POLYTECHNIQUE MONTRÉAL

LE GÉNIE EN PREMIÈRE CLASSE

INF8808 – Visualisation de données

Lab 4

Authors:

Adrien Dessinges and Antoine Béland

Teacher's assistant

Olivia Gélinas

Winter 2020

Department of computer and software engineering

1 Objectives

The aim of this lab is to make a bar chart and a chord diagram from open data from a JSON file. The user will be able to update which data is used by the bar chart and will be able to select the elements to highlight in the bar chart.

Before starting this lab, we recommend reading Chapter 11 of Scott Murray's book [1].

2 Introduction

A bar chart is a graph representing data with bars. The height of each bar is used to encode the value it represents. It's one of the simplest graphs to draw and allows you to easily compare across multiple categories. The chord diagram, on the other hand, is much more complex and allows you to visualize links (represented by chords) between several categories (the arcs in a circle). The width of a chord makes it possible to encode the value starting from one category and heading towards another. The length of an arc, which represents a category, encodes the accumulation of the values of the incoming and outgoing chords of this same category.

In this lab, you will have to make the two types of graphs mentioned above using data on the number of trips made between ten BIXI stations during the month of August 2015 in Montreal. This data comes from the open data portal of BIXI Montreal. The data is grouped in the file "`bixi-destinations.json`" located in the folder "`data`".

The JSON file provided consists of a list containing ten elements, corresponding to one element per BIXI stations. Each of these elements has two properties: "`name`" and "`destinations`". The "`name`" property corresponds to the name of the departure station while the "`destinations`" property is a list indicating the number of trips that have been made to arrival stations from the departure station. For the sake of clarity, Figure 1 shows the format of an item in the list in the JSON file. In this example, the departure station is "Mont-Royal / Clark", 71 trips return to the departure station while 22 others go to the "Pontiac / Gilford" station.

```
{
  "name": "Mont-Royal / Clark",
  "destinations": [
    {
      "name": "Mont-Royal / Clark",
      "count": 71
    },
    {
      "name": "Pontiac / Gilford",
      "count": 22
    }
  ]
}
```

Figure 1: Example of the format used for the elements in the list of the JSON file

3 What to do

For this lab, you will have to complete the JavaScript code for the two graphs which will represent the BIXI data. The bar chart will be displayed in the "Detailed view" tab while the chord diagram will be displayed in the "General view" tab.

The subsections below present the different parts that will need to be completed for this lab. It should be noted that it is necessary to carry out the first part of this lab (data preprocessing) before carrying out the following two parts. Make sure to complete the different "`TODO`" sections found in the files located in the "`assets/scripts`" folder.

3.1 Data preprocessing

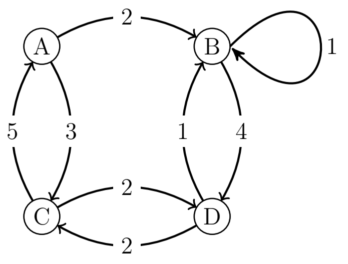
For this first part, you will need to define the different scales that will be used by the bar graph. Also, you will need to create the adjacency matrix needed to draw the chord diagram.

An adjacency matrix is a matrix of size $n \times n$ where n is the number of parameters to consider. In our case, this matrix will be modeled by an array of arrays (`Array[Array]`) of size $n \times n$ where n is the number of BIXI stations considered. An element $m_{i,j}$ of the matrix corresponds to the number of trips leaving from station i and going to station j. Note that in general, $m_{i,j} \neq m_{j,i}$. As an example, Figure 2 shows an equivalent representation between an adjacency matrix (2a) and an oriented graph (2b) in order to illustrate the transition from one form to another.

Unlike the last two labs, you will not have to convert the data provided. Indeed, since the data comes from a file in JSON format, it is not necessary to perform type conversions since this information is kept in the storage format unlike in CSV.

$$\begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 2 & 3 & 0 \\ B & 0 & 1 & 0 & 4 \\ C & 5 & 0 & 0 & 2 \\ D & 0 & 1 & 2 & 0 \end{array}$$

(a) Adjacency matrix



(b) Directed graph

Figure 2: Example of two equivalent representations

To carry out this part, you will have to complete the file "**`1-preproc.js`**" located in the folder "`assets/scripts`". More specifically, you will have to complete the following:

- Specify the domain of the color scale (function "`domainColor`");
- Specify the domain of the axes x and y (functions "`domainX`" and "`domainY`");
- Create the necessary adjacency matrix (function "`getMatrix`");
- Calculate the total number of journeys made (function "`getTotal`").

