- ORIGINAL ARTICLE -

GLC-Frame: A Framework and Library for Exploration of Multidimensional Data with General Line Coordinates

GLC-Frame: Un Framework y Libreria para la exploracion de Datos Multidimensionales con Coordenadas Generales de Línea

Leandro Luque[1,2], Antonella Antonini[1,2], Maria Lujan Ganuza[1,2], and Silvia Castro[1,2]

[1] *VyGLab, UNS-CIC, Argentina*
{leandro.luque, antonella.antonini}@cs.uns.edu.ar
[2] *Institute for Computer Science & Engineering, UNS-CONICET, Argentina*

Abstract

General Line Coordinates (GLC) are a relatively new set of line-based representations for visualizing multidimensional data with the distinctive characteristics of being reversible and lossless. Given these characteristics, the GLC have a high potential for exploratory multidimensional data analysis, however only implementations of specific GLC techniques are available for the visualization community. In this paper, we present the *GLC-Frame*, an online exploration tool that supports a dual view and allows users to upload their own dataset and interactively explore the different GLC representations without writing code. We also present the *GLC-Vis* Library, an open-source data visualization library supporting GLC along with traditional interactions. Finally, we provide a set of usage examples showing how the different techniques behave in both the occlusion and the cluster identification problem. In addition, we present the interactions on GLC representations using the cars dataset. Both the *GLC-Frame* and the *GLC-Vis* Library provide an exploration space that will allow the visualization community to use these new techniques and evaluate their potential.

Keywords: Visual Analysis, General Line Coordinates, Interactions in General Line Coordinates, Visualization of Multidimensional Data

Resumen

Las Coordenadas Generales de Línea (GLC) son una serie de representaciones visuales basadas en líneas relativamente nuevas para visualizar datos multidimensionales con las características distintivas de ser reversibles y sin pérdidas. Dadas estas características, las GLC tiene un alto potencial para el análisis exploratorio de datos multidimensionales, sin embargo, solo existen implementaciones específicas de algunas de las técnicas de GLC disponibles para la comunidad de visualización. En este artículo, presentamos *GLC-Frame*, una herramienta de exploración en línea que admite una vista dual y permite a los usuarios cargar su propio conjunto de datos y explorar interactivamente las diferentes representaciones de GLC sin necesidad de escribir código. También presentamos *GLC-Vis* Library, una biblioteca de visualización de datos de código abierto que admite las representaciones GLC junto con interacciones tradicionales. Finalmente, proporcionamos un conjunto de ejemplos de uso que muestran cómo se comportan las diferentes técnicas tanto en el problema de la oclusión como en la identificación de clusters. Además, presentamos las interacciones en las representaciones de GLC utilizando un conjunto de datos conocido de automóviles. Tanto *GLC-Frame* como *GLC-Vis* Library proporcionan un espacio de exploración que permitirá a la comunidad de visualización utilizar estas nuevas técnicas y evaluar su potencial.

Palabras claves: Análisis Visual, General Line Coordinates, Interacciones en General Line Coordinates, Visualización de Datos Multidimensionales

1 Introduction

Visualization of multidimensional (n-D) datasets is a major challenge, as visualization methods must overcome the difficult problem of mapping complex high-dimensional data into representations suitable for visualization. One approach to this problem, considering the human capacity of perception, is to represent the data in a low-dimensional space. Thus, visualization techniques must provide low-dimensional representations but, at the same time, they must preserve certain properties of the structure of the dataset (like clusters, outliers, distances, and topology) as faithfully as possible [1].

In this regard, multiple visualization methods for multidimensional data have been proposed over the last decades [2, 3, 4]. However, many of them do not allow the complete restoration of the data from its reduced representation or do not represent the complete dataset. Undoubtedly, the design of reversible and lossless methods constitutes one of the main issues in multidimensional visualization.

A limited number of reversible and lossless visual representations have been developed. The Parallel and Radial Coordinates are two of the most popular and valuable alternatives [5, 6]. These techniques have been proven to be very useful in various application areas revealing information from various datasets, however, both suffer several problems, occlusion being the most characteristic [7]. Recently, Kovalerchuk [8] and Kovalerchuk and Grishin [9] introduced a new reversible and lossless approach for visualizing multidimensional data called General Line Coordinates (*GLC*). The *GLC* are a relatively new concept, hence the need for a library that allows the use of those techniques for the analysis of large datasets. To the best of our knowledge, there are only implementations of some specific *GLC* techniques available in the literature.

In this context, we present a web-based open-source data visualization library, *GLC-Vis*, which supports all the *2D-GLC* techniques and traditional interactions. This library is an extension of *npGLC-Vis* [10] and supports traditional interactions like panning, zooming, axes reordering, and brushing and linking. We also present the *GLC-Frame*, a web exploration tool to upload multidimensional datasets and interactively explore different *GLC* configurations without writing a single line of code. This tool provides a dual view mechanism to show a dataset under two different representations, a *NP-GLC* and a *P-GLC* technique. This feature empowers users to visually compare the same dataset from multiple angles, which can be immensely useful in gaining deeper insights and identifying patterns that might not be evident with a single representation. In addition, brushing and linking between these views is supported, favoring exploration. By brushing and linking between representations, users can easily spot correlations, trends, and anomalies, ultimately facilitating more effective data analysis and exportation. Both the *GLC-Vis* library and the *GLC-Frame* tool will allow the visualization community to explore the potential of these techniques in an easy and accessible way. In addition, we include several usage examples to show how different representations may or may not be better than others in different scenarios.

This paper is organized as follows: We begin by establishing the context of the problem in the Related Work and General Line Coordinates sections. Following that, we present the *GLC-Vis* (and its sub modules) and the *GLC-Frame* built on top of it, describing the interactions it provides. Subsequently, we introduce several Usage Scenarios based on three main concerns in visualization techniques: occlusion, line crossing, relationship, and cluster identification. Finally, in the Discussion and Conclusion sections, we analyze our results, evaluate the limitations of our work, and suggest directions for future research.

2 Related Work

The analysis of multidimensional (n-D) data is a longstanding task that motivated the development of a broad spectrum of visualization techniques that have proven to be very valuable for exploratory multidimensional data analysis [2, 3, 4]. The earlier contributions focused on line-based methods [11, 12, 13], that mostly employ single 2-D or 3-D plots where each attribute value of a data item is represented by a point on an axis, and a data item is represented with a straight or curved line that joins these points. These line-based visualization methods play a relevant role since they constitute a lossless projection of n-D data space onto 2-D or 3-D space. However, these techniques present some problems like occlusion and cluttering.

From 2014 onwards, Kovalerchuk [8, 14] and Kovalerchuk and Grishin [9] introduced the *GLC* techniques for visualizing, losslessly, multidimensional data in 2-D and 3-D and identified two classes of *GLC*: the Non-Paired *GLC* (*NP-GLC*) that generalize the Parallel Coordinates and the Radial Coordinates, and the Paired *GLC* (*P-GLC*) that generalize the Cartesian Coordinates. The demand for implementations emerges in parallel with the development of the *GLC* so that users can use these representations to explore their own datasets.

Commercial and public tools like Plotly [15] and D3.js [16] can be used for creating custom solutions to solve a specified task or problem. Plotly provides functions to build Parallel and Radial Coordinates plots, however, it presents limitations regarding the graphical elements that can be used to visually map the data items and provides a restricted set of basic interactions. *D3.js* is a powerful and flexible data visualization library that offers low-level graphical primitives to build visualizations from scratch, nevertheless, its learning curve is somehow steep [16].

In terms of dedicated software and open-source libraries supporting *GLC* techniques, we can notice the absence of a complete and integrated tool that allows the study and comparison of these techniques. Some libraries only cover one particular technique in detail, like Parasol [17] (which focuses exclusively on Parallel Coordinates) and EllipseVis[18] (which focuses exclusively on Elliptic Paired Coordinates). In 2021, Luque et.al [10] presented the *npGLC-Vis*, a library that supports a specified subset of the *GLC* techniques, covering exclusively the Non-Paired branch and traditional interactions like brushing, zooming and panning. The *npGLC-Vis* served as an excellent starting point in the exploration of *GLC* techniques, however, the need for a unified library that also provides support for the Paired *GLC* techniques remained. EllipseVis [18] and DSCVis [19] present particular cases of *GLC-P* implementations as desktop applications. On the other hand, DV2.0 [20] adds *GLC-Linear* and some specific extensions to generate interpretable visualizations for classification and regression models.

This motivated the design and development of *GLC-Vis*, a web-based open-source data visualization library, that supports all the *GLC* techniques. This library supports the *P-GLC* techniques that generalize the Cartesian Coordinates. The *GLC-Vis* library, also integrates the (*NP-GLC*) techniques, previously supported in *npGLC-Vis* library.

