.

In order to optimize the performance of our selection heuristic, we therefore conducted a user study with 50 participants, in which we collected information about both the brushing goals (which dataset subset did user wish to brush) as well as the associated interaction (which click-and-drag gesture would the user do to actually selected the targeted data subset). A subset of the acquired information from this user study (training data) was then used to optimize the relevant parameters of our selection heuristic.

4. Fast and accurate brushing in scatterplots

Figure 2 provides an overview of the new brushing algorithm. We use a simple click-and-drag interaction for sketching the data subset to brush (click into the middle of the targeted data subset and drag the pointer to the boundary of the subset). The start- and end-point of this interaction provides us with a first hint concerning the size of the data subset, which the user wishes to brush. Similarly to the original Mahalanobis brushing technique [RSM*16], we also consider a circular data subset, centered around the start-point of the interaction, and estimate the shape and orientation of the data in this region by looking at the local covariance information. As an improvement, we then start an iteration, until convergence, that refines this data subset selection, based on the local covariance information. After convergence, we eventually make a selection of data points, based on the Mahalanobis distance, taking the local covariance information into account. In the following, we go into more details with respect to the individual components of our solution.

Since the Mahalanobis distance, introduced by P. C. Mahalanobis in 1936 [Mah36], is central in our technique, we briefly review it here. It is based on the correlation between data variables and helps with the identification and analysis of patterns in the data. It is unit-less and scale-invariant, which is a useful way for determining the similarity of an unknown sample to a known one. It differs from the Euclidean distance in that it measures with respect to the available data. Mathematically, the Mahalanobis distance between vectors **a** and **b** is defined by

$$d_\Sigma(\mathbf{a},\mathbf{b}) = \sqrt{(\mathbf{a}-\mathbf{b})^\top \Sigma^{-1} (\mathbf{a}-\mathbf{b})} \qquad (1)$$

where Σ is the covariance matrix of the sample (its diagonal elements consist of the variance of each variable and the off-diagonals are the mutual covariances). The location of equal Mahalanobis distances forms an ellipse around the sample mean (in 2D).

The click-and-drag interaction, which we use to sketch the data subset to be selected, provides two locations that subsequently are essential as input to our technique, i.e., the click-point $\mathbf{s} = (s_x, s_y)^\top$ and the end-point of the drag-interaction $\mathbf{e} = (e_x, e_y)^\top$.

4.1. Mahalanobis distance computation

Our technique is based on the approach to only consider the local covariance structure of a data subset around the click-point **s**. It is an important part of our overall approach to determine, which data subset should be used for this computation. In our technique, we do two steps to get these sample points.

Initially, we consider a circular area with the radius $\alpha \cdot d_E(\mathbf{s},\mathbf{e})$, where α is a weighting factor and $d_E(\mathbf{s},\mathbf{e})$ is the Euclidean distance between **s** and **e**. All points within this circle are used to compute the first instance of the local covariance information, Σ_1.