3.2 Bar chart

For this second part, you will have to make the bar graph which will display the number of journeys to other stations for a specified departure station. You will have to complete the functions found in the file "**`2-bar-chart.js`**" so that everything is functional. The following subsections detail the steps that will need to be completed to complete the diagram.

3.2.1 Creating the diagram

The first step consists in creating the bar graph for the data having as departure station "De la Commune / Place Jacques-Cartier" (by default). To do this, you will have to draw the bars of the graph for each of the destination stations, making sure that the color of the bar corresponds to the color of the associated station. Also, you will have to draw the axes x and y of the graph. For the sake of clarity, Figure 3 shows what the graph obtained after this step should look like.

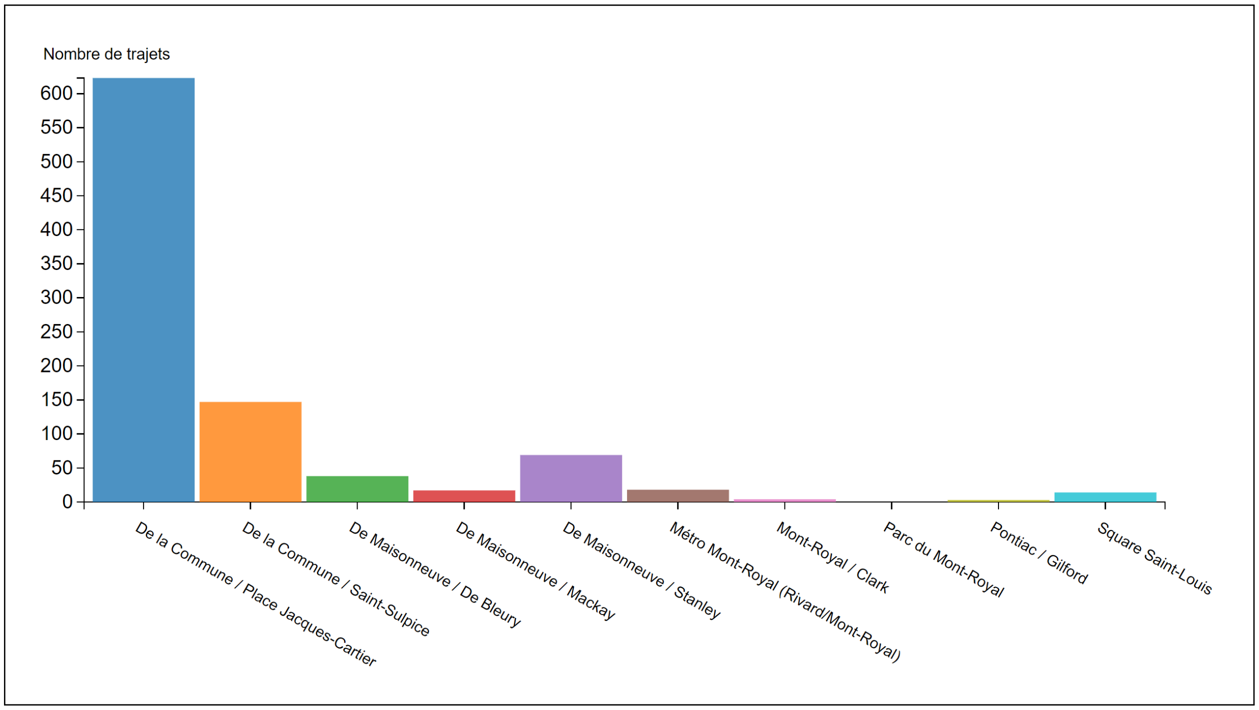


Figure 3: Bar chart using "De la Commune / Place Jacques-Cartier" as departure station

For the creation of the axes, you can use the classes "`axis x`" and "`axis y`" with the groups responsible for drawing the axes x and y respectively.

