MultiStream: A Multiresolution Streamgraph Approach to Explore Hierarchical Time Series

Erick Cuenca Pauta, Arnaud Sallaberry, Florence Ying Wang, Pascal Poncelet

MultiStream: A Multiresolution Streamgraph Approach to Explore Hierarchical Time Series

Erick Cuenca, Arnaud Sallaberry, Florence Y. Wang, and Pascal Poncelet

Abstract—Multiple time series are a set of multiple quantitative variables occurring at the same interval. They are present in many domains such as medicine, finance, and manufacturing for analytical purposes. In recent years, streamgraph visualization (evolved from ThemeRiver) has been widely used for representing temporal evolution patterns in multiple time series. However, streamgraph as well as ThemeRiver suffer from scalability problems when dealing with several time series. To solve this problem, multiple time series can be organized into a hierarchical structure where individual time series are grouped hierarchically according to their proximity. In this paper, we present a new streamgraph-based approach to convey the hierarchical structure of multiple time series to facilitate the exploration and comparisons of temporal evolution. Based on a focus+context technique, our method allows time series exploration at different granularities (e. g., from overview to details). To illustrate our approach, two usage examples are presented.

Index Terms—Streamgraph, Stacked Graph, Time Series, Aggregation, Multiresolution Visualization, Overview+detail, Focus+context, Fisheye

1 Introduction

A time series is a sequence of quantitative values taken at successive points in time. Sometimes, the evolution of these values over time contains patterns that are useful for data analysis. A common way to plot it is over a 2D Cartesian coordinates system, where the horizontal axis encodes the temporal dimension while the vertical axis encodes the quantitative dimension. Usually, line charts, scatterplots, and area charts are well adapted to plot an individual time series (interested readers may refer to Harries *et al.* [1]). Multiple time series are defined by a set of quantitative variables occurring at the same interval (set of time series). To deal with multiple time series, different visual representations have been defined, such as stacked graphs [1], ThemeRiver [2], [3], and streamgraphs [4].

A hierarchical structure in multiple time series can be expressed as an ordered set of time series, where individual time series are grouped hierarchically according to their proximity. Many datasets can be organized in a hierarchical structure. For example, in music evolution, genres can be the result of the *aggregation* of several sub-genres (e. g., metal genre is the aggregation of black metal, doom metal, alternative metal, and so forth) or the *disaggregation* in different sub-genres (e. g., jazz genre can lead to different sub-genres such as classic jazz, soul jazz, contemporary jazz, and so on). Current visual representations such as stacked graphs [1] or streamgraphs [4] model individual time series but they suffer from scalability problems when the number of time series increases. To overcome this problem, multiple time series can be aggregated into a hierarchical structure to depict the information at different levels of abstraction.

When time series data grow, a technical challenge arises: how to interact and extract valuable information. To help users in this task, there are many well-known interaction techniques that can be adapted to stacked graph based visualization such as: overview+detail [5], focus+context [5], fisheye [6], zooming [7], etc.

In this paper, we propose the design of a new streamgraph-based approach to convey the hierarchical structure of time series. More precisely, our approach combines various interaction techniques to expand the advantages of a streamgraph representation and thus, facilitate the exploration and comparisons of time series. As a result, the end-user is provided with a dynamic tool to explore different levels of abstraction in the hierarchical structure of time series. Fig. 1 illustrates our approach. A demo is available at http://advanse.lirmm.fr/multistream/. The main contributions in our work are:

- **A new streamgraph-based approach** to convey the hierarchical structure of multiple time series, in order to ease the exploration and comparisons of temporal evolution patterns.
- **A multiresolution view** to depict the hierarchical organization of time series at different levels of abstraction (i. e., aggregation/disaggregation of time series).

2 Related Work

Several visualization techniques were proposed to display multiple time series and their evolution over time. The interested readers may refer to Aigner *et al.* [8] which discuss several visualization techniques for time-oriented data. For instance, Interactive Horizon Graphs [9] and Qualizon Graphs [10] use a split-space technique, where horizontal space is divided in order to show each time series in a reduced area. In this paper, we focus on stacked graphs-based visualization and interaction techniques, as well as a hierarchical structure incorporated in the visual representation.

2.1 Stacked Graphs and Streamgraphs Visualization

A widely followed approach to represent temporal variation in multiple time series is using stacked graphs [1]. They feature a

- *Erick Cuenca and Pascal Poncelet are with LIRMM and the University of Montpellier, France. E-mail: firstname.lastname@lirmm.fr.*
- *Arnaud Sallaberry is with LIRMM and the Paul Valéry University of Montpellier, France. E-mail: arnaud.sallaberry@lirmm.fr.*
- *Florence Y. Wang. is with CSIRO, Australia. E-mail: florence.wang@csiro.au.*

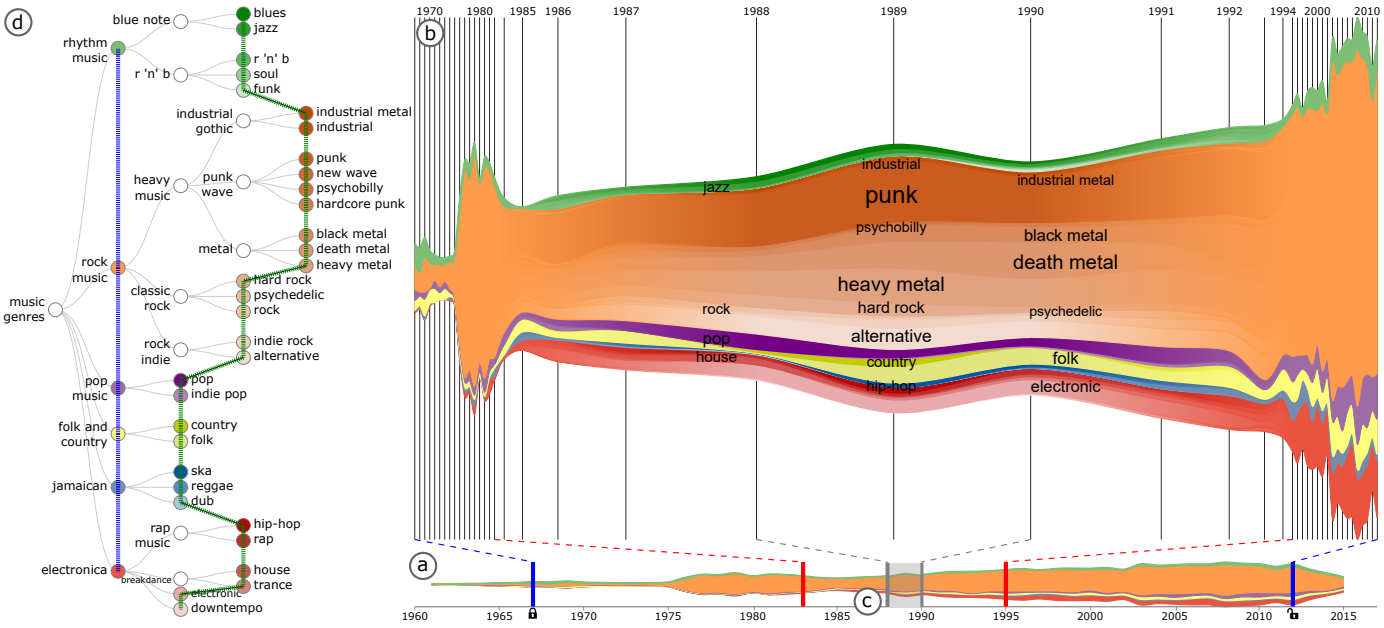

Fig. 1. (a) An overview depicts the time series at a high level of abstraction. (b) Multiresolution view depicts the time series at different levels of abstraction. (c) Controller links the overview and the multiresolution view. (d) Hierarchy manager allows navigating through the hierarchical structure in the time series.

straight baseline (commonly horizontal axis) to stack time series on top of each other. Each time series is visualized as a colored *layer*[1], and thickness represents the value at the given time step. Finally, the direction of the layers from left to right indicates the evolution over time, and the thickness of the aggregated layers reflect the sum of the individual time series. For instance, TIARA [11] adopts a stacked graphs representation to visualize thematic content. NameVoyager [12] uses stacked graphs in addition to interaction techniques to explore trends in a historical dataset of baby names.

Several baseline algorithms are supported in a stacked graphs representation. For example, ThemeRiver [2], [3] is computed in a central baseline parallel to the temporal axis where layers are stacked in a smooth curved transition simulating the flow of a river. OpinionFlow [13] adopts ThemeRiver to detect opinion propagation patterns. EvoRiver [14] and TextFlow [15] merge ThemeRiver with interaction techniques in the document topic analysis field. On the other hand, streamgraphs [4] is computed in a similar central baseline to ThemeRiver, however, it uses a *wiggle* offset that attempts to minimize the weighted change in slope with relation to layer thickness. This difference makes streamgraphs more readable and natural than ThemeRiver flows. Streamgraphs are used in some domains such as body movement [16] or data stream visualization [17].

Like other visualizations, stacked graphs-based representations present scalability issues. Stacking many time series one on top of another increases visual clutter, which makes it harder to compare and interact with the layers. In order to solve this problem, STAC [18] proposes to use multiple coordinated views to analyze individual time series and their aggregation. Some approaches overcome the scalability problem by organizing time series in a hierarchical structure.

1. In this paper, we use the term *layer* to refer a time series in a stacked graphs-based visualization.

2.2 Hierarchical Structure in Time Series

Another technique to address the scalability issue in stacked graphs-based visualization is organizing the time series in a hierarchical structure, where individual time series are grouped according to their proximity. Hence, the aggregated levels should be equal to the sum of the corresponding elements of the group (Fig. 2). The grouping of time series depends on their nature. Hierarchical organization of time series is used in various domains. For instance, BinX [19] analyzes the behavior of currency exchange by grouping them over time. In BookVoyager [20], a stacked graphs shows the history of book sales organized into a hierarchy of categories and subcategories. ManyEyes [21] proposes a "stack graph for categories" to show the categorization of the US historical federal budget into different departments, such as overall defense, atomic weapons, and so forth. In TouchWave [22], a streamgraph shows a listening history of music organized in a hierarchy by genres, artists, and songs. These approaches use the layer ordering and colors to relate individual time series in a group.

More recently, the hierarchical organization has been used in topic-based domains. For instance, LensRiver on NewsLab [23] explores the relationships among the keywords of topics organized in a hierarchical structure. HierarchicalTopics [24] develop a visualization based on ThemeRiver to depict the temporal patterns of topics organized in categories. RoseRiver [25] extends the TextFlow [15] approach to analyze the relationships among topics in an evolutionary hierarchical structure. In order to navigate and explore the hierarchy, interaction techniques are proposed.