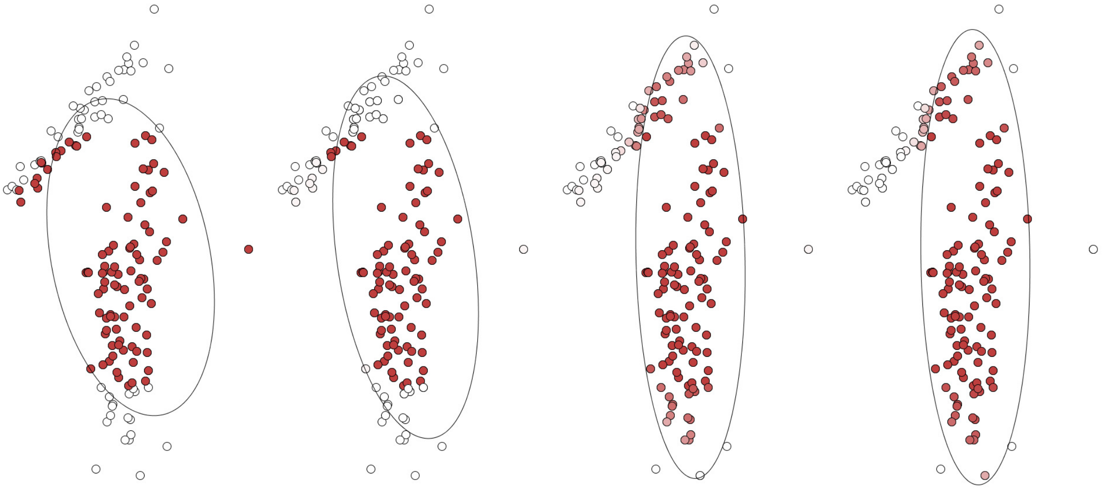

Figure 3: *Illustration of the convergence process: Initially, points in a circular neighborhood (red points on the left) are used to compute Σ_1 (illustrated by the ellipse); this leads to a new selection (red points in the 2^{nd} panel) and the computation of Σ_2; the ellipse in panel #3 illustrates Σ_{10} and the one on the right Σ_{100} (converged). Shades of red show points which are weighted, accordingly.*

Next, we consider all points within a Mahalanobis ellipse, based on Σ_1 and sized according to $d_\Sigma(\mathbf{s},\mathbf{e})$. This usually leads to a new data subset, which is similar but still different from the data subset as determined by the initial circle. Usually, this new subset is already a better approximation of the data subset to be brushed. To obtain an even more reasonable sample, we refine the sample iteratively by replacing them with the points in the Mahalanobis ellipse which is updated every iteration according to the samples in last iteration. Most often, we observe a quick convergence of this process. However, it can happen, that small fluctuations appear, for example, between two selections that replace each other, iteratively. Therefore, we stabilize the convergence by enabling the partial consideration of data points around the click-point, leading to a solution that is then based on a weighted covariance matrix [Gou09].

4.2. Weighted covariance matrix

In a weighted sample, each vector $\mathbf{x}_i$ is assigned a weight $\omega_i \geq 0$. Without any loss of generality, we assume normalized weights:

$$\sum \omega_i = 1 \tag{2}$$

Then the weighted mean vector $\bar{\mathbf{x}}$ is given by

$$\bar{\mathbf{x}} = \sum \omega_i \mathbf{x}_i \tag{3}$$

and the elements Σ_{jk} of the weighted covariance matrix Σ are

$$\Sigma_{jk} = \frac{1}{1-\sum \omega_i^2} \sum_i \omega_i (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k) \tag{4}$$

In our solution, we use an impact factor $\varepsilon_{i,0} = \varepsilon = 0.95$ that we apply to all the initial samples (as well as 0 to the remaining points). During the iteration, we update the impact factor for the points in the current Mahalanobis ellipse as follows:

$$\varepsilon_{i,n} = \varepsilon_{i,n-1} + \varepsilon^{n+1} \tag{5}$$

where n is the number of the iteration.

In order to obtain the weights, we normalize the impact factors:

$$\omega_i = \frac{\varepsilon_{i,n}}{\sum_j \varepsilon_{j,n}} \tag{6}$$

The above described mechanism achieves the following: with more iterations (growing n), the relative update of the impact factors (after normalization) decreases increasingly (the powers of ε^{n+1} drop below $1/3$ already after 20 iterations), suppressing any possible fluctuations and securing convergence at a high-quality result.

We considered to also optimize the value of ε, but found this unnecessary, because its specification could be determined by the number of iterations, which seemed to be more than sufficient to guarantee a good result (20 iterations is on the safe side). After convergence, the points with a positive impact factor are used to calculate the final, weighted covariance matrix Σ.

Figure 3 shows the weights by using different shades of red. We can clearly see that the points which are stable in the Mahalanobis ellipse are shown in darker red.

In certain situations, the covariance matrix can be singular, also (in particular, when all sample points are along a line) and no 2D Mahalanobis distance can be computed in such a case. We thus add a small jittering (scaled according to β) to the sample points to avoid this situation. The random jitter used in our work is based on a Gaussian distribution with the mean of 0 and the standard deviation of 1 pixel.

