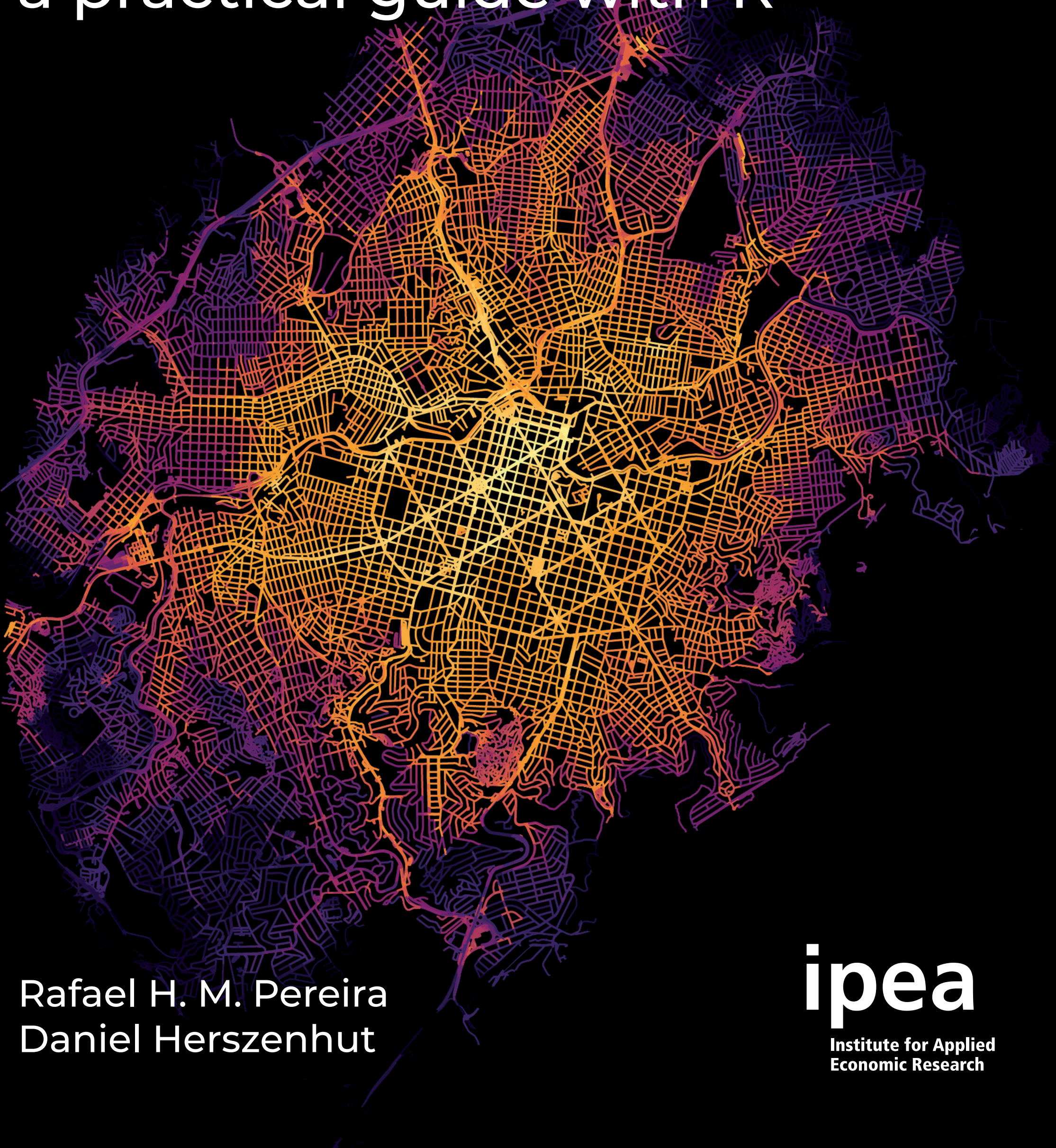
Introduction to Urban Accessibility

a practical guide with R



Rafael H. M. Pereira
Daniel Herszenhut

ipea
Institute for Applied Economic Research

Introduction to Urban Accessibility

a practical guide with R

Rafael H. M. Pereira
Daniel Herszenhut

ipea

ipea **Institute for Applied Economic Research**

A public foundation affiliated to the Ministry of Planning and Budget, Ipea provides technical and institutional support to government actions – enabling the formulation of numerous public policies and programs for Brazilian development – and makes research and studies conducted by its staff available to society.