To complete this part, you will need to do the following:

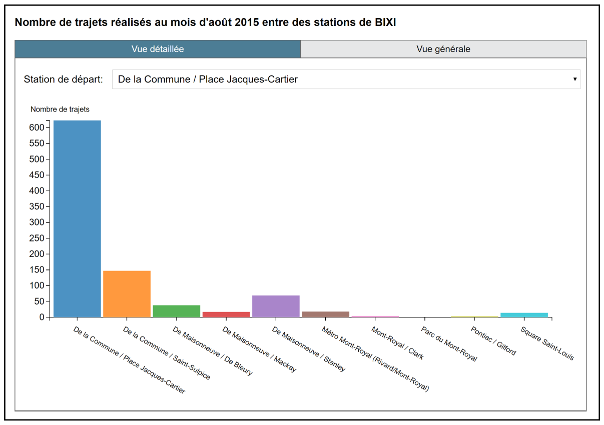
- Draw the axes x and y associated with the graph (function "`createAxes`");
- Draw the bars of the graph ("`createBarChart`" function).

Be careful!

Make sure to indicate a title in the y axis of the graph (e.g. "Number of trips"). Also, make sure that the orientation of the station names on the x axis is 30 degrees (see Figure 3).

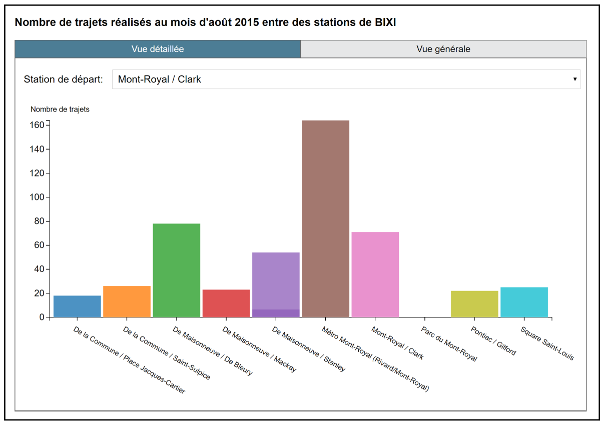
3.2.2 Transition

The second step is to insert a transition when the data bound to the graph is updated. When a new departure station is selected by the user via the drop-down list, the data in the diagram must be updated, so it uses the data associated with the chosen departure station. In this sense, a transition lasting one second must be made to update the height of the bars as well as the graduation of the axis of y. Figure 4 also illustrates the expected display for the departure stations "De la Commune / Place Jacques-Cartier" (4a) and "Mont-Royal / Clark" (4b).


Nombre de trajets réalisés au mois d'août 2015 entre des stations de BIXI

Station de départ: De la Commune / Place Jacques-Cartier

(a) De la Commune / Place Jacques-Cartier


Nombre de trajets réalisés au mois d'août 2015 entre des stations de BIXI

Station de départ: Mont-Royal / Clark

(b) Mont-Royal / Clark

Figure 4: Bar chart using different data

To achieve this functionality, you will have to complete the "`transition`" function.

3.2.3 Tooltip

The third step to perform consists in adding a tooltip when a bar of the graph is hovered. To do this, you must complete the text that must be displayed during this overview, indicating the number of journeys associated with the station and the percentage corresponding to the number of journeys arriving at this station compared to the total number of journeys departing from the departure station. The rendering of the tooltip should be similar to Figure 5 (data using "De Maisonneuve / Stanley" as the starting station). Note that the percentage must be formatted using the "`formatPercent`" function that is provided.

To implement this feature, you will have to complete the "`getToolTipText`" function.