In addition, we also present a web exploration tool, called *GLC-Frame*, that enables the interactive exploration of multidimensional datasets using the *GLC* techniques without writing a single line of code, therefore allowing the visualization community to test the potential of these techniques in an easy and accessible way. It's crucial to emphasize that previous research has primarily focused on individual forms of *GLC*, while *GLC-Frame* introduces an innovative approach, offering a dual-view capability that enables the simultaneous visualization of two distinct representations of *GLC* enriched with interactions like brushing and linking that allow the establishment of a direct relation between views. Also, *GLC-Frame* constitutes an alternative to current implementations of *GLC* and it is beneficial for different types of users.

On the other hand, the various examples provided in section 6 illustrate scenarios that frequently occur when visualizing multidimensional data, and they provide an opportunity to see how various representations may or may not be useful for carrying out particular tasks or solving particular issues.

3 General Line Coordinates (*GLC*)

General Line Coordinates (*GLC*), including *NP-GLC* and *P-GLC*, are a set of line-based representations for visualizing multidimensional data whose distinctive characteristics are reversibility and losslessness [14, 8, 9]. These techniques arrange a set of coordinate axes $X_1 - X_{na}$, being *na* the number of axes to represent a n-D dataset. From now on we use $\mathbf{x}$ to notate a n-D data item, and x_i for the $i-th$ dimension value of $\mathbf{x}$.

The *NP-GLC* directly generalize the Parallel and Radial Coordinates. Then, each $\mathbf{x}$ data item is represented by a polyline intersecting each X_i axis at the point corresponding to the x_i value. The various *NP-GLC* representations [8] result from the combination of different arrangements of the axes and different ways of drawing the data items. Besides, axes layouts can be sequential (the coordinate axes are placed one after the other) or non-sequential (the axes are freely oriented). The graphical elements to represent the data items and/or axes can be curves, straight lines or hybrids (curves and straight lines) and, finally, the mapping to represent a data element can be static or dynamic. In the static approach, each x_i coordinate of x is located in X_i, while in the dynamic approach, the location of x_{i+1} is calculated as a function of the location and the x_i value. The combination of all these characteristics determines multiple layouts, including n-Gon, Circular, In-Line, and Bush Coordinates.

The *P-GLC* generalize the Cartesian Coordinates. In these techniques, an n-D data item $\mathbf{x}$ is split into consecutive disjoint pairs $(x_1,x_2),...,(x_{n-1},x_n)$, each one representing a 2-D point associated to a pair of axes. Thus, an n-D data item is represented as a directed graph based on the idea of a collection of pairs of dimensions, producing much less clutter than the *NP-GLC* methods. In addition, they require half as many lines as *NP-GLC* such as Parallel and Radial Coordinates. The *P-GLC* include Collocated, Shifted, Anchored, Radial, Elliptic, and Crown Paired Coordinates [8]. While *NP-GLC* techniques facilitate static and dynamic mapping due to each location of x_i relying on a unique X_i, *P-GLC* techniques specifically support dynamic mapping in certain versions.

The *P-GLC* and *NP-GLC* are defined and illustrated in table 1.

A more detailed and comprehensive explanation of the mathematical principles that support the development of GLC can be found in Chapter 3 of Kovalerchuk [8].

4 *GLC-Vis*

4.1 Library Overview

GLC-Vis is an open-source library designed and implemented with *D3.js* [16]. This library allows the user to build custom visualization techniques and is widely supported in web browsers. For platform independence, we selected a web-based solution, rather than a desktop solution. However, the capabilities and computing performance of web-based solutions are limited for large datasets. Some implementations of visualization libraries use a GPU-based scheme to handle this issue, but this solution imposes a strong requirement for the user in terms of hardware. To avoid this limitation, we discarded a GPU implementation, but internal implementation allows adding this feature in case developers want to outperform our solution. For those cases where it is necessary to handle sensitive data and the web application is not an option, it is possible download the source code of *GLC-Frame* and compile it with Electron for a native app.

GLC-Vis supports all the *2D-GLC* techniques and traditional interactions. While many previous works identify interactions for Parallel and Radial Coordinates, there is no study that matches the other *GLC* approaches to see which interactions should be supported and how they impact the data analysis. In this context, the proposed library provides native support for elemental interactions such as zooming and panning, axes' orientation modification, and color and opacity modification, which are common to most techniques.

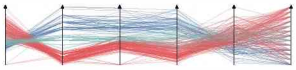
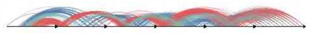
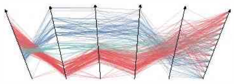
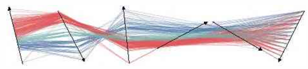
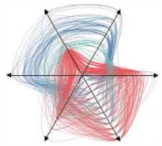
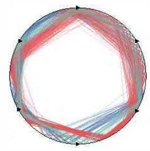
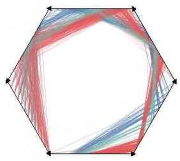
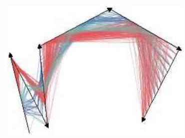
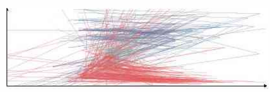
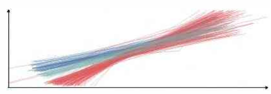
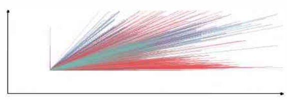
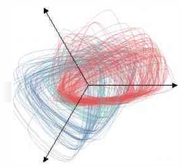
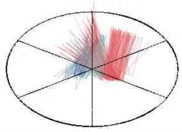
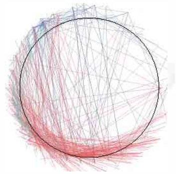
Non-Paired General Line Coordinates (*NP-GLC*)	
Parallel Coordinates  Set of equidistant vertical axes arranged from left to right.	In-Line Coordinates  Axes are horizontal, collinear, and oriented in the same direction; they may or may not overlap.
Bush Coordinates  This layout is a modification of the Parallel, where only the middle coordinate axis is vertically oriented. All other coordinate axes are tilted increasingly from the middle axis to shape a bush.	Generic Non-Sequential Coordinates  Axes are oriented in any direction, spaced apart, and arranged consecutively on the plane from left to right.
Radial Coordinates  Axes radiate at equal angles from a common origin.	Circular Coordinates  Axes are curved segments arranged to form a circle.
N-Gon Coordinates  Axes are arranged to form a polygon (triangle, square, pentagon, etc.). Each edge of the polygon corresponds to an axis and all of them are oriented clockwise or counterclockwise.	Generic Sequential Coordinates  Axes are arranged consecutively on the plane one after the other, clockwise or counterclockwise, and each axis is oriented in any direction.
Paired General Line Coordinates (*P-GLC*)	
Collocated Paired Coordinates  Each pair is drawn as a 2D point in the same two axes on the plane and these 2D points are linked to form a directed graph.	Shifted Paired Coordinates  After the first pair, each next pair is drawn in a shifted coordinate system.
Anchored Paired Coordinates  After the first pair, each next pair is drawn shifted, that is, the coordinates are displaced to the location of a given pair (anchor).	Paired Radial Coordinates  The circle is divided on $n/2$ equal sectors. Each pair is plotted in its own sector as a point, located at the distance r from the center
Ellipse Paired Coordinates  Each pair is represented as the intersection of their corresponding ellipses. These points are linked to form a poly-line.	Paired Crown Coordinates  Given a convex hull, the odd dimensions are plotted on it and the even coordinates are plotted orthogonal as a function of the odd coordinate.

Table 1: Summary of different GLC representations supported by the *GLC-Vis* Library.

However, as the library supports direct manipulation of axes and data items, it can be easily extended to support other interactions like axes reordering, brushing and linking, etc. As we will see in the following section, the library's modular design makes it simple to upgrade or add new functionalities as required.

In figure 2 we show the architecture of the library and how the different parts of the specification are processed.

4.2 API

In this section, we discuss the first steps for creating a *GLC* chart and the design aspects involved in the library. First of all, it is important to mention that it is necessary a minimal background in web programming to use our solution. However, the documentation and the examples provided on the website library are enough for novice developers. Like other Javascript-based libraries, it is mandatory to define a container space where the chart will be drawn. After that, in the script section, we define the parameters of the desired chart and call to *draw()* for generating it.

At the design level, our library follows the guidelines recommended for the *D3.js* community for reusable and independent charts favoring its integration with other existing libraries. With this in mind, we defined two large modules containing respectively the techniques related to the Paired and Non-Paired versions. This organization allows more flexibility to improve the implementation of a provided technique or develop a new one not available in this version of the library.