2.3 Interaction Techniques

Interaction techniques are used to enhance the effectiveness of a visualization. There are many, such as: *overview+detail* [5], *focus+context* [26], *graphical zooming* [7], *semantic zooming* as in [27], *brushing&linking* [28], etc. Each of these techniques interacts with the data in different ways. For instance, graphical zooming is a technique that consists in showing a specific area

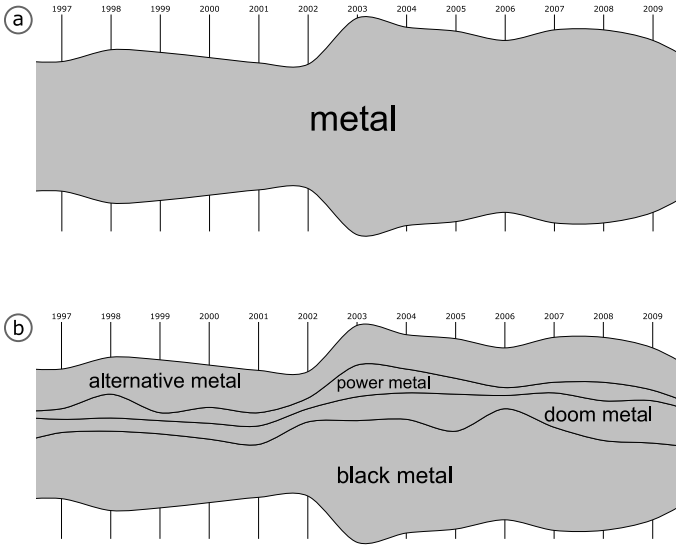

Fig. 2. (a) A streamgraph shows a high level of abstraction in a hierarchy. (b) A streamgraph shows full-details in the hierarchical structure.

in detail. Nevertheless, one of the drawbacks of zoom is that by focusing in, the overview is lost, and when zooming out, the details are no longer available. The focus+context approach overcomes the zooming issue by allowing the user to see both details and overview in one view. Distortion techniques have been proposed to show details in a magnifying area, and the context in a gradually smaller area (e. g., PerspectiveWall [29], fisheye [6], elastic presentation space [30]). Sometimes, this distortion causes confusion in users. The overview+detail technique overcomes this confusion by displaying two separate views; one for the overview and one for the details. Thus, the user can focus on the details without losing the sense of the overall context.

Stacked Graphs-based approaches combine interaction techniques to enhance their interfaces and deal with multiple time series. Table 1 shows a comparison of different interaction techniques. Some of these techniques allow exploration in a hierarchical structure. For example, BookVoyager [20], ManyEyes [21], and HierarchicalTopics [24] integrate a separate view (e. g., tree-control) to allow the user to navigate the hierarchical structure of time series. Tree-control represents the hierarchy so as to filter time series and show them in a stacked graphs. NewsLab [23] uses a structure-based bushing approach [31] to navigate a set of topics in a hierarchical organization. HierarchicalTopics [24] uses a tree-control to explore and merge topics in the hierarchical structure. In contrast, TouchWave [22] proposes a touchable stacked graphs to interact with the hierarchy and help users with a task (e. g., layer ordering, layer filtering, and layout changing).

As shown in Table 1, zooming techniques are widely used to explore details in stacked graphs-based approaches. In a hierarchical structure, zooming allows the user to depict the levels of abstraction. However, when using this technique, only one level of the hierarchy is displayed [20], [21], [22], [24]. TIARA [11] uses a fisheye to display details in a selected topic. In TouchWave [22] a fisheye view is used to focus in on a time segment, but also to show the same level in the hierarchy in both regions (i. e., detail and context). LensRiver on NewsLab [23] allows to change the levels of detail by layer by using a set of interactions. Our work aims at combining different interaction techniques (e. g., overview+detail, zooming, and focus+context) to enhance the analysis of the hierarchical structure of time series and depict the different levels of abstraction.

TABLE 1
Comparison of various stacked graphs-based approaches and interaction techniques to display hierarchical time series.

Approach	Hierarchical time series	Overview +detail	Zoom	Fisheye view
TIARA [11]			x	x
NameVoyager [12]			x	
EvoRiver [14]			x	
TextFlow [15]			x	
STAC [18]			x	
BookVoyager [20]	x		x	
ManyEyes [21]	x		x	
HierarchicalTopics [24]	x		x	
TouchWave [22]	x		x	x
NewsLab [23]	x	x	x	
Our approach	x	x	x	x

3 Requirement Analysis

Time series data is common in many domains such as medicine [32], finance [19], etc. By analyzing time series, interesting questions can be proposed to extract relevant information from the data. For example, in a marketing company, tracking affective information embedded in social media could help monitor people's responses for a campaign. In that case, time series of affective information (i. e., emotion states) can be classified according to an affective model (e. g., Russell's circumplex model, used in [33]). As a result, a hierarchical structure of affective information is formed. In this classification, the *pleasant-activation* group consists of *happy, elated, excited, and alert* emotions. The *unpleasant-deactivation* group consists of *sad, depressed, bored, and lethargic* emotions. A basic question is how to represent this hierarchical organization to enhance the perception of the affective information over time. The user may also be interested in comparing the affective information evolution through the month during which the campaign was launched (e. g., during summer sales). Finally, the user could be interested in displaying the hierarchical structure of this affective information during the campaign period.

In order to effectively answer such questions, we identify the following list of requirements:

[R1] Visualize temporal patterns over the entire time series: This requirement refers to display the evolution of time series over time in order to reveal patterns, trends, peaks, etc. In the previous example, the user can view the distribution of the affective information at a high level of abstraction. Thus, the user is able to detect where time series conveys temporal patterns.

[R2] Select a time segment of interest: This requirement refers to enabling the selection of a time segment for further analysis. In the previous example, the user can focus in on the month of May (beginning of the summer sales) to compare the distribution of affective information structure (e. g., happy, elated, excited, alert, etc.).

[R3] Handling selected time segments at different levels of abstraction: This requirement refers to displaying the hierarchical structure of time series at different levels of detail. In the previous example, the user selects the month of May to depict the distribution of the affective information in a full-detail level (e. g., happy, elated, excited, alert, etc.). In order to maintain the cognitive map, preceding and subsequent periods are displayed at a high level of abstraction (e. g., pleasant-activation and unpleasant-deactivation levels).

The above list of requirements attempts to meet the common and specialized tasks in the visual analysis of time series datasets. Since these datasets are present in many domains, the intended users are vast. Our intention is to allow the use of our approach by the general public, as well as specialized users.

4 VISUAL MAPPINGS AND FUNCTIONALITY

This section describes our visualization and interaction techniques. Our exploration process follows the principle of the *visual information seeking mantra* proposed by Shneiderman: "overview first, zoom and filter, then details on demand" [34].

4.1 Design Rationale Summary

Based on the requirement analysis, we propose a new visual approach to facilitate the analysis of hierarchical time series. It is based on four interactive components: overview, multiresolution view, controller, and hierarchy manager. The *overview* (Fig. 1a) shows time series in a high level of abstraction in order to facilitate recognizing occurrences and evolution of patterns over time **[R1]**. The *multiresolution view* (Fig. 1b) conveys time series at different levels of detail **[R3]**. The *controller* (Fig. 1c) allows to select an interesting time segment (e. g., when a peak occurs) and update the multiresolution view **[R2][R3]**. Finally, the *hierarchy manager* (Fig. 1d) allows to explore and navigate through the different levels of the hierarchy over a tree layout **[R3]**. These four visual elements are described in detail below.

4.2 Overview

As described in related work, stacked graphs representation is well adapted to depict time series and reveal patterns over time **[R1]**. We adopt a streamgraph approach to show more readable time series using a river flow metaphor (Fig. 3). We represent multiple time series as colored layers in a 2D Cartesian coordinates system, where the time dimension is encoded in a discrete scale along the horizontal axis and quantitative dimension is encoded along the vertical axis.

Fig. 3 shows a streamgraph of the entire multiple time series at a high level of abstraction. The highest level represents the top at the hierarchy and the thickness of a layer at this level conveys the sum of time series in the group at each given time step. The overview conveys temporal patterns and distinguishes their evolution over time **[R1]**. In order to explore and display the different levels of the hierarchy **[R3]**, we design a multiresolution view.

4.3 Multiresolution View

The multiresolution view is based on a *focus+context* approach to show a part of the data in detail. This view depicts time series on different levels of granularity **[R3]**. Our goal is to show the top (Fig. 2a) and the lowest level (Fig. 2b) of the hierarchy in one view. For this purpose, we propose a view formed by areas where the granularity of time series varies.

Fig. 4 shows the multiresolution view representing the example of the evolution of musical genres presented in the introductory section. In the multiresolution view, we define three types of areas: context-area, detailed-area, and transition-area. The ***context-area*** (Fig. 4a) is the interval of time where time series are displayed at a high level of abstraction. In this case, metal and jazz categories are plotted using two different hues. The ***detailed-area*** (Fig. 4c) represents the interval of time to display the lowest level of the hierarchy. In this case, the sub-genres of the metal and jazz categories (e. g., black metal, doom metal, classic jazz, contemporary jazz, and so forth). In the detailed-area, the time series related to a group are colored with the same category family hue but with a different level of saturation. Finally, the ***transition-area*** (Fig. 4b) represents the time segment dedicated to the transition between these two levels of abstraction. In this area, a color interpolation is applied to preserve the context of this transition.

The time intervals in the multiresolution view are distorted to depict different levels of details over time (see horizontal axis in Fig. 4). This distortion is related to the focus+context approach where details are shown in a magnifying area and the context in a smaller area. Thus, the length of the time steps in the detailed-area (Fig. 4c) are larger than the length of the time steps in the context-area (Fig. 4a) and the transition-area (Fig. 4b). As our visualization uses a 2D Cartesian coordinates system, the transition-areas (Fig. 4b) implement a Cartesian distortion function to provide a smoothing effect for aggregation/disaggregation from the context-areas to the detailed-area, and vice versa (the distortion function is described in Sec. 5)

Vertical scaling in the multiresolution view (see vertical axis in Fig. 4) depicts the hierarchical structure, where categorical colors are used to discretize time series in the hierarchical organization. By giving each layer a separate hue, we convey a distinction between them and decrease ambiguity. Various approaches use the order of layers in a streamgraph to enhance the effectiveness of the visualization. For example, in NameVoyager [12] the ordering of the layers is alphabetical because they emphasize the visualization by name. In our approach, the order of layers provides a way of representing the grouping of time series. In Fig. 4, the layers of sub-genres are plotted together according to their parent category. The vertical axis is not distorted in order to avoid loosing the context of the overall distribution, which would make the comparison difficult.

Contextual information is always necessary to enhance the expressiveness of a visualization. Sometimes, the legibility problem linked to the number of layers in a streamgraph makes it impossible to set a name for all of the layers. Thus, our approach uses a brute-force labeling layers algorithm [4] to find the best place to plot the label and avoid overlap. The font size of the labels encodes the data quantity represented. Thus, the bigger the font size, the bigger the value will be at that time step. Layer labeling is available in the detailed-area to convey at a glance which layer has the highest value in this interval of time (Fig. 4c).

The distribution of areas in the multiresolution view allows the user to display the different levels of granularity in a hierarchy. As a result, the user observes the highest and lowest levels of the hierarchy in a single view. In order to handle the intervals of time of the detailed-area, context-areas, and transition-areas, we designed a *controller* that allows customizing them according to user requirements.