Ombudsman: http://www.ipea.gov.br/Ouvidoria
URL: http://www.ipea.gov.br

Introduction to Urban Accessibility

a practical guide with R

Rafael H. M. Pereira
Daniel Herszenhut

ipea

Rio de Janeiro, 2023

P436i Pereira, Rafael H. M.

Introduction to urban accessibility : a practical guide with R / Rafael H. M. Pereira, Daniel Herszenhut. – Rio de Janeiro : Ipea, 2023.

152 p. : il.

Inclui referências bibliográficas.

ISBN: 978-65-5635-065-3

1. Acessibilidade Arquitetônica. 2. Acessibilidade Urbana 3. Meios de Transporte. 4. Planejamento Urbano. 5. Brasil. I. Herszenhut, Daniel. II. Título.

CDD 362.4

Catalographic card drafted by Andréa de Mello Sampaio CRB-1/1650

How to cite:

PEREIRA, Rafael H. M.; HERSZENHUT, Daniel. **Introduction to urban accessibility**: a practical guide with R. Rio de Janeiro: Ipea, 2023. 152 p. ISBN: 978-65-5635-065-3. DOI: http://dx.doi.org/10.38116/9786556350653.

Ipea publications are available for free download in PDF (all) and EPUB (books and periodicals). Access: https://repositorio.ipea.gov.br/.

The opinions expressed in this publication are of exclusive responsibility of the authors, not necessarily expressing the official views of the Institute for Applied Economic Research and the Ministry of Planning and Budget.

CONTENTS

PREFACE[1]

Accessibility is the ease with which people can reach places and opportunities such as jobs, health and education services, cultural activities, green areas, etc. The accessibility conditions of a city or neighborhood depend on the efficiency and connectivity of the transport network and on the proximity between people and activities. The topic of accessibility has been receiving increased attention from transportation agencies, funding institutions, decision makers and researchers in the fields of urban and transport planning (Papa et al., 2015; Boisjoly and El-Geneidy, 2017). In the last few years, we have seen a growing number of scientific papers (Miller, 2018; Van Wee, 2021) and books (Levine, Grengs and Merlin, 2019; Levinson and King, 2020) that deepen our understanding of urban accessibility issues. However, there are currently no books or articles that serve simultaneously as introductory material to the subject and as a practical manual to teach computational methods to calculate and analyze accessibility data. The lack of this type of material helps to explain, at least in part, why several transportation agencies and analysts still face challenges to incorporate accessibility issues and indicators into the day-to-day planning and research practice (Silva et al., 2017; Büttner, 2021).

The aim of this book is to equip its readers with the fundamental concepts, the data analysis skills and the processing tools needed to perform urban accessibility analyses and transportation projects impact assessments. The book was written with the problems faced by public managers, policy makers, students and researchers working on urban and transportation planning in mind. Hence, the book is essentially practical. All the material in the book is presented with reproducible examples using open datasets and the R programming language.[2]

[1] This book was written under a partnership between the Institute for Applied Economic Research (Instituto de Pesquisa Econômica Aplicada – Ipea) and the Secretariat of Mobility and Regional and Urban Development (Secretaria Nacional de Mobilidade e Desenvolvimento Regional e Urbano – SMDRU) of the Ministry of Regional Development (Ministério do Desenvolvimento Regional – MDR) of the Brazilian federal government. The authors are grateful for the comments and suggestions from Lucas Mation. The authors would also like to thank the contributions of Anna Grigolon, Benjamin Büttner and Aaron Nichols for proofreading the English version of the book. Any remaining errors are the sole responsibility of the authors.