As mentioned in section 2, the *GLC-Vis* is presented as an extension of the *npGLC-Vis* [10]. In figure 1, we illustrate the inner structure of the library. Highlighted in the darker font, we show how the new modules of Paired *GLC* techniques were incorporated to extend the *npGLC-Vis* library. This structure allows modifying or adding new techniques if it is required.

The API for interacting with the charts has three families of functions, that is those related to graphic elements (polyline) and their properties (color, opacity), to the axes layout, and to a specific chart. The functions *setOpacity()*, *colorize()*, *setLine()*, and *getItems()* allow the developer to interact directly with the data items while the function *getAxis()* allows the developer to get the axes and their spatial position according to the employed layout. Another group of functions is linked to the technique. For instance, according to the definition of InLine layout, it has two functions to interact with it: *compareClasses()* and *focusClass()*. A similar situation occurs with other techniques like Radial, Circular, and Generic.

The *GLC-Vis* provides three traditional interactions to manipulate the generated charts: panning, zooming and brushing. Moreover, as it supports direct manipulation of data items and axes, the library can be extended to support any potential interaction over these elements. To take a case in point, the *GLC-Frame*, which was implemented upon the *GLC-Vis*, supports a rich set of interactions like axes reordering, filtering by classes, opacity and color modification, etc. All these interactions were implemented using the *GLC-Vis* and are described in detail in section 5.

A complete specification of *GLC-Vis* is available on Github [1]. For a full programmer's guide please visit the repository.

4.3 Reordering Algorithms

GLC-Frame incorporates a module with different axis reordering algorithms based on the work of Blumenschein et al. [21] as an alternative to improve the quality of the generated visualization. While these methods are intended solely for parallel coordinates plots, they can serve as a starting point for analyzing their impact on different GLC techniques.

The user can choose from various reordering alternatives before (or after) plotting the dataset for analysis. Some of the available methods include:

- Peng et al. [22] define outliers as data points that don't belong to a cluster and propose measuring the outlier ratio against the total number of data points. Maximizing the ratio highlights outliers, while minimizing it avoids highlighting outliers.

- Lu et al. [23] suggest sorting axes based on their contributions to the dataset. They use singular value decomposition (SVD) to compute the dimensions' singular values and sort them accordingly.

- Artero et al. [24] introduce a metric based on ordering axes to make lines more parallel or divergent. This approach helps identify correlations but may favor clusters to some extent. They propose using the total length of polylines as a pattern metric.

- Makwana et al. [25] propose a different metric that focuses on ordering dimensions to create neighboring axes with different line slopes, resulting in a cluttered appearance.

- Blumenschein et al. [21] note that polylines and clusters with strong slopes are visually more prominent than horizontal ones. Two reasons contribute to this effect: increasing slopes reduce the distance between polylines, and diagonal lines require more pixels to encode a single data point, resulting in a low data-to-ink ratio. These geometric effects facilitate perceiving neighboring lines as a group or cluster. Interestingly, strong slopes are achieved by ordering dimensions based on dissimilarity.

[1] https://github.com/visualprojects/glcvis

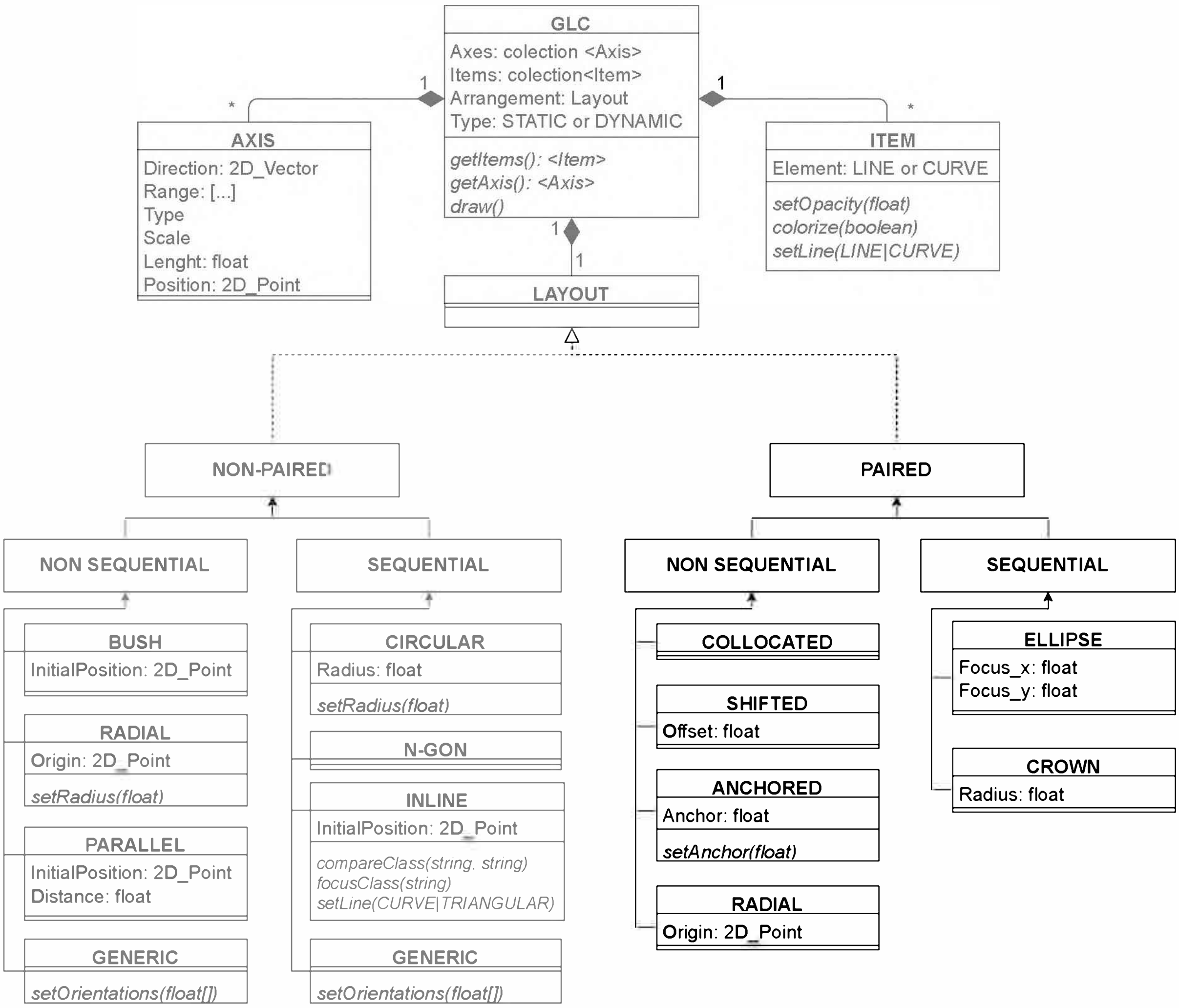


Figure 1: Inner structure of *GLC-Vis*. The new modules of Paired *GLC* techniques that were incorporated to extend the *npGLC-Vis* library are highlighted in the darker font. The modules drawn in the lighter font correspond to the previous work Luque et al. [10].

For further insights on these algorithms, we recommend referring to [21].

5 *GLC-Frame*

In addition to the library, we provide *GLC-Frame*, a web application tool built upon the *GLC-Vis* that allows the user to upload their own datasets and, through an easy-to-use user interface, interactively explore the different *GLC* representations without the need of writing code.

The application user interface is composed of two sections: the Configuration Section and the Visualization Section (see figure 3). The Configuration Section includes a data upload area (see figure 3A) and some popular datasets that users can download and use in the analysis of their data (see figure 3B).