4.4 Controller

Since multiple time series datasets are often large, filtering out irrelevant data helps users to focus on interesting data. Based on the brushing&linking technique [28], we designed a *controller* to interact with the overview along the horizontal axis (i. e., time dimension) and focusing on interesting data according to users requirements **[R2]**. This movable tool is designed over the overview

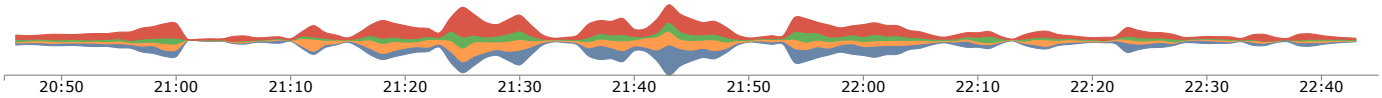

Fig. 3. An overview depicting the top level of the hierarchy to conveys temporal patterns over time.

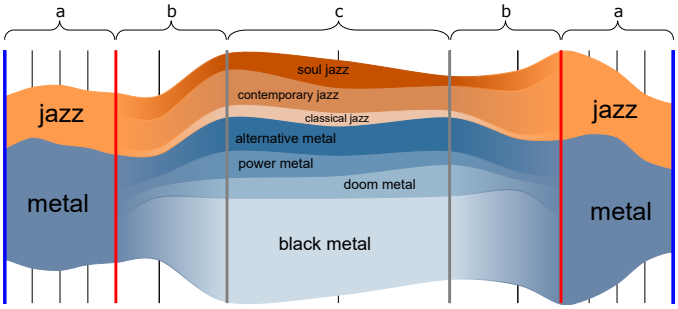

Fig. 4. Multiresolution view. (a) context-areas depict the top level of the hierarchy, (c) detailed-area depicts the lowest level of the hierarchy, and (b) transition-areas depict the transition between the context-area to detailed-area, and vice versa.

to handle the intervals of time used by areas in the multiresolution view (i.e, detailed-area, context-areas, and transition-areas).

Fig. 5 shows the design of the controller. Vertical interactive lines control the position and extension of the areas. Context-areas are handled by the blue lines (Fig. 5a). Transition-areas are handled by the red lines (Fig. 5b). Finally, the grey lines handle the detailed-area (Fig. 5c). All of the lines in the controller can be expanded and collapsed interactively at regular intervals of time (constraints are described in Sec. 5).

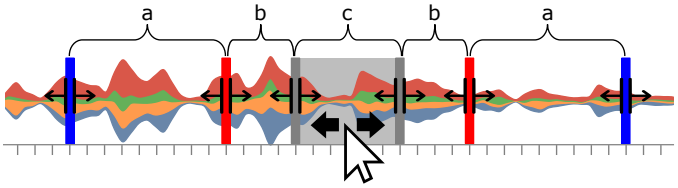

Fig. 5. Controller design. (a) context-areas, (b) transition-areas, and (c) detailed-ar ea are handled by colored lines that can be expanded and collapsed interactively. Detailed-area remains colored in grey reminding the viewer the focus point.

As the lines guide the position of the areas, we manage the interactivity between them. When dragging the mouse over the detailed-area, all other areas dynamically update, keeping their length. For instance, Fig. 6a shows the detailed-area dragged to the left, and the other areas updated in their new position after moving left. In Fig. 6b the detailed-area is expanded to the right by dragging the right grey vertical line. It updates the length of the adjacent areas (transition and context areas) in the same direction. When a transition-area or a context-area is expanded or collapsed, only the length of that area is updated. Fig. 6c shows a transition-area expanded to the right. Note that all other areas remain static. The position of context-areas can be locked by clicking on the padlock icon below the context lines. Fig. 6d shows the two context-areas locked, and the detailed-area dragged to the right. Notice that when the detailed-area is dragged, the length of transition-areas is updated, but not the length of the context-areas. With the controller, the user can view the time segments at different levels of abstraction **[R2][R3]**. Fig. 7 shows the projection of areas selected by the controller over the multiresolution view. Dragging any area of the controller updates the multiresolution view. This animation helps the user to conserve a mental map of the views (i. e., overview and multiresolution view).

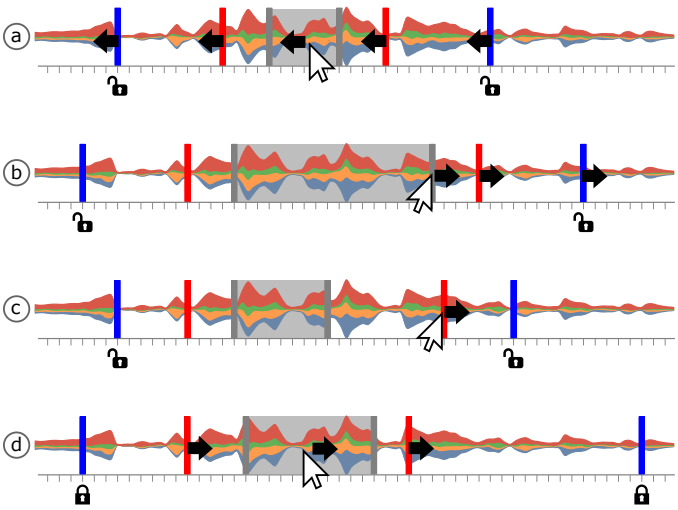

Fig. 6. Using the controller. (a) Detailed-area is dragged to the left. (b) Detailed-area is expanded to the right. (c) Right transition-area is expanded to the right. (d) The position of the context-areas are locked by clicking the padlock icon.

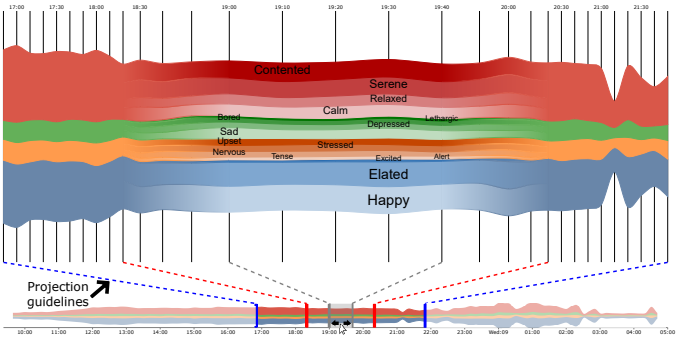

Fig. 7. At the bottom, the controller shows the areas displayed in the multiresolution view. At the top, the multiresolution view, result of the controller projection. Colored projection guidelines help the viewer to follow these projections.

Our approach allows changing the layout baseline. The layout by default is a streamgraph with a central baseline but a user can transition smoothly to a stacked graphs layout to compare layers (Fig. 17a).

In a streamgraph, it is often difficult to see layers with smaller values among layers with much larger values. We implemented a *vertical ruler* to overcome this problem and to visualize the exact value encoded for a given time series at a given time step (Fig. 8). While the mouse hovers over a layer, the saturation changes and the vertical ruler appears, producing a *tooltip box*

with the temporal and quantitative information at that time step. Clicking on this tooltip box will bring up a panel, showing more detailed information about the raw data.

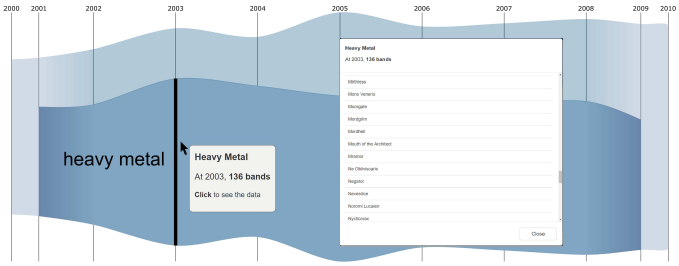

Fig. 8. Tooltip-box shows the value at every time step in the streamgraph.

4.5 Hierarchy Manager

Two levels of the hierarchical structure are displayed on the Multiresolution view: the *context-areas* show a high level of abstraction, *i.e.* streams from a high level of the hierarchy, while the *focus-area* shows details, *i.e.* streams from a low level of the hierarchy. When the depth of the hierarchy is 2, then the two levels are displayed. However, most hierarchical structures contains more than 2 levels. To deal with deeper hierarchies, and thus better fit **[R3]**, we propose a *hierarchy manager*.

Fig. 9 shows the musical genres classification represented in the hierarchy manager. In this view, we adopt a rooted tree layout as such representation allows grouping time series and facilitates the exploration of the hierarchy. This tree is horizontally oriented, where the first node on the left represents the root of the hierarchy. Every leaf node denotes a time series and the non-leaf nodes express the aggregation of time series. Label information is placed to the right of the leaf nodes and to the left for the other nodes. The hierarchy manager is coordinated with the multiresolution view. Therefore, the order of nodes in the tree are the same that the layers in the multiresolution view. In addition, nodes are filled by the same color coding than the layers in the multiresolution view (see Fig. 1).

The hierarchy manager acts as a controller in the navigation through different levels of the hierarchy in the multiresolution view. Vertical dotted lines are implemented to depict the current levels of the context-areas and detailed-area. In Fig. 9, a blue vertical dotted line crosses the nodes that are depicted in the context-areas (i. e., high level of abstraction). Similarly, a green vertical dotted line crosses the nodes that are shown in the detailed-area (i. e., low level of abstraction). Following this configuration, the vertical green line will always be at least one level lower than the vertical blue line in the hierarchy. Fig. 9 shows a hierarchy structure with many levels. In order to provide an optimal initial view of the hierarchy organization, our approach initially shows the first level nodes of the hierarchy in the context-areas, and the lowest levels nodes of the hierarchy in the detailed-area.