4.3. Selecting a data subset using a selector

Based on the converged covariance matrix, a selector is used to determine the actually brushed data points. The selector is also based on the weighted covariance matrix Σ_n: We use the Mahalanobis ellipse, according to Σ_n, that corresponds to point $\mathbf{m} = \mathbf{s} + \alpha(\mathbf{e} - \mathbf{s})$. Accordingly, the set of all brushed points is defined as

$$\{\, \mathbf{x}_i \mid d_{\Sigma_n}(\mathbf{s},\mathbf{x}_i) \leq d_{\Sigma_n}(\mathbf{s},\mathbf{m}) \,\} \tag{7}$$

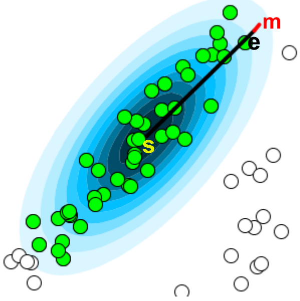

Figure 4: *Selecting data points, based on the local, weighted covariance information: **s** and **e** denote the start- and end-points of the click-and-drag interaction; the ellipses illustrate Σ_n. **m** lies on the Mahalanobis ellipse which acts as the eventual selector.*

Figure 4 shows contours of the selector, selecting all green points based on **s** and **m** (all points within the Mahalanobis ellipse, which corresponds to location **m**).

5. User study

The new brushing technique has two not-yet-optimized parameters: α (the size of the circular area determining the initial sample, influencing also the selector) and β (jittering size). In order to achieve as accurate as possible brushing, we conducted a user study to get information about how users would use our technique to brush and what they actually wanted to select from the dataset (ground truth). Based on this information, we then did an optimization of α and β. In the following, we provide details about this user study.

5.1. Study datasets

In our user study, we used six representative datasets as shown in Figure 5. In order to use as representative datasets as possible, we looked at a variety of sample data and according scagnostics. Scagnostics is short for Scatterplot Diagnostics, first mentioned by John and Paul Tukey [TT88] to help characterize scatterplots and find interesting structures according to density, skewness, shape, outliers, etc. Wilkinson et al. [WAG05] revived the topic and implemented concrete measures in the R package scagnostics. We wished to choose sample datasets with mutually as different as possible scagnostics, aiming at a healthy spread of scatterplots of different type. Accordingly, we chose four scatterplots (A, B, C, D) from the Boston Housing data [HR78], which consists of 14 variables and 91 different scatterplots, that had maximally different scagnostics from each other. We then complemented this set of four datasets with two additional ones: E shows Gauss-type clusters (standard case) and F is a path-based spectral clustering dataset (particularly difficult case due to the bent, elongated outer cluster).

5.2. User study process

Our user study consisted of three parts:

In the first part, all users were asked to look at a scatterplot. Then they were instructed to choose a particular data subset according to a question that was posted along with the scatterplot—we used one out of three questions in any case: choose a large cluster, choose a small cluster, and choose an elongated cluster.

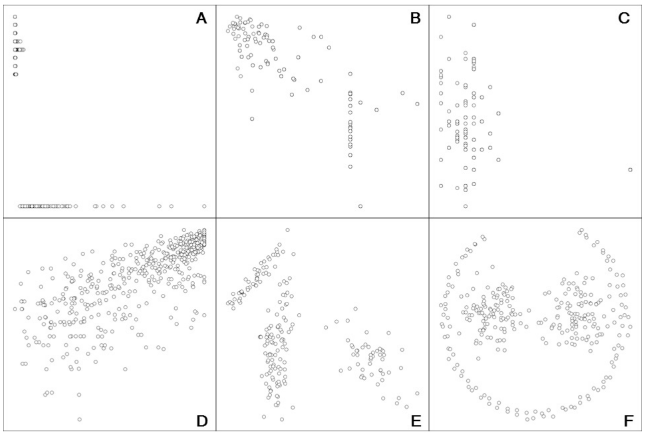

Figure 5: *Overview of the six datasets that we used in the user study: A–D show Boston housing data (with as different scagnostics as possible); E shows Gaussian clusters and F shows path-based spectral clusters (as a particuarly difficult case).*

Then, the users were asked to use a lasso to accurately select the points which they had choosen in the first step. This was done in order to record the user's brushing goal (later used as ground truth during the optimization).

In the last step, the users were required to use our new technique to select the same points which they had already selected in the second step. To do so, the users had to click on the center of the points to choose and then drag the pointer, while holding down the button of the mouse to the border of the points, and then release to finish the selection.

In the user study, we recorded all points selected by the lasso (the brushing goal), the sketching interaction (i.e., the start point and the end point of our new brushing technique), and the time spent on the interaction(think time is not included) during both of these two techniques. 50 individuals, all students or employees from the University of Bergen, Norway, participated in our study. Each one was asked to do 12 selections (six different datasets and two different questions each). We formulated two questions for each dataset in advance based on our perception of the datasets. Before the users were doing their selections, we presented our new Mahalanobis brushing in a training session, where we showed the main features by examples of brushing a test dataset. These sessions took approximately 10 minutes and the participants were free to interrupt for questions and to take over the software to experiment with the new brushing technique until they were comfortable to do the study.