GLC-Frame supports a rich set of interactions which are briefly explained below:

- Selecting the *GLC* Technique: In the Visualization Section, the user can select (and later change) the *GLC* technique used to represent their data from a drop-down menu (see figure 3G).
- Axes Reordering: The Configuration Section includes an area where the dimensions of the loaded data set are listed (see figure 3C). The user can interactively change the dimensions' order by dragging them to reorder the associated axes in the different representations.
- Opacity and Color mapping: The user is able to modify the level of opacity and the color of the lines representing the data items by using the corresponding checkbox and slider in the Configuration Section (see figure 3D). Once the order of the dimensions has been changed, the views must be redrawn to apply the changes.
- Filtering by Classes: Through this interaction, the user may choose which class to show on the graph from a list of available classes. In order to select a

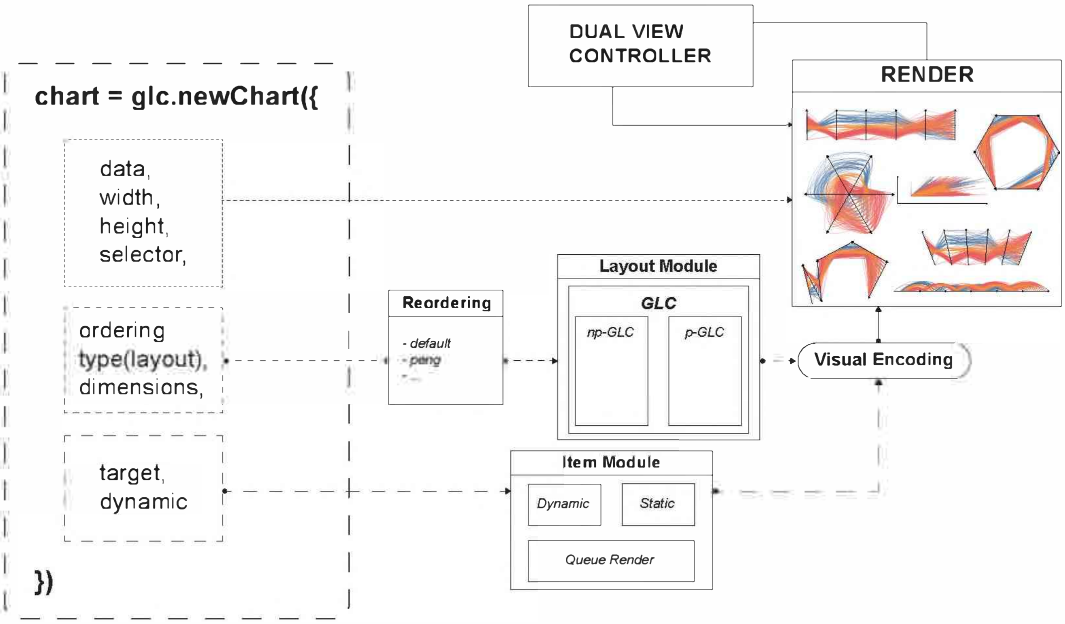


Figure 2: Architecture of *GLC-Vis* library

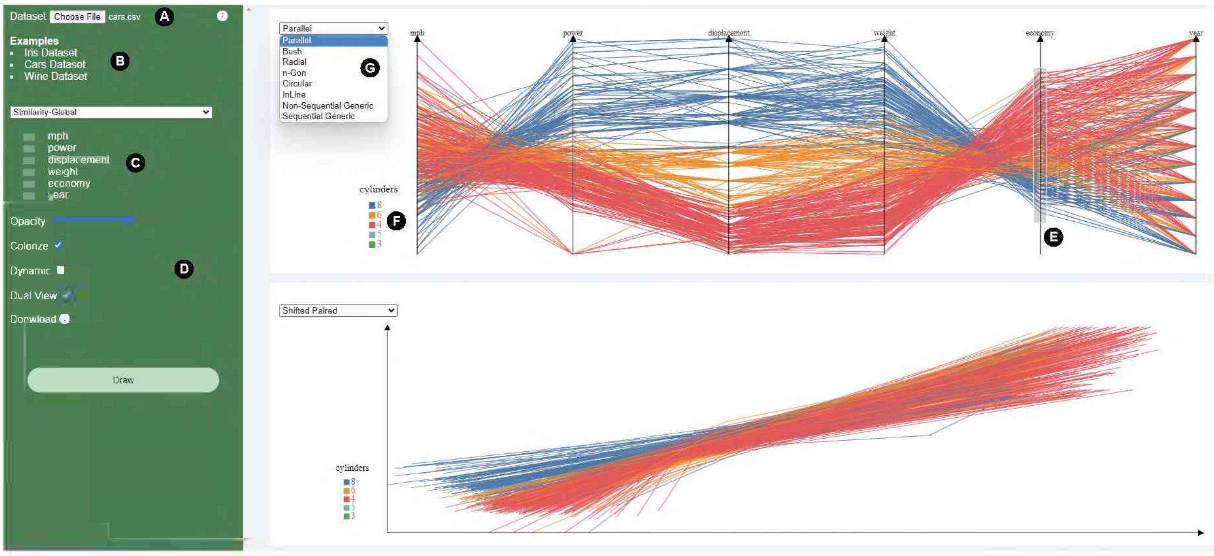


Figure 3: The user interface of the *GLC-Frame* is integrated by the Configuration Section and the Visualization Section. The first one is composed of options for uploading a custom dataset (A) or using the default ones provided (B). Additionally, there are options available for axis reordering (C) and visualization of data items (D). In the second section, there are panels with the selected GLC techniques and their options for manipulation (E, F, G).

class, the user must click the corresponding label on the references in the Visualization Section (see figure 3F).

- Dual View Activation: The provided tool also supports dual views. Users can explore their data in the Visualization Section using an *NP-GLC* technique and a *P-GLC* technique at the same time. The Dual View can be activated/deactivated in the Configuration Section (see figure 3D).

- Brushing and Linking: When Dual View is active, any selection (brushing) performed on the *NP-GLC* chart will be consistent in the *P-GLC* chart (linking), making it easier to interpret and compare the two charts (see figure 3E).

- Zooming: This interaction is supported in the Visualization Section. It allows the user to zoom in or out of views by rolling the mouse wheel up or down.

- Panning: This interaction is supported in the Visualization Section, allowing moving views by clicking and holding the left mouse button, and dragging the mouse in any direction.

GLC-Frame is implemented with PreactJS [26] to minimize the size of the final bundle and avoid unnecessary configurations that impact loading performance. It is available online [2], can be freely accessed from any browser (for example, Google Chrome, Internet Explorer, etc.), and does not require any installation process.

6 Usage Examples

In this section, we explore many datasets using the *GLC-Frame* to show how the library's capabilities make it possible to, among other tasks, compare and analyze various *GLC* techniques.

6.1 Occlusion in *GLC*

As is well known, although Parallel Coordinates are very useful for n-D data visualization, they suffer from partial or total occlusion. In this first scenario, we analyze how the drawing order has a different impact on how the data is visualized, producing total or partial occlusion, depending on the chosen *GLC* technique. In this first example, we analyze a synthetic dataset consisting of 37 data items, classified into four classes, and 24 dimensions (see figure 4). After loading the dataset, we can start by selecting the Parallel Coordinates technique to display the full dataset (using the drop-down list illustrated in figure 3G). The data correspond to four different classes (*Class 1* to *Class 4*). By filtering by classes (see section 5), we obtained the representations of each of the four different classes (see figures 4A-D). When plotting all the data items together (without filtering), it can be observed that the data items corresponding to *Class 1* and *Class 2* are totally occluded (see figure 4E). The order in which the data items are drawn determines which elements will be occluded. In the case of figure 4E the drawing order is the following: first, the data items corresponding to *Class 1* are drawn, then the data items corresponding to *Class 2*, then those corresponding to *Class 3* and finishing with those corresponding to *Class 4* (*C*1-*C*2-*C*3-*C*4). We can also use the same dataset but with the data in a random order, without considering the classes, and configure the Parallel Coordinates view again 4F. In this case, the lines of different colors interleave generating new patterns that are perceived as misleading colors. Details from the upper and lower part of this plot are shown in figure 4F.1 and figure 4F.2.

Occlusion is also a problem when plotting the dataset in Radial Coordinates. Figure 4G.1 shows the result of configuring the Radial Coordinates technique in the *GLC-Frame* and plotting all the data items in the same order as figure 4E (*C*1-*C*2-*C*3-*C*4). Again, it can be observed that the data items corresponding to *Class 1* and *Class 2* are totally occluded and the order in which the data items are drawn determines which elements will be occluded. In fact, if we change the order in which the data items are plotted (see figure 4G), the total occlusion problem remains, and if classes are plotted in different orders, different color patterns emerge.

As the next step in our analysis, we plotted the same dataset by configuring a dynamic Radial Coordinate technique (see figure 4H) and a Shifted Paired Coordinates technique (see figure 4I). In both cases, there is no total occlusion and the plots show a clear differentiation of the classes regardless of the order in which the items are plotted. The Shifted Paired Coordinates (see figure 4I) show partial occlusion for this dataset. Although all classes can be differentiated, there is partial occlusion between the data items corresponding to classes 1 and 2 with the data items corresponding to *Class 3* and between the element corresponding to classes 1 and 2 with those corresponding to *Class 4* (see figure 4I.1 for more detail).

Nevertheless, if we decrease the opacity of the data items (see section 5), we can see how the classes can be distinguished (see figure 4I.2). Finally, when plotting the dataset using dynamic Radial Coordinates (see figure 4H) it can be seen that there is only partial occlusion at the origin of coordinates, but everywhere else the classes are clearly differentiated. Moreover, regardless of the order in which the dataset is drawn, total occlusion is not a problem for these two representations.