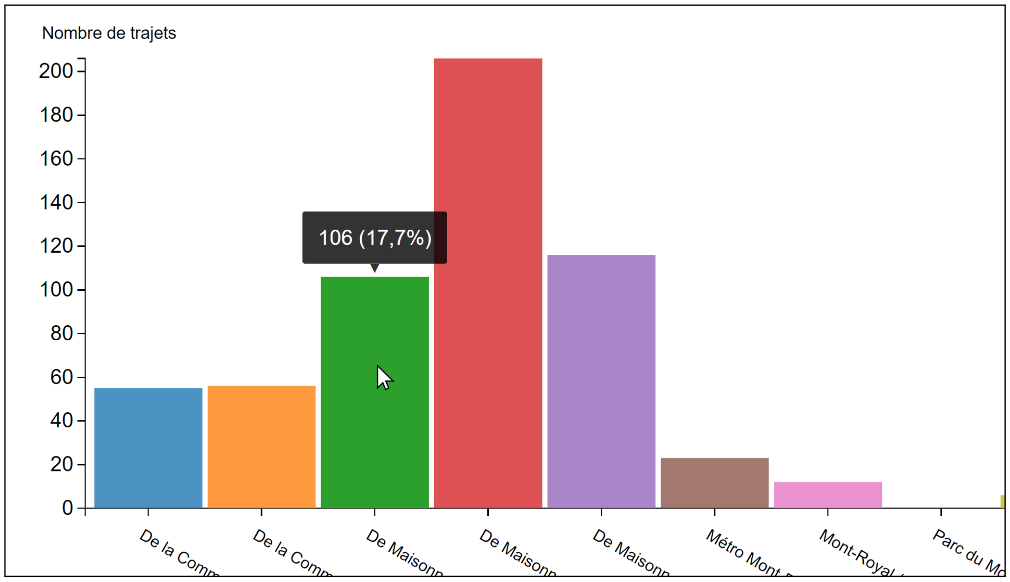


Figure 5: Tooltip that is displayed when a bar is hovered

3.3 Chord diagram

For this third and last part, you will have to create the chord diagram which will allow you to visualize the number of journeys made between the ten BIXI stations. In order for everything to work properly, you will need to complete the functions found in the file "**`3-chord-diagram.js`**". To help you, you can take a look at this code sample to create this type of graph using D3. The following subsections describe the steps to perform to obtain the requested diagram.

3.3.1 Creating groups

The first step is to create the different groups used by the diagram. A group corresponds to an arc and is associated with a BIXI station. Each group must be colored in the color of the station associated with it and must indicate the name of the station in white. In addition, the station names should follow the curvature of the group's arc using a "`textPath`". In the event that a BIXI station name is too long, it must be truncated so that the text can enter the group (necessary for the stations "Métro Mont-Royal[Rivard / Mont-Royal]" and "Pontiac /Gilford"). Figure 6 illustrates the display which is expected once the creation of the groups is completed.