Aggregation and disaggregation in the hierarchical structure. For detailed analysis, the hierarchy manager aims to allow users a flexible navigation through the hierarchy (e. g. aggregation/disaggregation). Disaggregation can be performed in nodes that have children and aggregation can be achieved in nodes that have a parent. A tooltip is implemented to provide these two features. This tooltip is shown when the user moves the mouse over a node that is crossed by one of the vertical dotted lines. A

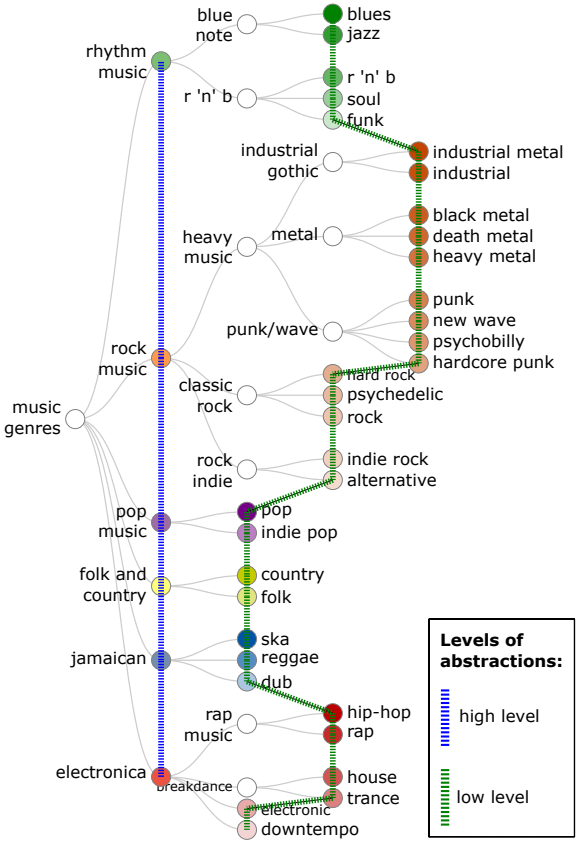

Fig. 9. Hierarchy manager. A time series hierarchy is represented with a tree layout, where vertical dotted lines convey the levels of abstractions shown on the multiresolution view. Precisely, a blue vertical dotted line crosses the layers depicted in the context-areas (i. e., high level of abstraction), a green vertical dotted line crosses the layers depicted in the detailed-area (i. e., low level of abstraction).

right arrow button is shown to perform a disaggregation and a left arrow button to perform an aggregation. For example, Fig. 10a shows the disaggregation of the heavy music genre. This genre is crossed by the dotted blue line (i. e. layer is plotted in context-areas). After the disaggregation, the dotted blue line now crosses the three children: industrial gothic, metal, and punk/wave. The context-areas of the multiresolution view are also updated to show these three genres. Fig. 10b shows the aggregation interaction. In this example, genres such as black metal, death metal, and heavy metal are aggregated into the metal genre. After the aggregation, the dotted green line crosses the new aggregated genre. The detailed-area of the multiresolution view is also updated showing the metal genre.

Highlighting nodes. The hierarchy manager provides some useful interactions that enhance navigation tasks. Users can highlight a node by moving the mouse over it. If the highlighted node is crossed by the dotted green line (i. e., layers are plotted in the detailed-area) then only this layer is highlighted in the Multiresolution view, as shown in Fig. 11a. If the highlighted node is crossed by the dotted blue line (i. e., layers are plotted in the context-areas) then all the children that are crossed by the dotted green line are also highlighted, as shown in Fig. 11b. These animations are intended to assist users in the navigation task by coordinately highlighting the hierarchical relationships in the hierarchy manager and the multiresolution view.

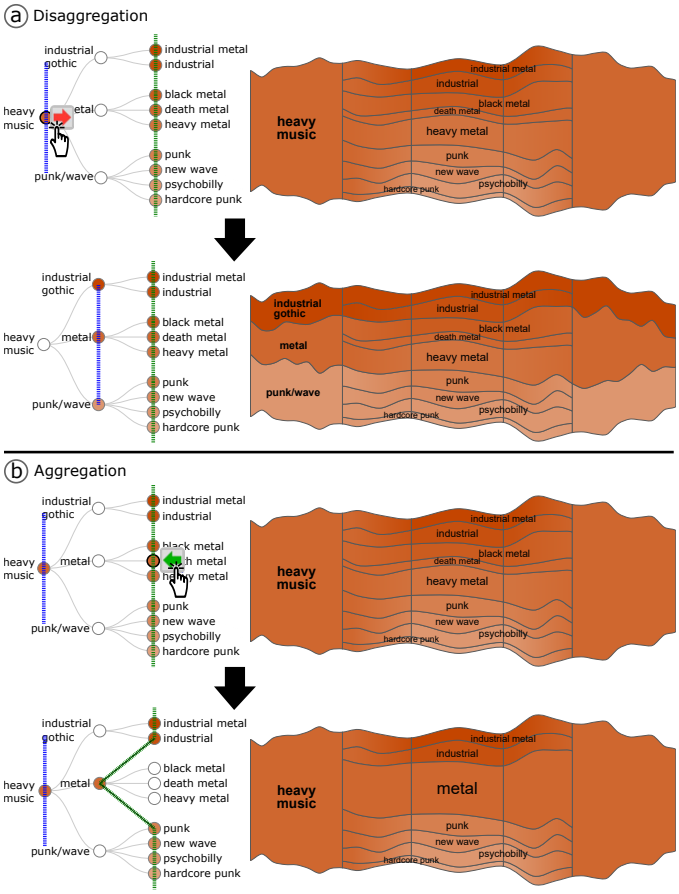

Fig. 10. Supported interactions to explore different levels of abstraction in the hierarchy. (a) Disaggregation: split the selected node into children nodes. (b) Aggregation: group children nodes in their parent node.

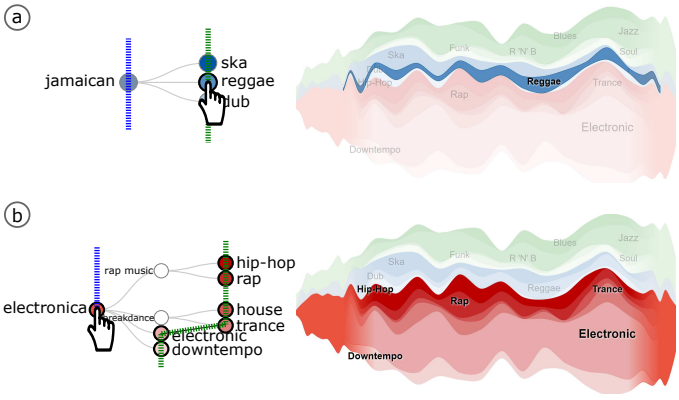

Fig. 11. Highlighting nodes. (a) As the selected *reggae* node is depicted in the detailed-area (i. e. crossed by the dotted green line), then only this node is highlighted. (b) As the selected *electronica* node is shown in the context-areas (i. e., crossed by the dotted blue line), thus all the children that are shown in the detailed-area are highlighted too.

Filtering nodes. To help users in the exploration task, the hierarchy manager allows to filter time series. This feature is performed by clicking on a node that is crossed by one of the dotted vertical lines. Fig. 12a shows the filter of the electronic genre node that is depicted in the detailed-area (i. e., crossed by the dotted green line). After node filtering, layers in the multiresolution view are updated. Fig. 12b shows the filtering of a node that is represented in the context-areas (i. e., crossed by the dotted blue line). In this case, as this node has children depicted in the detailed-area (i. e., crossed by the dotted green line), all the children are also filtered. All these interactions smoothly update the Multiresolution view.

In summary, the hierarchy manager presents the hierarchical structure in a visual tree representation and combines it to user interactions to enhance the exploration and navigation task.

5 Technical Requirements for the Multiresolution View

In this section, we explain the projection of the controller over the multiresolution view, as well as the constraints that the application must handle in this projection. We also discuss color transitions.

5.1 Time step lengths

The controller and the multiresolution view are based on an overview+detail technique. The areas selected by the controller are projected over the multiresolution view, creating a cognitive link between them. Fig. 13 shows the variables involved in this design, where the time steps are colored according to the area that they represent: blue for the context-areas, red for the transition-areas, and grey for the detailed-area.

With the controller (at the bottom of Fig. 13), the number of time steps is defined by: c_i for context-areas, t_i for transition-areas, and d for detailed-area; where $i \in \{1,2\}$. Since these areas are projected over the multiresolution view, we must know the total number of the time steps. We write this sum as N, where

$$N = c_1 + t_1 + d + t_2 + c_2.$$

The next step is to manage the projection of the controller areas over the multiresolution view (at the top of Fig. 13). The horizontal space available on the screen is represented by S. This view is based on a focus+context technique. In order to obtain time steps in the detailed-area larger than the time steps in the other areas, we use *projection factors* (Fig. 13). The detailed-area is managed by the α factor, transition-areas are managed by β, and context-areas are managed by γ, where $\{\alpha,\beta,\gamma\} \in \mathbb{N}$. The constraint $\alpha > \beta > \gamma$ is imposed over these factors to preserve the relation between the time steps of these areas. Without this restriction, the length of time steps in context-areas could be larger than the length of time steps in the detailed-area, losing the sense of the focus+context technique.

In order to calculate the relative horizontal length for each area in the multiresolution view, we divide the number of time steps of an area by the total number of the time steps (N) and multiply it by the corresponding projection factor. Therefore, $RSC_i = \gamma \cdot \frac{c_i}{N}$ is the relative length for the context-areas, $RST_i = \beta \cdot \frac{t_i}{N}$ is the relative length for the transition-areas, and $RSD = \alpha \cdot \frac{d}{N}$ is the relative length for the detailed-area; where $i \in \{1,2\}$. The total sum of these relative lengths is represented by RS, where

$$RS = RSC_1 + RST_1 + RSD + RST_2 + RSC_2.$$

The horizontal length of each area in the multiresolution view is obtained using the relative horizontal length and the horizontal space available (S). Thus, the context-areas length is $SC_i = S \cdot \frac{RSC_i}{RS}$, the transition-areas length is $ST_i = S \cdot \frac{RST_i}{RS}$, and the detailed-area length is $SD = S \cdot \frac{RSD}{RS}$.

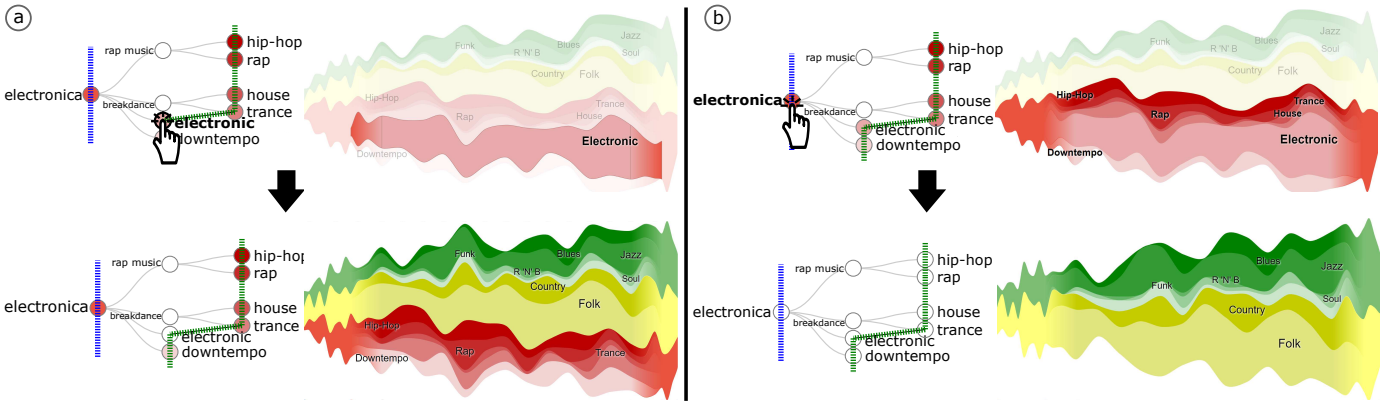

Fig. 12. Filtering nodes. (a) As *electronic* genre node is depicted in the detailed-area (i. e., crossed by dotted green line) then only this layer is filtered. (b) As *electronica* genre layer is depicted in the context-areas (i. e., crossed by dotted blue line) then all their children are also filtered.