6.2 Line Crossings in *GLC*

In Parallel and Radial Coordinates, crossing lines produce different patterns that represent relationships between the dimensions they connect. However, line crossings can add clutter and make it difficult to identify which lines are going where. In this scenario, we aim to explore how line crossing differs based on the *GLC* technique employed. In this case, we analyze two datasets each consisting of 201 data items, classified in two classes and 8 dimensions (*Synthetic 1* and *Synthetic 2* respectively). We start analyzing the datasets with the Parallel Coordinates technique. As already mentioned in section 5, the application supports an interaction that allows the coloring of the data items by classes, when classes are determined. As shown in figures 5B and 5C, when drawing these datasets assigning different colors to different classes, it becomes apparent that the structure of those classes is completely different. However, both representations are difficult to differentiate unless we color the items associated with each class with different colors (see figure 5A). Several different datasets plotted in Parallel Coordinates could give identical results. When multiple polylines intersect an axis at the same point in Parallel Coordinates, it becomes challenging to de-

[2] `https://visualprojects.github.io/`

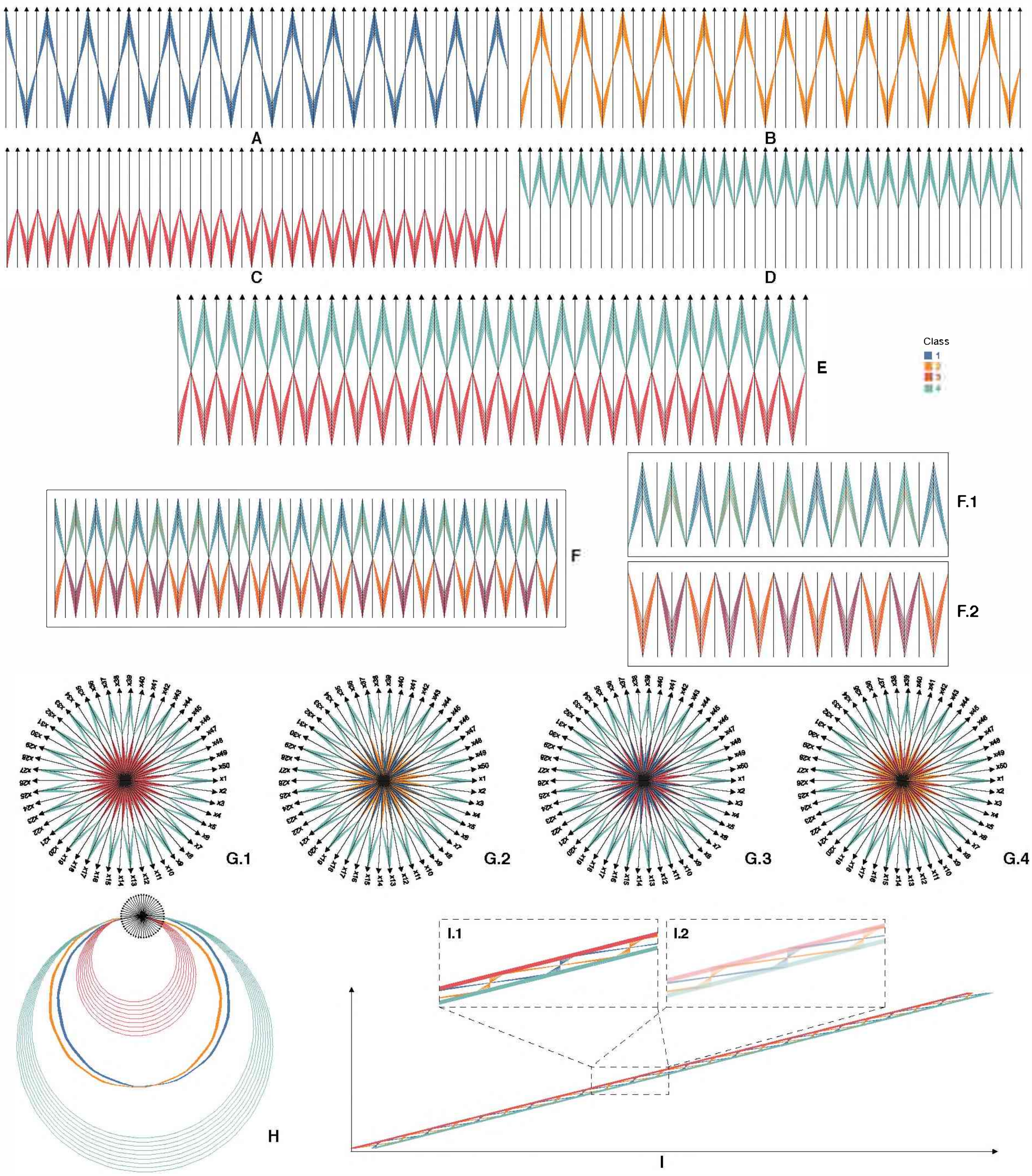


Figure 4: The four classes of a synthetic dataset are plotted using Parallel Coordinates. Data items corresponding to *Class 1* are plotted in (A), data items corresponding to *Class 2* are plotted in (B), data items corresponding to *Class 3* are plotted in (C), and data items corresponding to *Class 4* are plot in (D). The whole dataset is plotted using Parallel Coordinates in (E) drawing at first all the elements corresponding to *Class 1*, then those corresponding to *Class 2*, then those corresponding to *Class 3*, and finally those corresponding to *Class 4* (*C*1-*C*2-*C*3-*C*4). (F) All items in the complete data set are plotted in random order. (G) The dataset is plotted in Radial Coordinates drawing the classes in different orders: (G.1) *C*1-*C*2-*C*3-*C*4, (G.2) *C*3-*C*1-*C*2-*C*4, (G.3) *C*2-*C*3-*C*1-*C*4, and (G.4) *C*1-*C*3-*C*2-*C*4. The same dataset plotted using dynamic Radial Coordinates in (H) and Shifted Paired Coordinates in (I).

termine each polyline's specific orientation without a further visual cue like color, evidencing the line crossings problem.

A similar problem occurs with Radial Coordinates (see figure 5D). Next, we use the Radial Coordinates technique to color-code the two data sets according