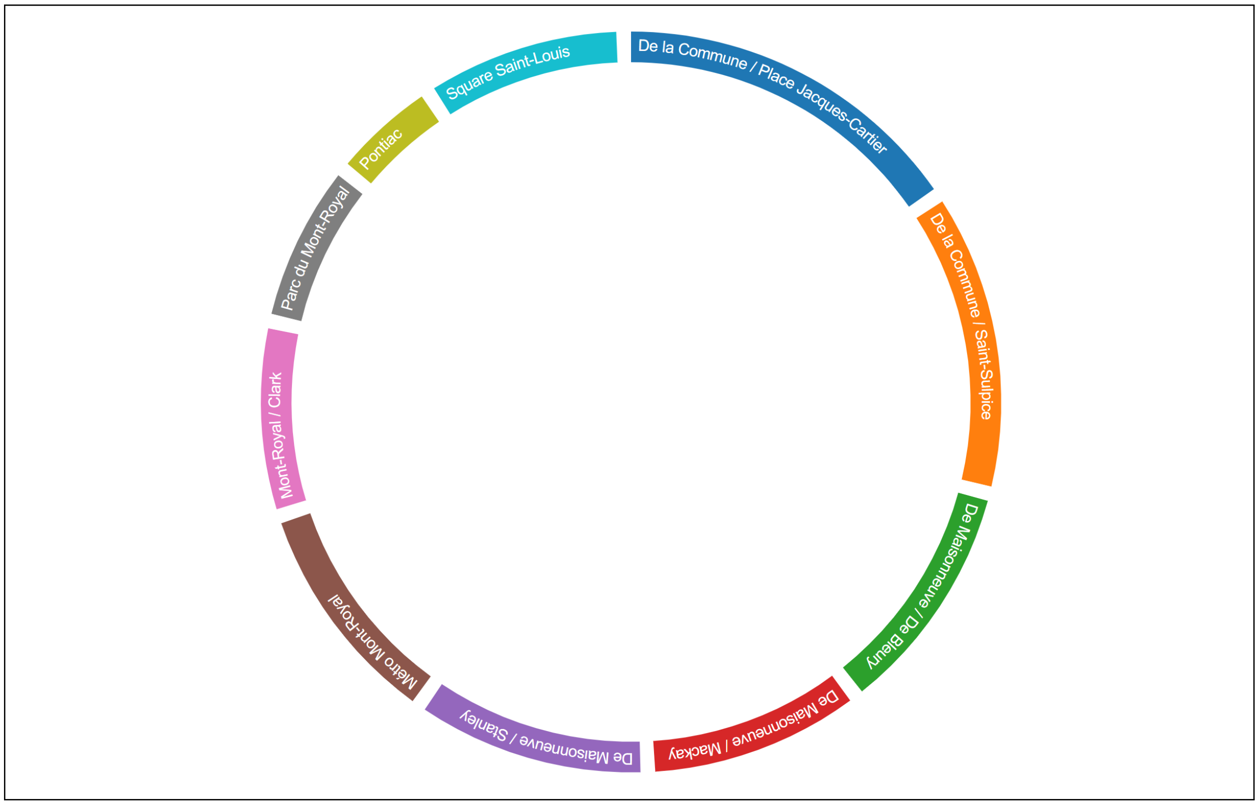


Figure 6: Chord diagram groups

When a group is over the mouse, it will display a tooltip using the tag "`title`". This tooltip must indicate the name of the station as well as the percentage of journeys starting from this station in relation to the total number of journeys. Figure 7 also shows the format of the tooltip that is expected. Make sure to use the "`formatPercent`" function to correctly format the percentage.

To perform this step, you will have to complete the "`createGroups`" function.

3.3.2 Creating chords

The second step is to create the diagram chords that will represent the number of journeys between BIXI stations. Each of the chords must be colored with the color of the station with the most departures between the two linked stations and must have an opacity of 80%. Figure 8 shows the graph that should be obtained once the creation of the chords is complete.

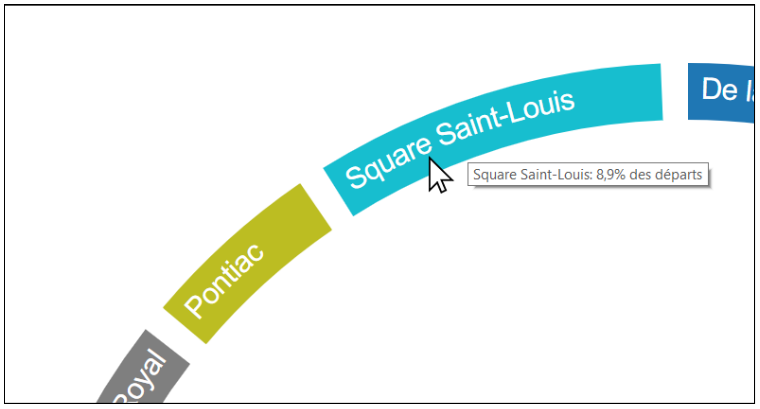


Figure 7: Tooltip to show when a group is hovered

When a chord is passed over with the mouse, the opacity of the latter will have to pass to 95 % and a tooltip will have to be displayed thanks to the tag "`title`". This tooltip must indicate the percentage of the number of journeys made in one direction as in the other. For clarity, Figure 9 shows what the requested tooltip should look like. Make sure to format the percentages using the "`formatPercent`" function.

To do this step, you will have to complete the function "`createChords`".

3.3.3 Selection of a group to display

The third step is to display only the chords of a group in particular when the latter is over the mouse in order to facilitate the readability of the graph. In this sense, when a group is overflown by the mouse, the incoming and outgoing chords of this group will have to be displayed with an opacity of 80 % while the opacity of all the others will have to be modified to 10 %. This selection should be kept as long as the mouse is inside the circle of the graph. Figure 10 illustrates what the selection of a particular station should look like. When the mouse leaves the circle, the diagram display must be reset to the default, as illustrated in Figure 8.

To do this step, you will have to complete the function "`initializeGroupsHovered`".

i **Advice**

To make this task simpler, you can complete this feature using CSS classes.