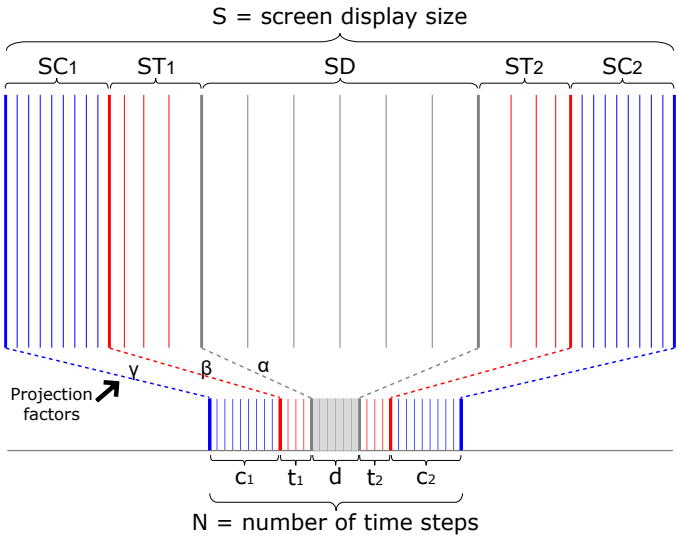

Fig. 13. Design of the projection of the controller areas (at the bottom) over the multiresolution view (at the top).

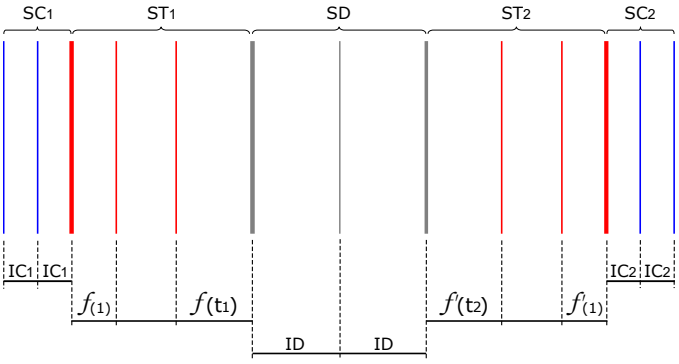

Fig. 14. Horizontal distribution of time steps in the Multiresolution view.

The length of the time steps is obtained by using these horizontal area lengths. Fig. 14 shows the time step distribution by area of the multiresolution view. Each time step is represented by a region on the horizontal axis. However, the length of a time step depends on the length of the area and the number of time steps to plot in that area. Thus, context-areas (SC_i) and the detailed-area (SD) have uniform time step lengths. Time step lengths in a context-area are $IC_i = \frac{SC_i}{c_i}$ and $ID = \frac{SD}{d}$ in the detailed-area. The length of time steps in transition-areas (ST_i) are distorted because they represent the transition between two different levels of granularity (e. g., context-area to detailed-area). Therefore, they are not uniform, and an exponential function is used to depict this behavior.

Let $T = \{1, \ldots, t_i\}$ be the set of time steps in a transition-area. The length of each time step $x \in T$ in that area is given by the function

$$f(x) = a \cdot b^x \tag{1}$$

where, a is a constant representing the minimum possible value of the function, and b the base of the function. The *base* must always be higher than 1 to ensure a strictly increasing function (i. e., $f(x+1) > f(x)$). In order for the distortion time steps be continuous and logical, this function must satisfy two requirements:

[r1] $f(1)$ must be greater than IC_i.
[r2] $f(t_i)$ must be smaller than ID.

Assuming $a = IC_i$ and $b > 1$ to accomplish **[r1]**, we must find the base (b) satisfying **[r2]** to achieve the desired exponential growth.

We use a real polynomial function of degree t_i

$$ST_i = a \cdot b^1 + a \cdot b^2 + \ldots + a \cdot b^{t_i} \tag{2}$$

where ST_i is the length of a detailed-area.

In our visualization while the areas are dragged in the control tool (at the bottom of Fig. 13), the base (b) in Eq. 2 is calculated for every transition-area. We use the RPOLY algorithm [35], then we apply the following rules to update the multiresolution view:

If $b <= 1$ (**[r1]** not respected) $\Rightarrow$ we show an alert message and do not update.
Else If $f(t_i) >= ID$ (**[r2]** not respected) $\Rightarrow$ we show an alert message and do not update.
Else we update areas in multiresolution view with f values.

This interaction gives the user the autonomy to expand and collapse the controller while respecting **[r1]** and **[r2]**. For example, Fig. 15 shows three multiresolution views. In the first image, the transition areas (b) respect **[r1]** and **[r2]**. As a result, we

obtain a view where the relation between the time steps of all the areas is correct. However, the middle image of Fig. 15 the transition-areas (b) do not respect **[r2]**. As a result, the time steps in transition-areas (b) are larger than the time steps in the detailed-area (c). In this example, the multiresolution view no longer shows great resolution of the detailed-area. The third image of Fig. 15 shows the importance of using the transition-areas in our approach. Notice how the absence of a transition-area on the left of the detailed-area (c) results in a sudden change between the time steps of the context-area (a) and the detailed-area (c). This absence might hamper the user's perception of changes between different levels of granularity. On the other hand, the right of the detailed-area (c) shows a transition-area (b) where the length of the time steps decrease from these two areas, thus improving the user's perception of the changes between levels of granularity.

In order to control the projection factors (i. e., α, β, and γ) a menu is implemented at the bottom of the user interface. This menu shows an alert message if $\alpha < \beta$ or $\beta < \gamma$.

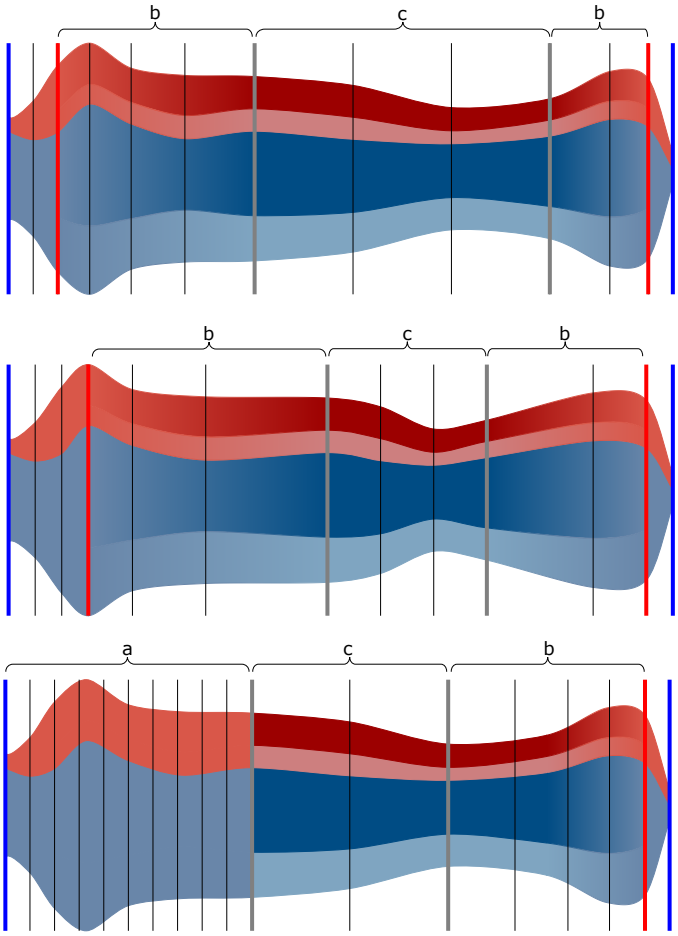

Fig. 15. In the first image, a multiresolution view correctly depicts the transition between a context-area (b) to detailed-area (c) and vice versa. In the middle image, transition-areas (b) where the constraint is overlooked. The third image shows the importance of using transition-area where the transition-area to the left of the detailed-area (c) is not shown.

5.2 Colors

Transition-areas represent the interval of time between two levels of abstraction (e. g., context-area to detailed-area). Thus, for a smooth transition between these two levels, a color interpolation is necessary.

Fig. 16 shows the multiresolution view. The left transition-area (see b in the left of Fig. 16) depicts the passage between context-area (a) to detailed-area (c) in increasing time intervals. However, the sudden color shifts between these two areas makes it difficult to follow the transition. To overcome this problem, we use an interpolation of colors along the transition area. The right transition-area (see b in the right of Fig. 16) shows an interpolation of color between the detailed-area (c) to the context-area (a). Distinguishing the transition between different levels of granularity (i. e., categories and subcategories) helps the user.

Color coding choice is an issue when dealing with deep hierarchies and multiple time series. To handle this scalability problem, our approach adopts a set of the 20 categorical colors available in the D3.js[2] library. Each layer of a high level of abstraction uses a specific hue (see a in Fig. 16), where hues between neighboring layers must be different to avoid ambiguities. The layers of the children of a low level of abstraction (see c in Fig. 16) maintain the hue of their parent while varying the saturation.

Fig. 16 illustrates also three additional features. The first option addresses the problem of readability when the number of layers increases in the detailed-area (c) so it becomes hard to distinguish the layers with less saturated colors. To overcome this issue, the contour of layers are painted black, thus reinforcing the link between the parent layer with their children. This feature is enabled by clicking the "outline-layers" checkbox. The second option allows to show the borders of the projected areas in the multiresolution view. It is activated by clicking the "border areas" checkbox. This option improves the perceptual relationship between the overview, controller, and the multiresolution view. The third option is enabled by clicking the "highlight detailed-area" checkbox. This option enhances the transition between the areas highlighting the colors of layers in the detailed-area more than the layers in the context-areas.

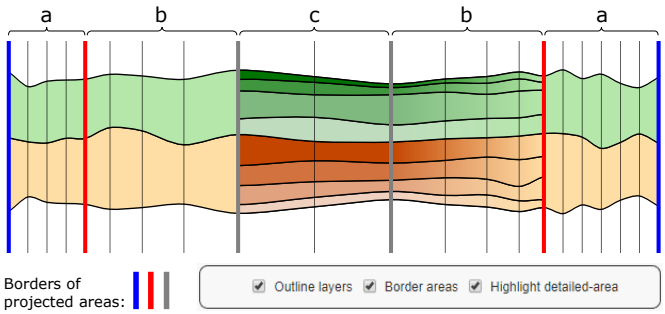

Fig. 16. The importance of an interpolation of colors in transition-areas (b) on the Multiresolution view.

6 Discussion

In this section, we discuss the flexibility of our approach and compare it with alternative techniques that support a hierarchical organization.

6.1 A flexible approach

The controller manages the navigation in our approach. The user can accomplish various interaction techniques, such as overview+detail or fisheye. These interaction techniques overcome the

2. https://d3js.org/ [last access December 01, 2017]

legibility and comparison issues of a streamgraph. For instance, in Fig. 17b, a simple overview+detail is performed by expanding the detailed-area (grey) and collapsing the other areas (i. e., context and transition areas). As a result, a selected area on the overview is projected in full-detail over the multiresolution view **[R2]**.