6. Optimization

Figure 6 demonstrates the influence of α on brushing results compared to the user goal (encoded by color). The true positives (correctly brushed), true negatives (correctly not brushed), false positives (falsely brushed) and the false negatives (falsely left out) are

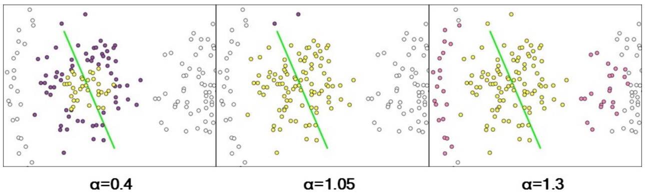

Figure 6: *Demonstration of the influence of different values of α: too small values of α will underselect, while too large values will overselect.*

colored in yellow, white, pink and purple, accordingly. The green line is the diameter of the circular area determined by the user sketch. We can easily see that there are more false negatives when α is too small (left). Conversely, more false positives appear when the value of α is too big (right). Concerning β, we need a sufficiently large β (to avoid a singular covariance matrix), while also β is bound to be small: with steadily increasing values of β, the structure of the brushed data gets increasingly diluted (for really large values of β, the Mahalanobis distance basically degenerates to Eucledian distances.

In the user study we collected 600 selections, of which we randomly chose 400 as training data, leaving 200 selections for the validation. In order to compare the similarity between the selection goal by the user and the corresponding results by our technique, we used the Dice coefficient as a cost function. The Dice coefficient is a similarity measure related to the Jaccard index, developed by Lee Raymond Dice [Dic45]. For sets X and Y, and the estimated parameters α and β, the coefficient can be defined as

$$s(\alpha,\beta) = \frac{2\,|X \cap Y|}{|X| + |Y|}$$

where $|X|$ and $|Y|$ are the cardinalities of the two sets. In our data training, X is the result of our brushing technique and Y is the user goal. In the case of optimal agreement, i.e., X=Y, $s(\alpha,\beta)$ equals 1, while in the case, where X and Y do not overlap, $s(\alpha,\beta) = 0$.

After collecting the ground truth (lasso data) from the user study as well as the click- and release-points from the sketching interaction, we were able to conduct a numeric optimization of α and β according to the following procedure, not involving the users anymore: Based on a particular choice of α and β, we execute our selection heuristic, using the datasets from the user study and the recorded interaction data, leading to a particular X(α,β)—this was then straight-forward to compare to Y (always the same, of course), leading to $s(\alpha,\beta)$, accordingly. We started with a large matrix of systematically different combinations of the two parameters, covering a domain, which for sure was big enough. Inspecting the s-values for all these combination lead us to further examining a more detailed subset of the parameter space (basically, we refined our optimization hierarchically, doing the refinement manually). Eventually, we ended up with the following optimal values for both parameters when obtaining a highest overall accuracy of 400 training selections: $\alpha = 1.05$ and $\beta = 11$ (wrt. a view size of 800×800).

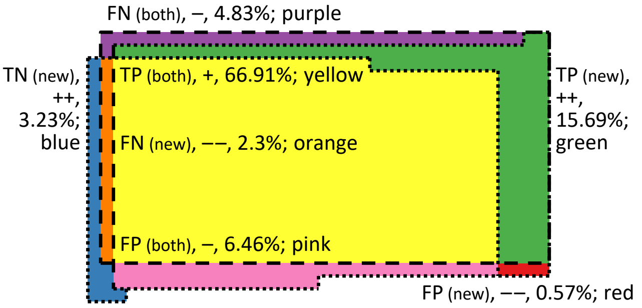
FN (both), –, 4.83%; purple
TN (new), ++, 3.23%; blue
TP (both), +, 66.91%; yellow
FN (new), ––, 2.3%; orange
FP (both), –, 6.46%; pink
TP (new), ++, 15.69%; green
FP (new), ––, 0.57%; red

Figure 7: *Statistics of the comparison between our technique and the original Mahalanobis brushing based on the user's goal (details in the text).*

7. Detailed discussion of accuracy

After the parameter optimization, we obtained the optimal value of α and β for our brushing technique ($\alpha = 1.05$ & $\beta = 11$). Based on this, we did a quantitative accuracy comparison with the previously published Mahalanobis brush [RSM*16] using the interaction information from our user study. Figure 7 shows a Venn-diagram-like visualization of this comparison.