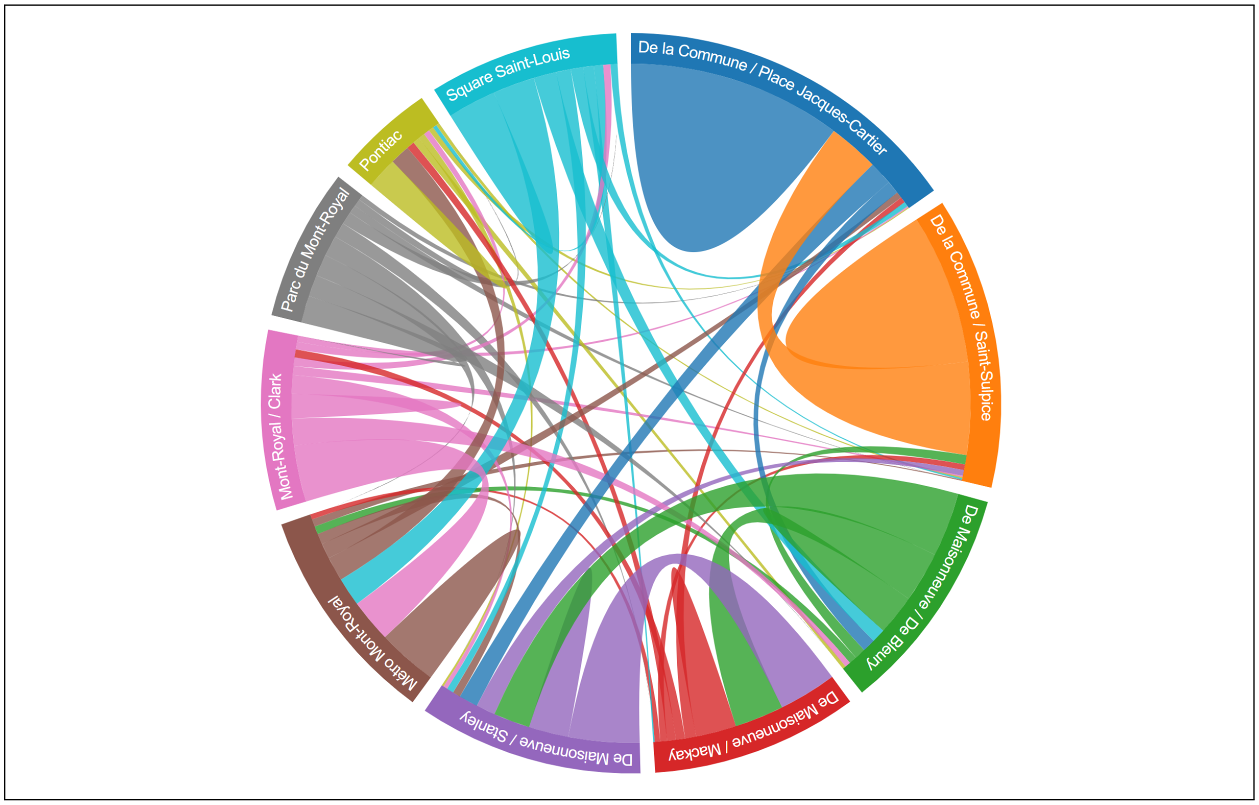
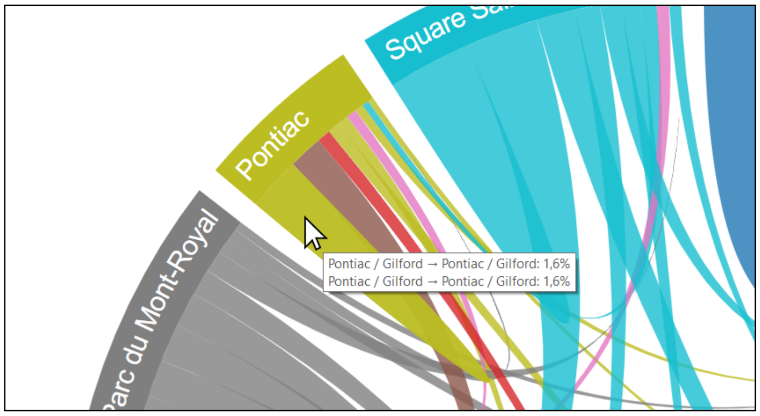


Figure 8: Chord diagram using the provided data



Pontiac / Gilford → Pontiac / Gilford: 1,6%
Pontiac / Gilford → Pontiac / Gilford: 1,6%