[2] This book assumes the reader has a basic knowledge of the R programming language. If you want to familiarize yourself with it, we recommend the books Damiani et al. (2022), Wickham and Grolemund (2017) and Lovelace, Nowosad and Muenchow (2019).

1 BOOK STRUCTURE

This book is divided into 5 sections, as follows.

1) *Introduction to urban accessibility*: the first section presents the concept of urban accessibility, clarifies the differences between accessibility and mobility and presents the main indicators used in the literature to measure urban accessibility.

2) *How to measure urban accessibility*: the second section teaches how to calculate urban accessibility estimates in R using open data and the `{r5r}` and `{accessibility}` packages and how to visualize the results with maps and charts.

3) *Public transport data*: the third section presents the General Transit Feed Specification (GTFS) specification of public transport data and shows how to work and analyze GTFS data using the `{gtfstools}` package.

4) *Impact assessment of transportation projects*: the fourth section brings a case study to illustrate how the combined knowledge of previous chapters can be used to assess the impact of transportation policies on urban accessibility conditions.

5) *Data from the Access to Opportunity Project*: finally, the fifth section shows how to download and visualize the data produced and made available by the Access to Opportunities Project (AOP), which brings detailed information on land use and accessibility patterns in Brazilian cities.

Reproducing the book in your computer

This book has been written with the Quarto publishing system. All the code used to prepare and publish it online can be found in this repository. To reproduce the book in your local machine, you must first download its source code. The easiest way to do this is by cloning the repository with Git, or, alternatively, by manually downloading it.[3] If you choose the second approach, you must also unzip the contents of the `.zip` file to a new folder.

To render the book, you must have Quarto installed in your computer. Reproducing the chapters also requires the `{renv}` R package, which manages the book dependencies.

After installing book dependencies, you can render each chapter as you would normally render any Quarto/Rmarkdown file. To render the entire book,

[3] To manually download the source code, please use the following link: https://github.com/ipeaGIT/intro_access_book/archive/refs/heads/main.zip.

use the following command (please note that currently both the Portuguese and the English books are rendered):

```
source("bilingual_render.R")
```

For more details on how to install the book dependencies and on how to run the book content locally, please see the installation instructions on the repository main page.

Running the book examples in the cloud with binder

A binder is a tool that allows one to use a browser, such as Chrome and Firefox, to run code in the cloud. The book is set up so that its code can be run using a server published by MyBinder.[4] Please note that MyBinder sessions are limited to 2 GB of RAM. This restriction can prevent chapter 6 from running properly. If you use binder, we suggest that you do not attempt to render the entire book with Quarto, as shown above.

[4] Available at: https://mybinder.org/v2/gh/ipeaGIT/intro_access_book/HEAD?urlpath=rstudio. After a few moments, an RStudio Cloud session will start running in your browser. This session includes all the files and data needed to run the code.

SECTION 1

INTRODUCTION TO URBAN ACCESSIBILITY

The purpose of this section is: i) to present the concept of urban accessibility, clarifying the difference between accessibility and mobility; and ii) to present an overview of the main indicators used to measure accessibility levels.

How many jobs can a person in a given location reach within an hour of travel using public transport? How long does it take for this person to get to the health center or the school closest to her home? The answers to these questions directly depend on local transport and urban development policies. These policies determine the urban accessibility conditions in each city – that is, the ease with which individuals can access opportunities such as employment, health and education services, cultural and leisure activities, among other types of activities. Accessibility, therefore, is simultaneously a result of the connectivity and performance of transport systems and the organization of cities in terms of the spatial distribution of their population, economic activities, and public services. Moving to accessibility-focused transport planning can help promote more sustainable and inclusive urban development. Throughout this section, we will look at the concept of urban accessibility in more detail, show why this concept is important to understand how cities function and present the indicators most commonly used to measure accessibility levels.

CHAPTER 1

1 WHAT IS ACCESSIBILITY?

1.1 Definition of urban accessibility

Accessibility is the ease with which people can reach places and opportunities – or, conversely, a characteristic of places and opportunities in terms of how easily they can be reached by the population (Geurs and Van Wee, 2004; Neutens et al., 2010).