The controller allows the user to handle the length of the detailed-area, which can be an interval of time or the entire dataset **[R1]**. Fig. 17c shows a fisheye distortion, where the position of the context-areas (blue lines) is locked at the edges. As a result, context-areas are compressed and the detailed-area (grey) is enlarged; giving the effect of a classic fisheye. When dragging the controller, the position of the detailed and transition areas shifts, while the boundaries of the context areas remain locked **[R2][R3]**.

Fig. 17d shows a combination between overview+details and fisheye, where the intervals in the context-area are more extended than the detailed-area. Using the controller the user can customize the time intervals shown at the highest and the lowest levels of granularity in the multiresolution view **[R3]**.

6.2 Comparison with alternatives techniques

We compare our approach with the streamgraph approaches that support a hierarchical organization, as described in the related work section. Mainly, we focus on the task of the visualization of large time series, navigation through the hierarchical structure, and depicted levels of the hierarchical organization.

Large time series visualization. Visualizing large datasets is a challenge due to the limitation of display screen size. Some previous approaches suffer from legibility problems when dealing with large time series. For instance, HierarchicalTopics [24] explores the hierarchy of a topic in different view panels, leading to a visual clutter when the number of sub-topics increases. In other approaches, such as BookVoyager [20], ManyEyes [21], and NewsLab [23] the length of time series is fixed to the size of the display screen (see Fig. 18). As a result, in large datasets, it is difficult to compare time series or find recurrent patterns over time **[R1]**. To overcome this issues, TouchWave [22] allows performing a focus+context technique (e. g., fisheye distortion) to show details in a large area, and the context in a smaller area. However, TouchWave also fixed the ends of the streamgraph at the screen display borders. Our work offers more flexibility than TouchWave and previous techniques, using the *controller*, users can select the ends of the context areas and also limit the focused area which is reflected in the multiresolution view **[R2][R3]**. This behavior allows to easily reach several interaction techniques such as zoom, focus+context, and fisheye distortion. This feature is useful when a user focuses on a precise time segment, while maintaining a flexible context of historical data.

Hierarchical structure. To navigate through the hierarchical structure, previous approaches propose various solutions. NewsLab [23] and TouchWave [22] use interaction techniques such a drill-down and roll-up actions that allow to depth in details in a selected leaf of the hierarchy. Nevertheless, with such a navigation it is difficult to compare the relationships between layers on the streamgraph since the current levels of the hierarchy are not depicted. Therefore, the user can get lost during exploration tasks. In order to overcome this issue, BookVoyager [20], ManyEyes [21], and HierarchicalTopics [24] use a visual tree representation to navigate into the hierarchical structure. Our approach extends this tree representation to the *hierarchy manager* by adding a set of user interactions **[R3]**. In this manner, our work enhances the drawbacks of TouchWave and NewsLab where only the current analyzed level in the hierarchy is shown. In addition, our hierarchy manager allows modifying the granularity of displayed time series, with aggregation/disaggregation, highlight, and filter **[R2][R3]**. These interactions aim at enhancing the navigation and exploration tasks in the hierarchy.

Depicted levels of the hierarchy. In previous approaches such as TouchWave [22] and NewsLab [23], the user can only visualize one level of abstraction of time series at the same time. Our work differs from these approaches since the *multiresolution view* allows two different levels of the hierarchy to be visualized at the same time: context (e. g., high level of the hierarchy) and focus (e. g., low level of the hierarchy) **[R3]**. Using the hierarchy manager and the controller, users can explore/navigate through the hierarchy at a period of time for a depth analysis **[R1][R2][R3]**. Thus, it shows how parents and children evolve over time in the same view.

7 EXAMPLES

This section presents two examples of applications of our approach. The first one shows the evolution of sentiments expressed in tweets on the US 2016 presidential election day. The second one shows the evolution of music genres from 1960 to 2016.

7.1 2016 US presidential election day

Our first example is based on tweets collected on the 2016 US presidential election day (8-9 November, 2016 UTC). This dataset contains the sentiments analyzed from 371584 tweets with the hashtag #Hillary or #Trump. The sentiments are classified according to the Russell's affective model [36] as described in [33]. This model forms a hierarchical structure of sentiments. In this organization, the pleasant-activation group consists of happy, elated, excited, and alert emotions. The activation-unpleasant group consist of tense, nervous, stressed, and upset emotions. The unpleasant-deactivation group consists of sad, depressed, bored, and lethargic emotions, and the deactivation-pleasant consists of calm, relaxed, serene, and contented emotions.

Fig. 19 shows our approach, where each layer in the streamgraph represents a sentiment. Color coding is used to distinguish the 4 main sentiment categories. The pleasant-activation group is depicted by blue, the activation-unpleasant group is depicted in orange, the unpleasant-deactivation group is depicted in green, and the deactivation-pleasant group is depicted in red. In Fig. 19, the overview (at the bottom) shows the evolution of sentiments at a high level of abstraction, displaying the main sentiments. The multiresolution view (on the right) and the hierarchy manager (on the left) are linked to depict the sentiments at different levels of details (main and disaggregated categories). In this example, the hierarchy contains 2 levels, where the nodes crossed by the dotted blue line are depicted in the context-areas and the nodes crossed by the dotted green line are shown in the detailed-area.

The overview in Fig. 19 shows some interesting peaks at the end of the election day. The multiresolution view depicts details from 23:00 to 06:00. We observe that until 23:00, all layers evolve homogeneously. Then, the first results are broadcast, making the elated sentiment layer grows (Fig. 19a). Between midnight and 02:00, when most of the polls were closed, the volume of elated sentiment remains constant. Note how in this period, the label height of the elated sentiment is bigger than the others. This conveys at a glance, the dominance of this sentiment. The peak

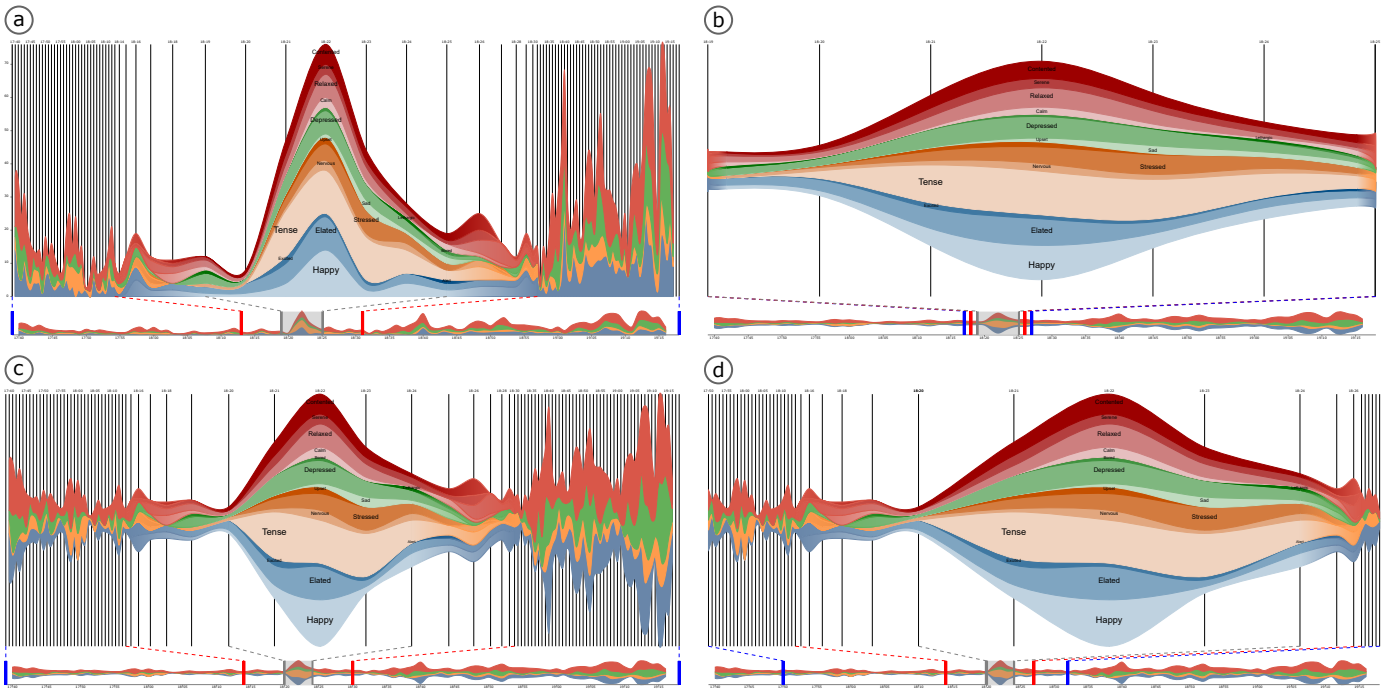

Fig. 17. Interaction techniques for hierarchical time series over a streamgraph. (a) Changing the baseline in a streamgraph (stacked graphs or streamgraph). (b) Overview+detail technique. (c) Fisheye distortion technique. (d) Combination between an overview+detail and a fisheye distortion.

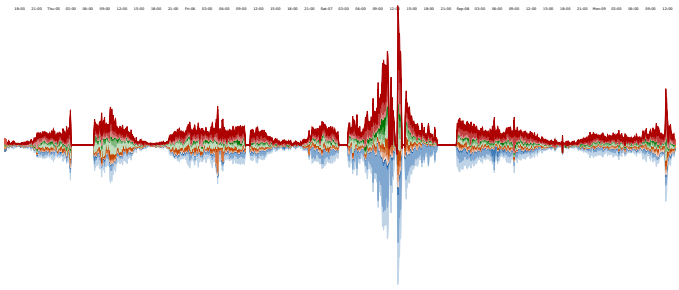

Fig. 18. A streamgraph of large time series limited by the screen display size.

in Fig. 19b shows the moment when the results of the state of Florida were broadcast. Another peak at 04:30 shows the moment when the results of the California were broadcast (Fig. 19c).

The multiresolution view allows the user to focus on an area while keeping a customizable context. For example, while we focus on the peaks between midnight and 05:00, we can see that the highest peak of the entire dataset is at 09:00, when Donald Trump was declared the winner of the election.

7.2 Music genre evolution

This example is based on music information extracted from the MusicBrainz[3] web site. The dataset contains the metadata information of about 10642 bands from 1960 to 2016. We used this information to design our own genre taxonomy. This taxonomy can be organized in a hierarchical structure. Thus, the first level is composed of 6 main categories (e. g., rhythm music, rock music, pop music, folk and country, jamaican, and electronica), and the lowest level is composed by 32 subcategories (e. g., blues, jazz, r 'n' b, soul, funk, industrial metal, industrial, black metal, death metal, heavy metal, punk, new wave, psychobilly, hardcore punk, hard rock, psychedelic, rock, indie rock, alternative, pop, indie pop, country, folk, ska, reggae, dub, hip-hop, rap, house, trance, electronic, and downtempo). This taxonomy aims at helping the viewer to navigate the history of music with different levels of granularity (i. e., genre and sub-genres).