The area surrounded by the dashed line represents the user goal, accumulated over all selections. The area surrounded by a solid line represents the brushing results by our technique while the dotted line surrounds the results by the old Mahalanobis technique. Same areas correspond to same numbers of brushed data points.

We calculated the percentages of how many data points fall in each of the eight possible overlap regions between the user's goal, our brushing, and the original brush after accumulating over all cases (the all-negative region corresponding to the overall context of points outside of all selections was left out from the visualization). The colors used in this visualization correspond also to the colors of points in the other scatterplots in this discussion section:

- least interesting are yellow points (both brushing technique succeed to select the point correctly (both true positive), purple points (both brushing techniques fail to select), and pink points (both techniques select falsely).
- more interesting are green points (the new technique succeeds, while the original fails), blue points (the original technique selected falsely, while the new one does not), orange points (the new technique fails to select, while the original did), and red points (the new technique selects falsely, while the original did not)—assuming the perspective of this paper, green and blue points are very good (better than the original)!

Based on the percentages as presented in Fig. 7, we can calculate the overall accuracy for the original technique to be ≈65% and for the new technique to be ≈92% (the very positive areas, green and blue, are significantly larger than the very negative results, orange and red). Next, a few cases are discussed in detail.

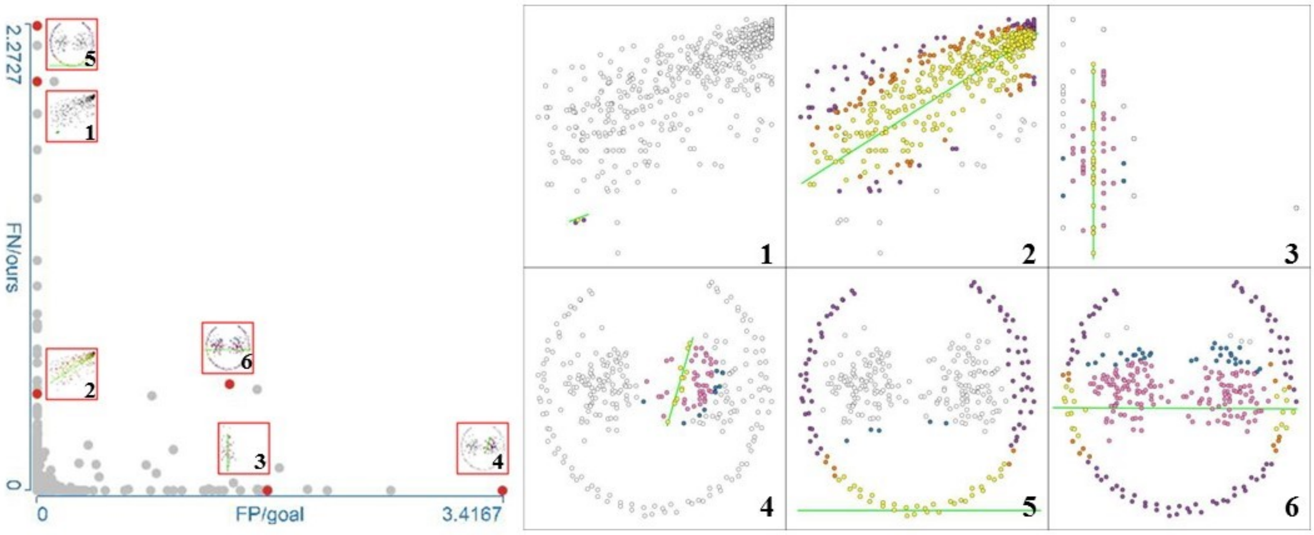

Figure 8: *Left: A visualization of how certain selected cases from the user study deviated in terms of accuracy. Right: Six (most extreme) cases of suboptimal matches between the user's goal and the new brushing technique (details in the text).*