Accessibility conditions are influenced both by the spatial co-distribution of the population, economic activities and public services, as well as by the configuration and performance of the transport network. In this sense, urban accessibility plays a fundamental role in shaping people's ability to move in order to access opportunities, such as jobs, schools etc.

Urban accessibility levels, therefore, are determined by three distinct components, as follows.

1) Infrastructure: how easy it is to access activities depends on existing infrastructure and transportation services. This includes, for example, the spatial coverage and connectivity of the public transport and street networks, the existence of rapid transit services such as trains and subways, etc. Here, both the efficiency and the spatial and temporal connectivity of the transport network are of utmost importance.
2) Land use: how easily activities can be accessed also depends on the spatial co-distribution of people and activities, such as schools, health services, leisure areas, etc. This component is related to the geographical proximity between people and opportunities: the further away an activity is, the more difficult it is to access it.
3) People: finally, it is important to note that the individuals' ability to access activities is also affected by their personal characteristics. Factors such as motor and cognitive difficulties, age, gender, race, and income, for example, can significantly influence people's ability to get around, use certain transport modes, and move around the city without fear of some kind of violence or discrimination.

This last component can be of critical importance for equity and social inclusion analyses. However, the influence of people's personal characteristics on accessibility conditions is usually better assessed through qualitative surveys: due to operational and computational challenges, this dimension of accessibility usually receives little attention from impact assessments of large-scale transportation projects. Chapter 2 discusses the operational, theoretical and communication advantages and disadvantages of different accessibility measures.

1.2 Difference between micro-accessibility and urban accessibility

In order to clarify the concepts we use throughout the book, it is important to distinguish between what we mean by urban accessibility and what is the colloquial use of the term accessibility.

The term *accessibility* is commonly used to refer to issues of universal design standards and regulations, as well as construction and planning practices aimed at the inclusion of people with different degrees of motor and cognitive challenges. This is usually understood as *microaccessibility*, because it covers issues of access to services and activities at the micro scale – i.e. how the planning of public and private spaces, and the design of vehicles and buildings, for example, affect the ability of individuals to access places, services, products etc.

Urban accessibility, on the other hand, can be understood as *macroaccessibility*, because it deals with a broader understanding of access. When we talk about urban accessibility, we focus on how structural issues of planning and urban development, such as the configuration of transport corridors and the spatial distribution of people and activities, affect people's ability to access opportunities. Urban accessibility addresses how the ability to access activities is influenced by people's ability to use transportation technologies, by the spatial co-distribution of people and activities, and by the spatial coverage and connectivity of transportation networks.

Microaccessibility and macroaccessibility are complementary elements of a broader notion of accessibility. Microaccessibility conditions, for example, directly affect the ability of people to board and use different modes of transport, to move safely on sidewalks, to cross streets etc. It is of little use for a person to live in a region served by various transport modes if, for example, she has limited mobility and the transport network and vehicles are not adapted to these challenges.

In this book, we will focus only on urban accessibility analyses and will often use the term accessibility as a synonym of macro-accessibility. It is important to recognize, however, that macroaccessibility alone provides only a limited account of one's accessibility conditions, and a more nuanced understanding of accessibility requires a closer inspection of microaccessibility conditions as well (Grisé et al., 2019; Buliung et al., 2021).

1.3 Why does urban accessibility matter?

The concept of accessibility is critical to transport and planning studies for different reasons. First, it explicitly articulates how the interaction between transport, urban development and land use policies impact people's ability to access opportunities dispersed in space. Moreover, access to opportunities and activities, such as jobs, education, and health services, plays a fundamental role satisfying individual and social needs and promoting social inclusion (Pereira and Karner, 2021; Luz and Portugal, 2022). Good accessibility is also a necessary condition, although not sufficient on its own, to expand people's freedom of choice (Church, Frost and Sullivan, 2000; Lucas, Van Wee and Maat, 2016; Van Wee, 2022). Therefore, the concept of accessibility helps us understand how transport and land use investments relate to elements that constitute the notions of social exclusion and wellbeing, such as freedom and one's satisfaction of basic needs.