Fig. 20 illustrates the proposed visualization. Each layer in the streamgraph represents a genre, and their horizontal length depicts their longevity over time. At the bottom, an overview shows the entire dataset at a high level of abstraction. The hierarchy manager (on the left) and the multiresolution view (on the right) are coordinated to depict the genre's layers at different levels of details. Categorical color coding is used to convey the main genre's categories. Thus, the *rhythm music* genre is shown in green, *rock music* genre is shown in orange, *pop music* is shown in purple, *folk and country* genre is shown in yellow, *jamaican* genre is shown in blue, and *electronic* genre is shown in red. In this example, the depth of the hierarchy varies according to the level of details in each musical genre. The thickness of a genre's layer depicts the number of bands created per year. For example on the multiresolution view, the vertical ruler shows that 167 new bands of rock music genre were created in the year 2000.

The overview in Fig. 20 depicts an interesting peak of the rock genre (orange hue) from the mid-1970s to the mid-1980s. In order to explore the details of the genre composition in this period, we use the controller to focus on this period of time and the hierarchy manager to navigate through the levels of the hierarchy. As result, the multiresolution view shows the projection of the controller defined areas. We depict the 1970s and 1990s period in full detail (i. e., genres crossed by the green vertical line). In order to not lose the context of this period, we have taken into account periods of time before and after. We observe that the peak in the genre of rock is due to a peak in the heavy music genre in the 70s. We use the

3. MusicBrainz is an open music encyclopedia that collects music metadata. https://musicbrainz.org/ [last access December 01, 2017]

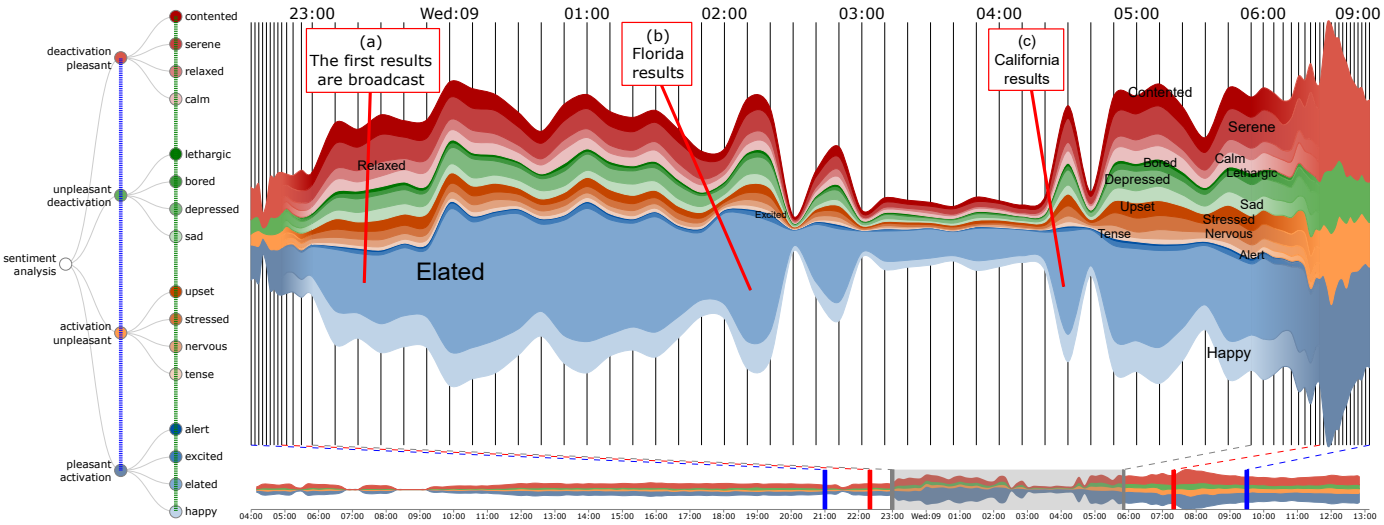

Fig. 19. Evolution of sentiments expressed in tweets on the US 2016 presidential election day: results.

hierarchy manager to display more details on this genre. Fig. 21 explores the rock music sub-genres. We observe that the peak in the heavy music genre is due to a peak in the punk/wave genre in the 70s. However, in the early 80s, the punk/wave genre was low in popularity, making the heavy and rock genre unpopular equally. After 1985 other sub-genres such as black metal, death metal, heavy metal, and indie rock gain ground, making the rock music genre regain the level of popularity of the 70's. Notice as in Fig. 21 the hierarchy manager shows the depth in each genre and sub-genres (i. e., blue and green vertical lines). In this example, it is particularly important having a large context around our focus area to clearly distinguish the trends of all main genres. For instance, we observe in detail a specific time period as well as the growth of the rock genre (orange hue) over time.

The overview in the Fig. 20 shows that rock (orange hue) is the most dominant genre. This causes problems in the analysis of other genres. Using the hierarchy manager, we filtered out the rock genre, it allows us to improve the comparison between genres. Fig. 22 shows the filtered streamgraph. It is interesting to detect the time step where the rap music genre was born (Fig. 22a). A peak of the rhythm music genre (green hue) from 1965 to 1970 conveys the dominance of this genre (Fig. 22b). However, after 1975, the electronic genre (red hue) begins to grow. The multiresolution view in Fig. 22 shows the 2000 decade in full-details. The layer labeling conveys the genre that predominates in this period at a glance. Detecting the highest peaks in a period is easy, thanks to the position and the height of the labels. For example, the electronic genre reaches a peak in 2007 with 26 bands formed.

8 Conclusion

In this paper, we presented a new approach for visualizing large, multiple time series organized in a hierarchical structure. It is based on three interactive views: an overview showing the main trends of the dataset, a multiresolution view showing details on a subset of the dataset, and a hierarchy manager allowing to explore the hierarchical structure. The hierarchical organization of the time series is displayed with different granularities on the multiresolution view. Following the Shneiderman mantra "overview first, zoom and filter, then details on demand", the flexible interactive feature covers a wide range of possibilities, from overview + detail to classical fisheye.

One of the challenge during the conception phase of our approach was the interaction between the different views. We found that a set of interactions allow us to reach our goals. Thus, these techniques help to select a time segment and also navigate through the hierarchy on this time segment. As result, the multiresolution view depicts the hierarchy at different levels of details in a selected time period. Also, all the interaction are integrated and they can be achieved at any point of the exploration without losing the context.

Two examples show that our approach can be applied in many domains such as social data, music industry data. Its flexibility allows to navigate and explore hierarchies helping users to convey seasonal peaks and patterns over time. For future work, we plan to study the possibility of extending our visualization for dealing with time series available in real time, i.e. data stream.

References

[1] R. L. Harris, *Information Graphics: A Comprehensive Illustrated Reference*. Oxford University Press, 1999.

[2] S. Havre, E. Hetzler, and L. Nowell, "ThemeRiver: Visualizing Theme Changes over Time," in *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, 2000, pp. 115–123.

[3] S. Havre, E. Hetzler, P. Whitney, and L. Nowell, "ThemeRiver: Visualizing Thematic Changes in Large Document Collections," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 9–20, 2002.

[4] L. Byron and M. Wattenberg, "Stacked Graphs - Geometry & Aesthetics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1245–1252, 2008.

[5] A. Cockburn, A. K. Karlson, and B. B. Bederson, "A Review of Overview+Detail, Zooming, and Focus+Context Interfaces," *ACM Computing Surveys*, vol. 41, no. 1, pp. 2:1–2:31, 2009.

[6] M. Sarkar and M. H. Brown, "Graphical Fisheye Views," *ACM Communication*, vol. 37, no. 12, pp. 73–83, 1994.

[7] D. A. Keim, "Information Visualization and Visual Data Mining," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1–8, 2002.

[8] W. Aigner, S. Miksch, H. Schumann, and C. Tominski, *Visualization of Time-Oriented Data*. Springer, 2011.

[9] C. Perin, F. Vernier, and J. Fekete, "Interactive Horizon Graphs: Improving the Compact Visualization of Multiple Time Series," in *Proceedings of the Conference on Human Factors in Computing System*. ACM, 2013, pp. 3217–3226.

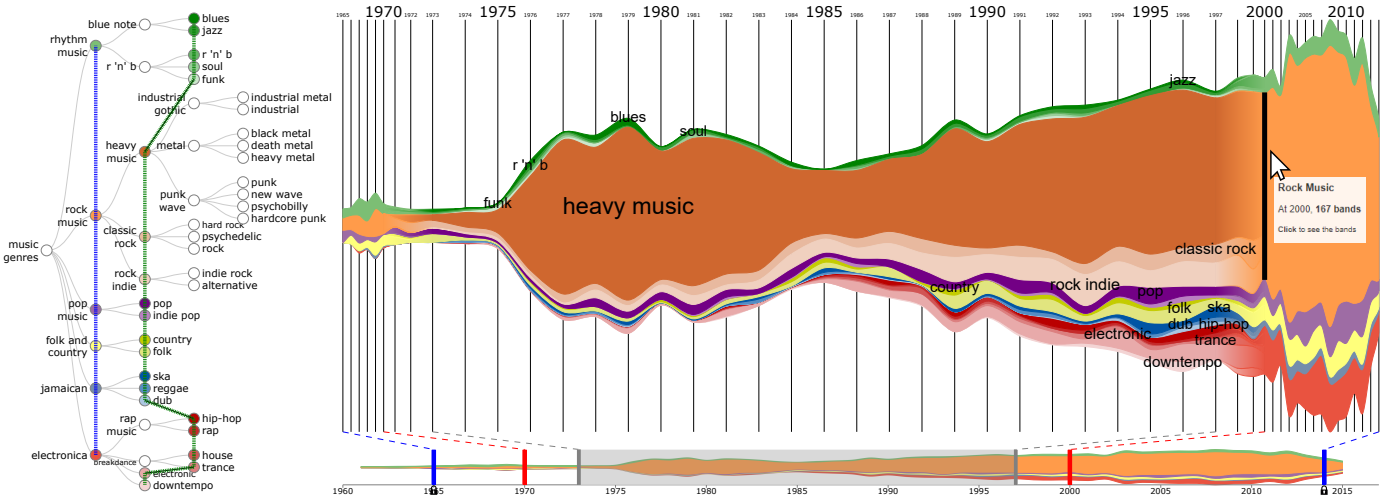

Fig. 20. Evolution of music genres from 1960 to 2016: focus on the 1975 - 1995 period.