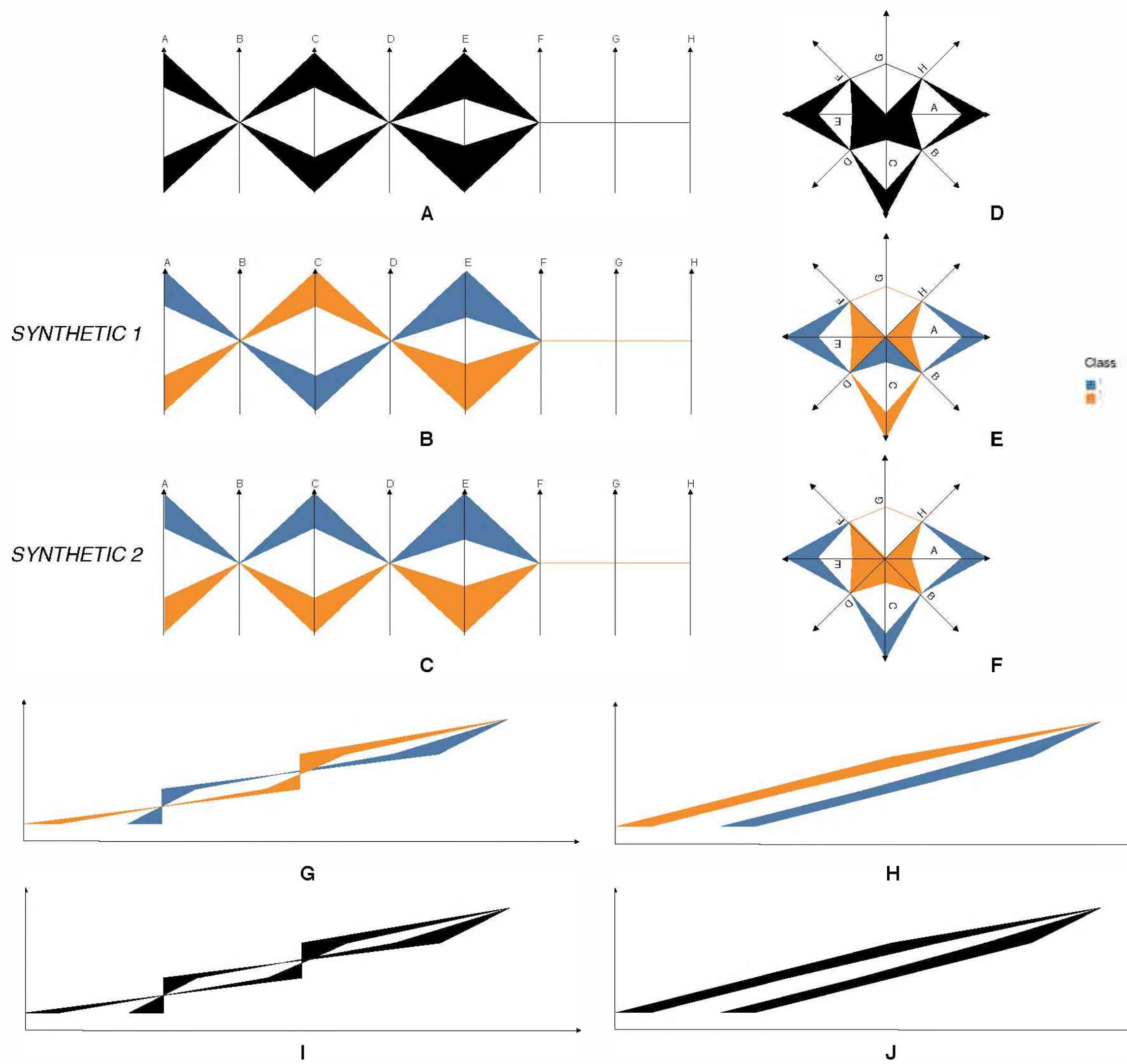


Figure 5: Line Crossing in Parallel Coordinates. (A) The *synthetic 1* and *synthetic 2* yield identical representations when drawn in Parallel Coordinates without coloring per classes. (B) The *synthetic 1* and (C) the *synthetic 2* datasets plotted in Parallel Coordinates with coloring per classes. (D) Both datasets plotted without coloring in Radial Coordinates also yield in identical representations. (E) The *synthetic 1* and (F) the *synthetic 2* datasets plotted in Radial Coordinates with coloring per classes. (G) The *synthetic 1* dataset plotted using Shifted Paired Coordinates with and (I) without coloring per classes. (H) The *synthetic 2* dataset plotted using Shifted Paired Coordinates with and (J) without coloring per classes.

to classes, and thus see that we can distinguish their differences (see figures 5E and 5F). However, if we deactivate the coloring by classes, the result is identical for both datasets (see figure 5D).

Fortunately, other *GLC* techniques overcome this issue. For example, the Shifted Paired Coordinates avoid such confusion (see figure 5G-J). With this technique, it is not necessary to color by class in order to see the two patterns (see figures 5I and 5J).

6.3 Linear and Non-Linear Relationships in *GLC*

The *GLC-Vis* library allows overcoming one of the main issues of the Parallel Coordinates related to nonlinear feature detection. In this scenario, we use the *GLC-Frame* to explore how Parallel Coordinates and Collocated Paired Coordinates represent a nonlinear synthetic dataset. For this matter, we use a synthetic dataset with Gaussian noise consisting of 530 data items and 42 dimensions (see figure 6). After loading the dataset, we configure the Parallel Coordinates technique in *GLC-Frame* (see figure 6A), and two groups of well-defined lines are identified. However, obtaining a global understanding of the data is not easy.

The Collocated Paired Coordinates, on the other hand, use a better configuration to display the data items (see figure 6B). When configuring the Collocated Paired Coordinates technique in the *GLC-Frame* we realize that two circles of different sizes and positions are visible using fewer dimensions. Also, it is possible to see that both circles have an intersection in the left-bottom corner of the plot. This behavior is also visible in the Parallel Coordinates version, but its interpretation is more challenging because it involves analyzing a series of consecutive dimensions (from p17 to p34).

A similar situation occurs when we plot the data using Radial Coordinates (see figure 6C) and Paired Radial Coordinates (see figure 6D). Using Radial Coordinates, figure 6C shows two regions where one part of the items are visually separable and others are not. The clutter region corresponds to the dimensions that determine the intersection of both clusters. The Paired Radial Coordinates version handles this problem by eliminating the conflictive region and generating the expected groups (see figure 6D). Like the Collocated Paired Coordinates configuration (see figure 6B), this approach preserves the sizes of clusters but adds new information to put one group inside the other.

6.4 Cluster identification in *GLC*

Typical cluster structures are represented as groups of elements. In this scenario we aim to explore how different *GLC* techniques visually represent clusters. For this purpose, we use *GLC-Frame* to analyze a synthetic dataset consisting of 34 data items, classified in six classes, and ten dimensions, by configuring the Parallel Coordinates technique (see figure 7A), the Collocated Paired Coordinates technique (see figure 7B), and the Shifted Paired Coordinates technique (see figure 7C) views. In Shifted Paired Coordinates each next pair of coordinates is shifted by 1.

Figure 7 illustrates that clusters with different densities present significantly different visible structures on each representation. In Parallel Coordinates (see figure 7A), the clusters are relatively distinguishable, although the data items corresponding to *Class* 3 and *Class* 5 show overlap (partial occlusion). In Collocated Paired Coordinates (see figure 7B), the clusters can be easily distinguished and separated, leaving empty space for other data to be displayed without overlapping or occlusion. In Shifted Paired Coordinates (see figure 7C) all clusters are overlapped. On the other hand, Parallel Coordinates indicate linear correlation, as all lines are roughly horizontal. The Collocated Paired Coordinates show those lines as small groups located on a straight diagonal line which also resembles a correlation among clusters. Although in Shifted Paired Coordinates the clusters overlap causing almost total occlusion, the correlation between them can be observed.

6.5 Interacting with the *GLC*

In this example we load the well-known Cars (Auto MPG) dataset from the UCI Machine Learning repository [27] into the *GLC-Frame* to illustrate how visual analysis is supported by interactions.

Figure 8A shows the dataset represented using Parallel Coordinates with the original dimensions' order. To find a better visual classification of the dataset we reorder the axes resulting in figure 8B. It is worth mentioning that although the original order of axes seems to reveal a better visual classification in the Parallel Coordinates, this is not necessarily the case for all *GLC* representations. Some orders are better for some techniques than for others; finding the best arrangement will depend not only on the dataset and the task at hand but also on the representation itself. In fact, if we plot the dataset using Shifted Paired Coordinates, it is observed that the original order of axes seems to reveal a better visual classification in the Parallel Coordinates (see figure 8A) than in the Shifted Paired representation (see figure 8C). On the other hand, the reordering of axes seems to work better for the Shifted Paired Coordinates (see figure 8D) than for the Parallel Coordinates representation.

Finally, figures 8E, 8F, and 8G show the result of filtering the data by classes to visualize the three most dense classes in detail.

7 Discussion

GLC-Vis is a solution that provides a new set of visualization techniques allowing the user to explore n-D datasets. Thus, it is necessary to examine how well these representations work for long-standing issues in visual analysis.

To show the potential of the library in terms of the implemented techniques, we provide a web application called *GLC-Frame* that facilitates the analysis of various datasets using these techniques. By means of this application, different usage examples were carried out that show how different representations may or may not be better than others in different scenarios. We provide a dual view mechanism to show a dataset under two representations, a *NP-GLC* and a *P-GLC* technique, leaving to the user the analysis of both representations.

However, the library has a linear complexity time that restricts the number of elements (dimensions and items) to display. *GLC-Vis* is built on SVG elements, and this type of implementation reduces the computing performance. Although the canvas-based approach is better than the SVG-based approach, there is a trade-off between the effort to interact with the graphical elements and the number of them to render. In the case of our web application, the number of linked views is limited to two to show how to interact with *GLC* techniques. Adding other views requires a state management approach to keep consistency among them.

We also define interactions to facilitate the exploration. Using the *GLC-Frame*, it became clear that some techniques can handle occlusion better than others, and also that in some representations, clusters are visually detected better than in others.