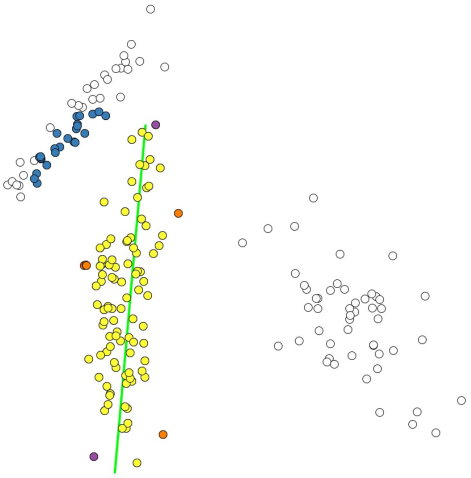

Figure 9: *An example of a good match between the user's goal and the new brushing technique.*

7.1. Good case analysis

Figure 9 shows a typical situation—our method performs very well, based on the weighted covariance information, but the original Mahalanobis brush results in a clearly worse selection (note the many blue points, i.e., points, which the original technique falsely selects, while the new technique does not).

7.2. Bad cases

Figure 8 (left) shows a scatterplot with statistical information for each selection result (with respect to the ratio of false negatives to our brushing result and false positives to the goal). Most results lie in the bottom-left corner (the good corner), only a few results show significant numbers of false negatives / false positives. We choose six cases, highlighted by red points in the scatterplot for a detailed analysis, shown also in detail on the right in Figure 8:

Case 1: Details of the user's interaction have a big influence when selecting very small subsets (here, the start point of the user interaction deviates a bit from the center point of the target cluster, leading to a bad performance in this case).

Case 2: Here, the new technique is too conservative and selects to few points (the old technique tends to select more circular regions).

Case 3: For scatterplots with linear structures that also are close to each other, our techniques selects wider clusters than what users seemingly wish (in this data, several users wished to select individual "lines" of data points).

Case 4: Here, we think that it is close to impossible to correctly predict the user goal computationally.

Cases 5 and 6: In both cases, the user wished to select the outer ring—something, which is by design impossible with our (linear) selection technique (the click-and-drag interaction gives too little information to correctly select such "advanced" clusters).

8. Discussion, Conclusion, and Future work

In this paper, we have made a serious attempt to contribute an improvement to a central procedure in many modern visual analytics solutions, i.e., to brushing (scatterplots). We have described and exercised an approach, which is all-too-little seen in the visualization literature, i.e., a user study based optimization of visualization parameters (here the crucial parameters of the new interaction technique). We could demonstrate, quantitatively, that we significantly improve the accuracy of Mahalanobis brushing from ≈65%

to ≈92%, while still using a very fast interaction technique (click-and-drag, the average time spent is only 41% of Lasso in our user study).

After the completion of this study, we now also have a better understanding of the influence of the two optimized parameters, α and β, which consequently could lead to a further improved approach. With respect to α (scaling parameter for the selection), we clearly see the need for an optimization as we performed it, but it may be advisable to search for an improvement of the actual optimization technique in order to compensate the stochastic influence of the jittering on the results (without having worked through the complete cycle yet, we see indications that one should arrive at a very similar, optimal value of α). With respect to β (amount of jittering), we found out that β really needs to become substantially large (like 100, or so), before a measurable negative impact is clearly detectable—as long as a small, non-zero value of β avoids the singularity of the covariance matrix, the results are good.

In terms of efficiency, it only costs 20ms for the computation of brushing 2000 points, which enables the user obtain the brushing result in real time. We see the potential that this work can motivate others to follow a similar approach in their visualization research, i.e., to do an automatic optimization of visualization parameters, based on data from a corresponding user study.

Certainly, we see several opportunities for future work, including

- extending our principal approach to other views and according brushes
- further improving the selection heuristic by including kernel density estimation to even better delineate the sample for computing the selection [R*56, Par62]

Acknowledgements

We thank the participants of our user study. We are also grateful to Krešimir Matković for valuable exchange related to the original Mahalanobis brush [RSM*16].