Figure 9: Tooltip to show when a chord is hovered

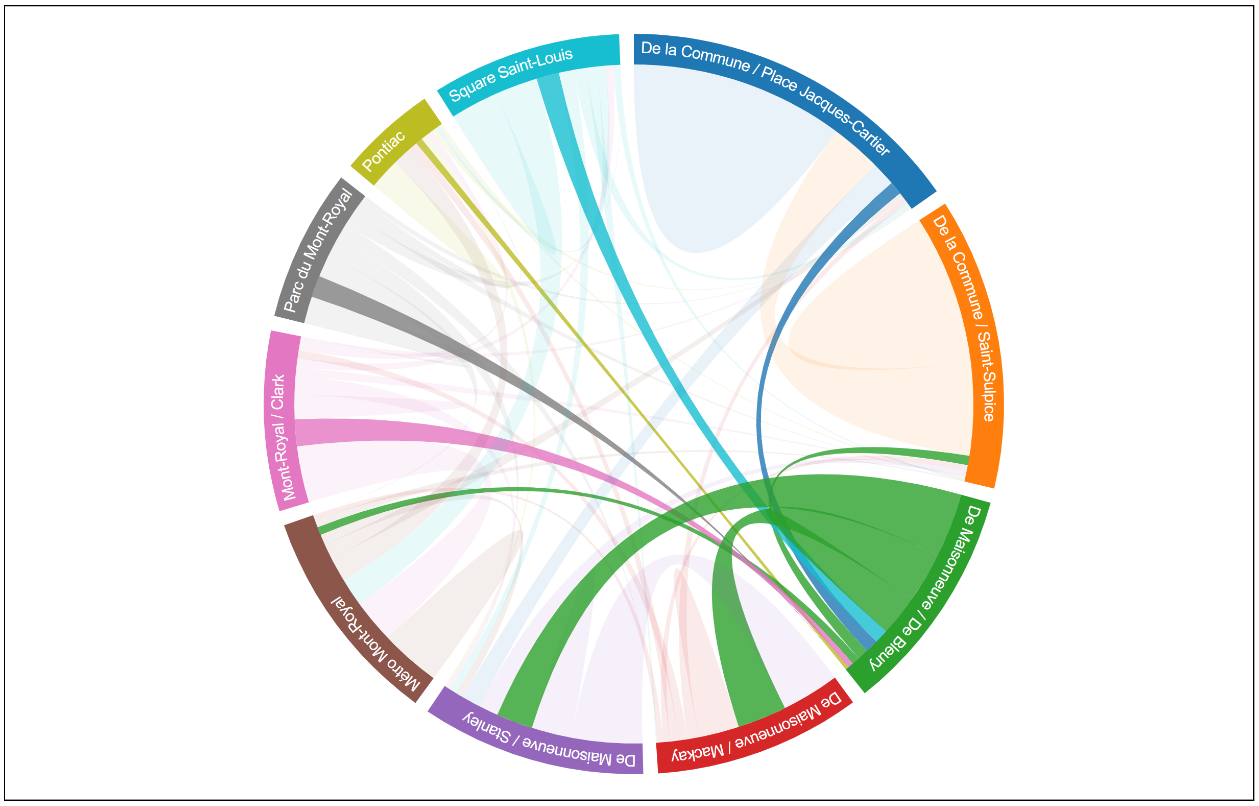


Figure 10: Display of information from station "De Maisonneuve / De Bleury"

4 Submission

Here are the instructions for submitting this lab:

1. You must place your project code in a compressed ZIP file called "`TP4_studentId1_studentId2_studentId3.zip`".
2. The work must be submitted before **11:59 PM**, **February 27 2020** on Moodle.

5 Evaluation

Overall, you will be assessed on the respect and proper functioning of the requested requirements. More specifically, the correction scale is as follows:

Requirement	Points
Data preprocessing	3
Bar chart	
Creating the diagram	4
Transition	1
Tooltip	1
Chord diagram	
Creating groups	4
Creating chords	3
Selection of a group to display	3
Code quality and clarity	1
Total	20

This lab is worth **10%** of the course grade.

References

[1] S. Murray, *Interactive Data Visualization for the Web: An Introduction to Designing with D3*. O'Reilly Media Inc., 2013.