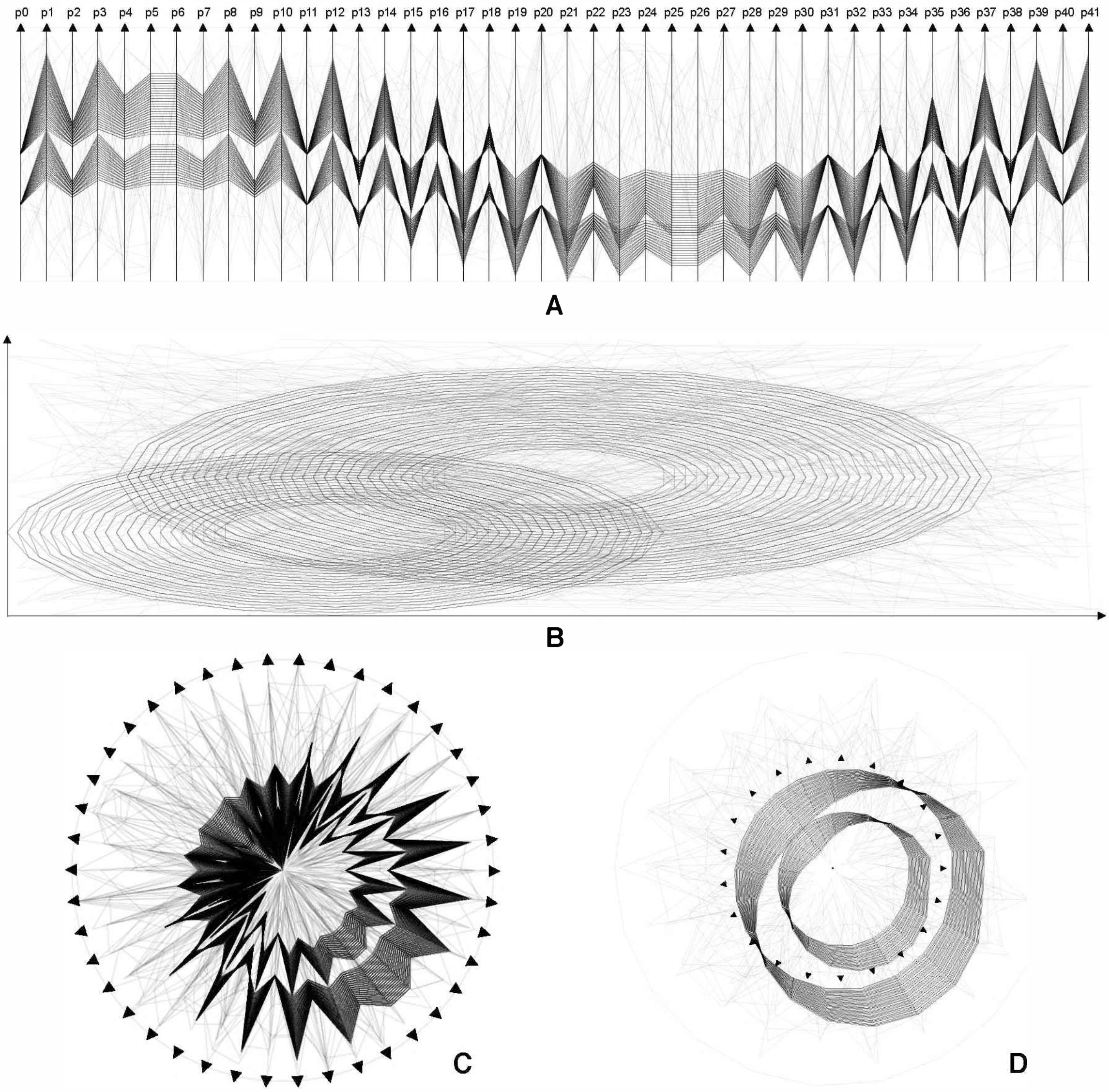


Figure 6: Visual analysis of non-linear structure in Non-Paired and Paired techniques. (A) and (C) display the Parallel Coordinates and Radial Coordinates respectively, and (B) and (D) their corresponding counterpart in Paired (Collocated Paired Coordinates and Paired Radial Coordinates).

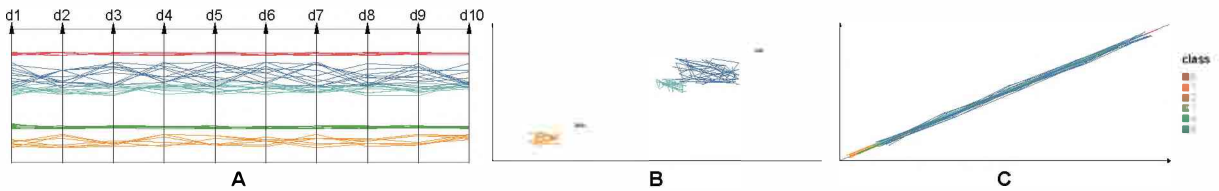


Figure 7: Cluster identification. A synthetic dataset represented using Parallel Coordinates (A), Collocated Paired Coordinates (B), and Shifted Paired Coordinates (C). Clusters with different densities present significantly different visible structures on each representation.

Following, we discuss how *GLC* overcomes some issues of traditional techniques in terms of the most relevant aspects mentioned in section 6.

Occlusion: We analyze a synthetic dataset under four different representations: Parallel Coordinates, Radial Coordinates, Radial Dynamic Coordinates, and Shifted Paired Coordinates. In this case, we see that the Parallel Coordinates show total occlusion of two groups of the dataset and that this is also true for the Radial Coordinates. However, this does not occur in the Radial Dynamic Coordinates and in the Shifted Paired Coordinates. In the last two, the groups are dis-

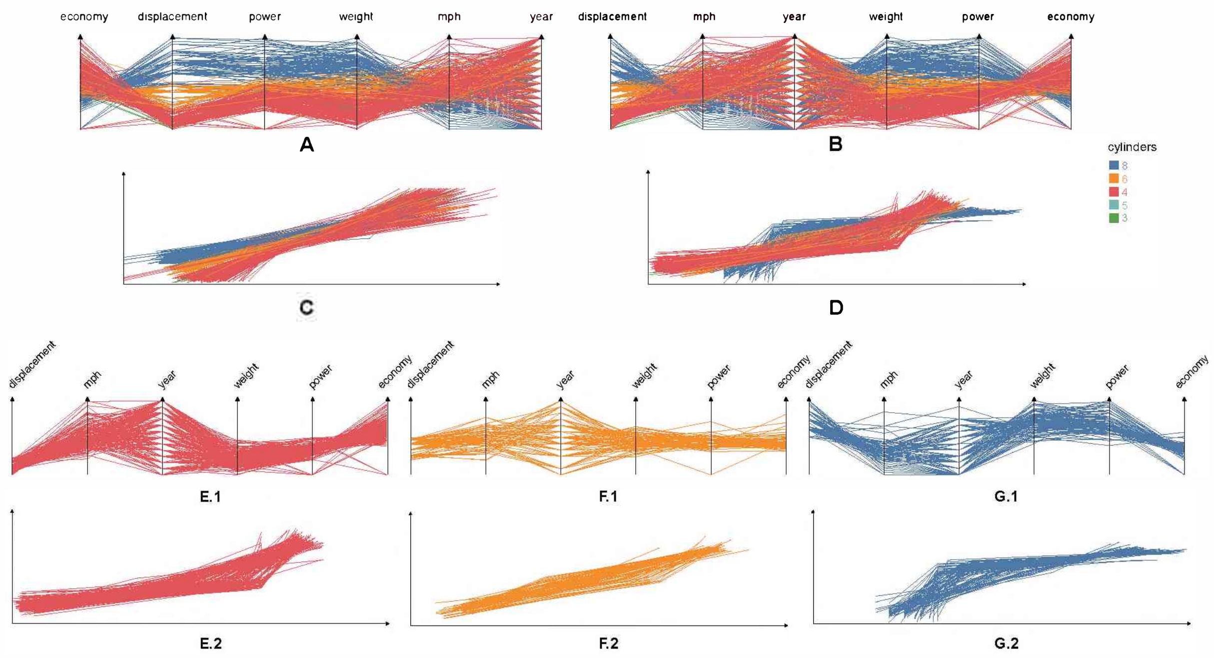


Figure 8: The Cars Dataset [27] represented with the original dimension ordering with Parallel Coordinates (A) and Shifted Paired Coordinates (C). After reordering the Shifted Paired Coordinates (D) seems to reveal a better visual classification than the Parallel Coordinates (B). Then, the user selects the data items with 4 cylinders (E.1 and E.2), 6 cylinders (F.1 and F.2), and 8 cylinders (G.1 and G.2).

tinguishable over almost the entire range of attributes. In the Radial Dynamic Coordinates there is a problem of occlusion at the origin of coordinates and in the Shifted Paired Coordinates, in some attributes but still allows to differentiate the different groups. In both cases, it is also possible to obtain more insight into the behavior by using transparency to plot the different data items.

On the other hand, the same dataset was also plotted with the items being both randomly distributed and sorted by groups but in different order. In these cases, it can be seen that both in Parallel Coordinates and in Radial Coordinates, what is plotted depends on the order. However, this is not the case in Radial Dynamic Coordinates and in Shifted Paired Coordinates. In the latter, the groups are differentiated in the same way as in the case where the dataset is ordered. This is a great advantage since the groups can be identified without the need to perform sorting on the data.