References

[BC87] Becker R. A., Cleveland W. S.: Brushing scatterplots. *Technometrics 29*, 2 (1987), 127–142. 1

[CRM91] Card S. K., Robertson G. G., Mackinlay J. D.: The information visualizer, an information workspace. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems* (1991), CHI '91, ACM, pp. 181–186. 1, 2

[DGH03] Doleisch H., Gasser M., Hauser H.: Interactive feature specification for focus+context visualization of complex simulation data. In *Proc. of the Symp. on Data Visualisation* (2003), VISSYM '03, pp. 239–248. 1

[Dic45] Dice L. R.: Measures of the amount of ecologic association between species. *Ecology 26*, 3 (1945), 297–302. 6

[EVMJ*11] Elmqvist N., Vande Moere A., Jetter H.-C., Cernea D., Reiterer H., Jankun-Kelly T. J.: Fluid interaction for information visualization. *Information Visualization 10*, 4 (Oct. 2011), 327–340. 1

[Gou09] Gough B.: *GNU scientific library reference manual*. Network Theory Ltd., 2009. 4

[Hau05] Hauser H.: Generalizing focus+context visualization. In *Scientific Visualization: The Visual Extraction of Knowledge from Data*, Bonneau, Ertl, Nielson, (Eds.). Springer, 2005, pp. 305–327. 1

[HR78] Harrison D., Rubinfeld D. L.: Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management 5*, 1 (1978), 81–102. 5

[KMG*06] Konyha Z., Matkovic K., Gracanin D., Jelovic M., Hauser H.: Interactive visual analysis of families of function graphs. *IEEE Transactions on Visualization and Computer Graphics 12*, 6 (Nov 2006), 1373–1385. 1

[Lan97] Langley P.: Machine learning for adaptive user interfaces. In *Annual Conference on Artificial Intelligence* (1997), Springer, pp. 53–62. 2

[LVDV99] Lieberman H., Van Dyke N., Vivacqua A.: Let's browse: a collaborative browsing agent. *Knowledge-Based Systems 12*, 8 (1999), 427–431. 2

[LWH03] Liu J., Wong C. K., Hui K. K.: An adaptive user interface based on personalized learning. *IEEE Intelligent Systems 18*, 2 (2003), 52–57. 2

[Mah36] Mahalanobis P. C.: On the generalised distance in statistics. In *Proceedings National Institute of Science, India* (Apr. 1936), vol. 2, pp. 49–55. 3

[MKO*08] Muigg P., Kehrer J., Oeltze S., Piringer H., Doleisch H., Preim B., Hauser H.: A four-level focus+context approach to interactive visual analysis of temporal features in large scientific data. *Computer Graphics Forum 27*, 3 (2008), 775–782. 1, 2

[Mun14] Munzner T.: *Visualization Analysis & Design*. CRC Press, 2014. 1

[MW95] Martin A. R., Ward M. O.: High dimensional brushing for interactive exploration of multivariate data. In *Proc. of the 6th Conf. on Visualization* (1995), VIS '95, IEEE Computer Society, pp. 271–278. 1, 2

[NH06] Novotnỳ M., Hauser H.: Similarity brushing for exploring multidimensional relations. 2

[Par62] Parzen E.: On estimation of a probability density function and mode. *The annals of mathematical statistics 33*, 3 (1962), 1065–1076. 8

[R*56] Rosenblatt M., et al.: Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics 27*, 3 (1956), 832–837. 8

[Rob07] Roberts J.: State of the art: Coordinated multiple views in exploratory visualization. In *Proc. of CMV '07* (July 2007), pp. 61–71. 1

[RSM*16] Radoš S., Splechtna R., Matković K., Đuras M., Gröller E., Hauser H.: Towards quantitative visual analytics with structured brushing and linked statistics. In *Computer Graphics Forum* (2016), vol. 35, Wiley Online Library, pp. 251–260. 2, 3, 6, 8

[SSM09] Shye A., Scholbrock B., Memik G.: Into the wild: studying real user activity patterns to guide power optimizations for mobile architectures. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture* (2009), ACM, pp. 168–178. 2

[TKBH17] Turkay C., Kaya E., Balcisoy S., Hauser H.: Designing progressive and interactive analytics processes for high-dimensional data analysis. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan 2017), 131–140. 1

[TT88] Tukey J. W., Tukey P. A.: Computer graphics and exploratory data analysis: An introduction. *The Collected Works of John W. Tukey: Graphics: 1965-1985 5* (1988), 419. 5

[WAG05] Wilkinson L., Anand A., Grossman R. L.: Graph-theoretic scagnostics. In *INFOVIS* (2005), vol. 5, p. 21. 5

[YA01] Yang M.-H., Ahuja N.: *Face detection and gesture recognition for human-computer interaction*, vol. 1. Springer Science & Business Media, 2001. 2