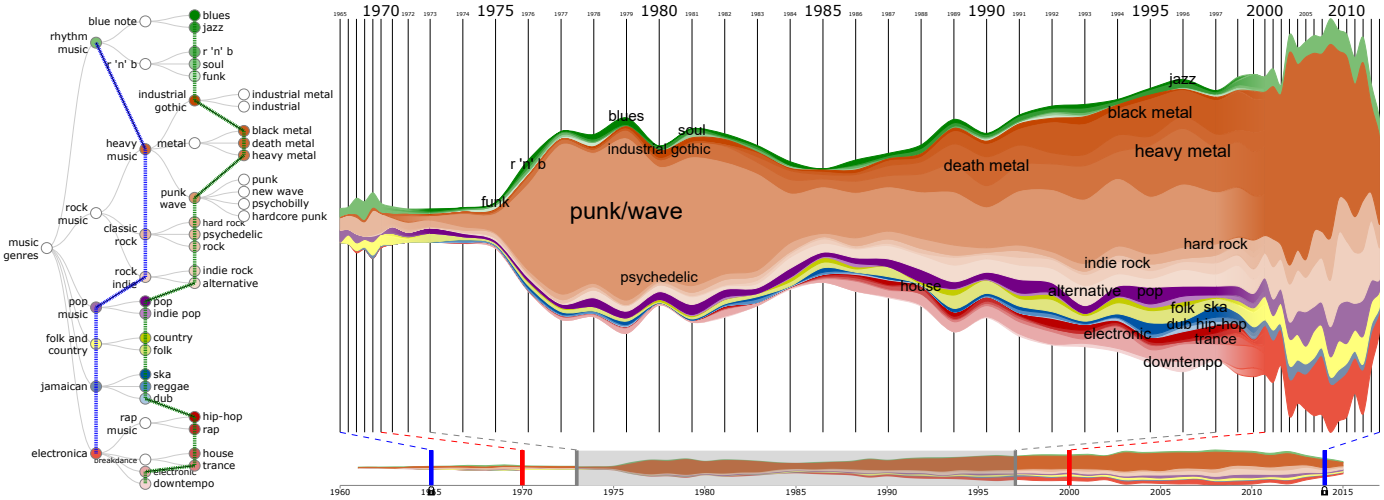

Fig. 21. Evolution of music genres from 1960 to 2016: navigation through the rock music sub-genres at different levels of detail.

[10] P. Federico, S. Hoffmann, A. Rind, W. Aigner, and S. Miksch, "Qualizon Graphs: Space-efficient Time-series Visualization with Qualitative Abstractions," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*. ACM, 2014, pp. 273–280.

[11] S. Liu, M. X. Zhou, S. Pan, Y. Song, W. Qian, W. Cai, and X. Lian, "TIARA: Interactive, Topic-Based Visual Text Summarization and Analysis," *ACM Transactions on Intelligent Systems and Technology*, vol. 3, no. 2, pp. 25:1–25:28, 2012.

[12] M. Wattenberg, "Baby Names, Visualization, and Social Data Analysis," in *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, 2005, pp. 1–7.

[13] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu, "OpinionFlow: Visual Analysis of Opinion Diffusion on Social Media," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1763–1772, 2014.

[14] G. Sun, Y. Wu, S. Liu, T.-Q. Peng, J. J. H. Zhu, and R. Liang, "EvoRiver: Visual Analysis of Topic Coopetition on Social Media," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1753–1762, 2014.

[15] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong, "TextFlow: Towards Better Understanding of Evolving Topics in Text," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2412–2421, 2011.

[16] O. Purwantiningsih, A. Sallaberry, S. Andary, A. Seilles *et al.*, "Visual Analysis of Body Movement in Serious Games for Healthcare," in *Proceedings of the IEEE Pacific Visualization Symposium*. IEEE, 2016, pp. 229–233.

[17] S. Huron, R. Vuillemot, and J.-D. Fekete, "Visual Sedimentation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2446–2455, 2013.

[18] Y. Wang, T. Wu, Z. Chen, Q. Luo, and H. Qu, "STAC: Enhancing Stacked Graphs for Time Series Analysis," in *Proceedings of the IEEE Pacific Visualization Symposium*. IEEE, 2016, pp. 234–238.

[19] L. Berry and T. Munzner, "Binx: Dynamic Exploration of Time Series Datasets Across Aggregation Levels," in *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, 2004.

[20] M. Wattenberg and J. Kriss, "Designing for Social Data Analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 549–557, 2006.

[21] F. B. Viegas, M. Wattenberg, F. Van Ham, J. Kriss, and M. McKeon, "ManyEyes: A Site for Visualization at Internet Scale," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1121–1128, 2007.

[22] D. Baur, B. Lee, and S. Carpendale, "TouchWave: Kinetic Multi-touch Manipulation for Hierarchical Stacked Graphs," in *Proceedings of the International Conference on Interactive Tabletops and Surfaces*. ACM, 2012, pp. 255–264.

[23] M. Ghoniem, D. Luo, J. Yang, and W. Ribarsky, "NewsLab: Exploratory Broadcast News Video Analysis," in *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*. IEEE, 2007, pp. 123–130.

[24] W. Dou, L. Yu, X. Wang, Z. Ma, and W. Ribarsky, "HierarchicalTopics: Visually Exploring Large Text Collections Using Topic Hierarchies," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2002–2011, 2013.

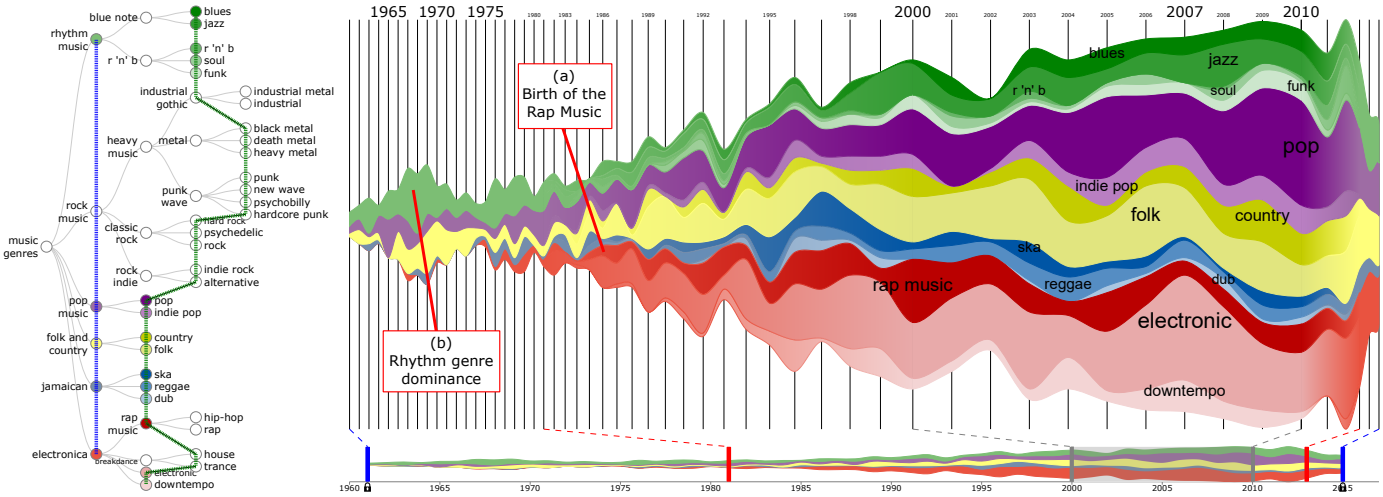

Fig. 22. Evolution of music genres from 1960 to 2016: focus on the 2000 - 2010 period.

[25] W. Cui, S. Liu, Z. Wu, and H. Wei, "How Hierarchical Topics Evolve in Large Text Corpora," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2281–2290, 2014.

[26] G. W. Furnas, "Generalized Fisheye Views," in *Proceedings of the Conference on Human Factors in Computing System*. ACM, 1986, pp. 16–23.

[27] R. Bade, S. Schlechtweg, and S. Miksch, "Connecting Time-Oriented Data and Information to a Coherent Interactive Visualization," in *Proceedings of the Conference on Human Factors in Computing System*. ACM, 2004, pp. 105–112.

[28] R. A. Becker and W. S. Cleveland, "Brushing Scatterplots," *Technometrics*, vol. 29, no. 2, pp. 127–142, 1987.

[29] J. D. Mackinlay, G. G. Robertson, and S. K. Card, "The Perspective Wall: Detail and Context Smoothly Integrated," in *Proceedings of the Conference on Human Factors in Computing System*. ACM, 1991, pp. 173–179.

[30] M. S. T. Carpendale, "A Framework for Elastic Presentation Space," Ph.D. dissertation, Simon Fraser University, 1999.

[31] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner, "Navigating Hierarchies with Structure-Based Brushes," in *Proceedings of the IEEE Symposium on Information Visualization*. IEEE, 1999, pp. 58–64.

[32] J. D. Manning, B. E. Marciano, and J. J. Cimino, "Visualizing the Data Using Lifelines2 to Gain Insights from Data Drawn from a Clinical Data Repository," *AMIA Summits on Translational Science Proceedings*, vol. 2013, p. 168, 2013.

[33] F. Y. Wang, A. Sallaberry, K. Klein, M. Takatsuka, and M. Roche, "SentiCompass: Interactive Visualization for Exploring and Comparing the Sentiments of Time-varying Twitter Data," in *Proceedings of the IEEE Pacific Visualization Symposium*. IEEE, 2015, pp. 129–133.

[34] B. Shneiderman, "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations," in *Proceedings of the IEEE Symposium on Visual Languages*. IEEE, 1996, pp. 336–343.

[35] M. A. Jenkins, "Algorithm 493: Zeros of a Real Polynomial [c2]," *ACM Transactions on Mathematical Software*, vol. 1, no. 2, pp. 178–189, 1975.

[36] L. Feldman Barrett and J. A. Russell, "Independence and Bipolarity in the Structure of Current Affect," *Journal of Personality and Social Psychology*, vol. 74, no. 4, pp. 967–984, 1998.

Erick Cuenca is a PhD student at the University of Montpellier, France. He's also a member of the data mining and visualization research group (ADVANSE) of the Montpellier Laboratory of Informatics, Robotics and Microelectronics (LIRMM). His thesis focuses on datastream and multivariate graph visualization. Cuenca received a Master's degree in computer science from the University of Montpellier, France. Contact him at erick.cuenca@lirmm.fr.

Arnaud Sallaberry is an assistant professor at the Paul Valéry University of Montpellier, France. He's also a member of the data mining and visualization research group (ADVANSE) of the Montpellier Laboratory of Informatics, Robotics and Microelectronics (LIRMM). His research interests include information visualization, visual analytics, and graph drawing. Sallaberry received a PhD in computer science from the University of Bordeaux, France. Contact him at arnaud.sallaberry@lirmm.fr.

Florence Y. Wang is a scientific visualization specialist at CSIRO, Australia. She received her PhD degree from the University of Sydney, Australia. Prior to joining CSIRO, she worked as a postdoctoral researcher for textual data visualization at the laboratory of informatics, robotics and microelectronics of Montpellier, France. Her research interests include information visualization, visual analytics, virtual reality and user interaction. Contact her at Florence.Wang@csiro.au.

Pascal Poncelet is a full professor at the University of Montpellier, France, and head of the data mining and visualization research group at the Montpellier Laboratory of Informatics, Robotics and Microelectronics (LIRMM). His research interests include advanced data analysis techniques for emerging applications, data mining techniques, and new algorithms for mining patterns. Poncelet received a PhD in computer science from University of Nice-Sophia Antipolis. Contact him at pascal.poncelet@lirmm.fr.