Cluster identification: We analyze a synthetic dataset under three different representations: Parallel Coordinates, Collocated Paired Coordinates, and Shifted Paired Coordinates. In our analysis of different *GLC* representations, we found *P-GLC* encouraging alternatives for detecting and analyzing clusters. In Parallel Coordinates and Collocated Paired Coordinates, the clusters are distinguishable, however, in the last case, we can appreciate more emptiness in the coordinate plane than in Parallel Coordinates. This allows to display more data and, therefore, to distinguish them better. It should be noted that, in all three cases, the correlation between clusters can be observed. These techniques allow us to visualize the proximity among clusters at the inter-cluster level and the intra-cluster level. However, in the Collocated Paired Coordinates, the variance between clusters can be compared in an estimated way.

8 Conclusions and Future Work

In this paper, we introduce the *GLC-Vis* and *GLC-Frame*, which support the generation and manipulation of GLC without coding requirement. This characteristic allows for a broader base of users who are interested in exploring GLC techniques and do not have the expertise to deal the details of their implementation. Having the library and the web application tool at our disposal allowed us to show different datasets and compare how different GLC techniques might be more convenient than others in a given situation. We detected how, with some representations, it is possible to solve the occlusion problem or identify clusters, as detailed in section 6. Undoubtedly, the GLC techniques have great potential in the visual analysis of n-D data, and having the *GLC-Vis* library that allows us to make them available is a great step towards taking advantage of their potential.

As future work, we propose to design and develop new interactions to manipulate the *P-GLC* space. This is a relevant unexplored area for improving the use of these techniques. Also, we study and plan the possibility to add the *GLC-L* and its variants to complete the full spectrum of GLC. However, there are specific implementations of these particular GLC techniques [19, 20] available as desktop application. It is also possible to incorporate different axes reordering techniques into the library. The reordering techniques have been extensively studied in the Parallel Coordinates [21, 28] and have recently been studied in Radial Coordinates [29, 30], but remain an unexplored topic for the other techniques. Finally, as the next step, we plan to conduct a systematic evaluation of the presented library and the web application.

Competing interests

The authors have declared that no competing interests exist.

Funding

This work was partially supported by PGI 24/N048 and PGI 24/ZN38 research grants from the Secretaría General de Ciencia y Tecnología, Universidad Nacional del Sur (Argentina), and by 28720210100824CO (PIBAA) granted by National Council for Scientific and Technical Research (CONICET).

Authors' contribution

Leandro Luque: Conceptualization, Methodology, Software, Writing- Original draft preparation. **Antonella Antonini:** Conceptualization, Methodology, Software, Writing-Original draft preparation . **Maria Lujan Ganuza:** Writing-Reviewing and Editing, Funding acquisition. **Silvia Castro:** Writing- Reviewing and Editing, Funding acquisition.

References

[1] G. Dzemyda, O. Kurasova, and J. Zilinskas, "Multidimensional data visualization," *Methods and applications series: Springer optimization and its applications*, vol. 75, no. 122, pp. 10–5555, 2013.

[2] P. C. Wong and R. D. Bergeron, "30 years of multidimensional multivariate visualization," in *Scientific Visualization, Overviews, Methodologies, and Techniques*, (USA), p. 3–33, IEEE Computer Society, 1994.

[3] D. Borland, W. Wang, J. Zhang, J. Shrestha, and D. Gotz, "Selection bias tracking and detailed subset comparison for high-dimensional data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 429–439, 2019.

[4] S. Liu, D. Maljovec, B. Wang, P.-T. Bremer, and V. Pascucci, "Visualizing high-dimensional data: Advances in the past decade," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 3, pp. 1249–1268, 2016.

[5] G. M. Draper, Y. Livnat, and R. F. Riesenfeld, "A survey of radial methods for information visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 5, pp. 759–776, 2009.

[6] A. Inselberg, "A survey of parallel coordinates," in *Mathematical Visualization*, pp. 167–179, Springer, 1998.

[7] J. Heinrich and D. Weiskopf, "State of the art of parallel coordinates," in *Eurographics (STARs)* (M. Sbert and L. Szirmay-Kalos, eds.), pp. 95–116, 2013.

[8] B. Kovalerchuk, *Visual Knowledge Discovery and Machine Learning*, vol. 144. Springer Publishing Company, Incorporated, 2018.

[9] B. Kovalerchuk and V. Grishin, "Adjustable general line coordinates for visual knowledge discovery in n-d data," *Information Visualization*, vol. 18, no. 1, pp. 3–32, 2019.

[10] L. E. Luque, M. L. Ganuza, A. S. Antonini, and S. M. Castro, "npglc-vis library for multidimensional data visualization," in *Conference on Cloud Computing, Big Data & Emerging Topics*, pp. 188–202, Springer, 2021.

[11] M. O. Ward, G. Grinstein, and D. Keim, *Interactive Data Visualization: foundations, techniques, and applications*. CRC Press, 2010.

[12] C. Tominski and H. Schumann, *Interactive Visual Data Analysis*. CRC Press, 2020.

[13] J. H. Claessen and J. J. Van Wijk, "Flexible linked axes for multivariate data visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2310–2316, 2011.

[14] B. Kovalerchuk, "Visualization of multidimensional data with collocated paired coordinates and general line coordinates," in *Visualization and Data Analysis 2014*, vol. 9017, p. 90170I, International Society for Optics and Photonics, 2014.

[15] P. T. Inc., "Collaborative data science," 2015.

[16] M. Bostock, V. Ogievetsky, and J. Heer, "D^3 data-driven documents," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[17] W. J. Raseman, J. Jacobson, and J. R. Kasprzyk, "Parasol: An open source, interactive parallel coordinates library for multi-objective decision making," *Environmental modelling & software*, vol. 116, pp. 153–163, 2019.

[18] R. McDonald and B. Kovalerchuk, "Lossless visual knowledge discovery in high dimensional data with elliptic paired coordinates," in *2020 24th International Conference Information Visualisation (IV)*, pp. 286–291, 2020.

[19] L. Huber, B. Kovalerchuk, and C. Recaido, "Visual knowledge discovery with general line coordinates," *arXiv preprint arXiv:2305.18429*, 2023.

[20] L. Huber and B. Kovalerchuk, "No-code platform for visual knowledge discovering in general line coordinates: Dv 2.0," in *2023 27th International Conference Information Visualisation (IV)*, pp. 316–322, IEEE, 2023.

[21] M. Blumenschein, X. Zhang, D. Pomerenke, D. A. Keim, and J. Fuchs, "Evaluating reordering strategies for cluster identification in parallel coordinates," *Computer Graphics Forum*, vol. 39, no. 3, pp. 537–549, 2020.

[22] W. Peng, M. O. Ward, and E. A. Rundensteiner, "Clutter reduction in multi-dimensional data visualization using dimension reordering," in *IEEE Symposium on Information Visualization*, pp. 89–96, IEEE, 2004.

[23] L. F. Lu, M. L. Huang, and J. Zhang, "Two axes reordering methods in parallel coordinates plots," *Journal of Visual Languages & Computing*, vol. 33, pp. 3–12, 2016.

[24] A. O. Artero, M. C. F. de Oliveira, and H. Levkowitz, "Enhanced high dimensional data visualization through dimension reduction and attribute arrangement," in *Tenth International Conference on Information Visualisation (IV'06)*, pp. 707–712, IEEE, 2006.

[25] H. Makwana, S. Tanwani, and S. Jain, "Axes reordering in parallel coordinate for pattern optimization," *International Journal of Computer Applications*, vol. 40, no. 13, pp. 43–48, 2012.

[26] Preact, "Preact," 2015. Accessed on 05.05.2022.

[27] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. Accessed on 2022-04-12.

[28] A. Tyagi, T. Estro, G. Kuenning, E. Zadok, and K. Mueller, "Pc-expo: A metrics-based interactive axes reordering method for parallel coordinate displays," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[29] A. Khalid and I. I. Kamsani, "Star coordinate dimension arrangement using euclidean distance and pearson correlation," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 12, no. 1, pp. 348–355, 2018.

[30] M. Rubio-Sánchez, D. J. Lehmann, A. Sanchez, and J. L. Rojo-Álvarez, "Optimal axes for data value estimation in star coordinates and radial axes plots," *Computer Graphics Forum*, vol. 40, no. 3, pp. 483–494, 2021.

Citation: L. Luque, A. Antonini, M. L. Ganuza and S. Castro. *GLC-Frame: A Framework and Library for Exploration of Multidimensional Data with General Line Coordinates.* Journal of Computer Science & Technology, vol. 24, no. 1, pp. 14-28, 2024.

DOI: 10.24215/16666038.24.e02

Received: August 24, 2023 **Accepted:** December 5, 2023.