Additionally, the idea of accessibility brings attention to the spatial dimension of inequality of opportunities, a central social justice problem. Urban accessibility helps to explicitly incorporate the notion of space into policy design to address inequalities (Farrington and Farrington, 2005; Pereira, Schwanen and Banister, 2017). Thus, accessibility is a fundamental concept when thinking about the equity implications of public policies and when evaluating which social groups and localities benefit from them.

As mentioned before, the accessibility levels in a city are a joint result of each person's ability to use transportation technologies, the spatial co-distribution of activities and population in the city, and the spatial and temporal connectivity of the transport network (Miller, 2018; Páez, Scott and Morency, 2012). As such, accessibility-oriented planning seeks to promote the integration between land use and the transport systems, getting people and activities closer together and reducing the dependence on motorized modes of transport (Banister, 2011). Planning cities and transport systems to improve accessibility conditions is therefore essential to promote more inclusive and sustainable cities.

1.4 Difference between accessibility and mobility

It is important to clarify the difference between accessibility and another concept widely used in our daily life: mobility. Unfortunately, the difference between these concepts is often ignored, even by researchers and planners who deal with these topics on a daily basis.

After all, there is a large intersection between accessibility and what is meant by "urban mobility" as a broad field of research and public policy: a field that deals with people's daily mobility patterns and which is related to the planning of public and individual transport systems, to the planning of cycling and pedestrian networks etc. In this context, it is not uncommon to hear, for example, that a given socioeconomic group has "less mobility" than another, when it's actually meant that this group has worse accessibility conditions. So what is the difference between accessibility and mobility?

In the urban and transport planning literature, the concept of mobility refers to people's daily travel behavior patterns – for example, how many trips are taken, which transport modes are used, the average trip distance and how much time people spend on commute.

Mobility data is commonly collected through household travel surveys. More recently, new technologies have been enabling the use of new data sources, such as mobile phones location services and smart cards, to examine daily mobility patterns (Anda, Erath and Fourie, 2017; Kandt and Batty, 2021). Mobility data and analyses provide information on how transport systems are used and on the travel behavior of people from different socioeconomic groups, which reflect important aspects of the economic and environmental performance of cities and of the well-being of the population.

Accessibility, however, refers to the *potential* ability to reach activities and opportunities. While a mobility analysis would focus, for example, on the time people spend commuting, an accessibility analysis would examine, for example, the quantity and variety of jobs one could potentially reach within a reasonable travel cost. Accessibility addresses how easy/feasible it is to reach a location, while mobility is concerned with the means of movement used to reach a location. Accessibility levels are, therefore, potential measures, while mobility data describe observed travel behavior.

Traditionally, urban and transport planning focuses on mobility (Banister, 2011; Vasconcellos, 2018; Levinson and King, 2020). Even today, the focus on mobility leads to the implementation of policies that prioritize private automobiles and that increase traffic flow and speed as a means to tackle congestion and reduce travel times (Levine, Grengs and Merlin, 2019). These policies tend to concentrate on the quantitative side of mobility, focusing on increasing the number of trips, increasing the average speed, decreasing congestion, etc.

From this perspective, mobility is understood as an end in itself, and the solutions to "improve" it would purely depend on technical solutions that "optimize" the quantitative aspects mentioned before. Mobility, however, cannot be seen as an end in itself. People seldom travel for the sake of moving around. On the contrary, people most oftenly travel as a means to access the activities or people they want to engage with at the trip destination.

In this sense, there is growing consensus among researchers and transportation agencies that the goal of a transport policy is to improve people's access to opportunities (Pereira, Schwanen and Banister, 2017; Martens, 2012; Bertolini, Clercq and Kapoen, 2005). If what people want is to access activities, we need to rethink how urban, land use and transport planning practices could be redesigned to improve accessibility without necessarily increasing traffic speeds or the dependence on motorized vehicles, which are known to cause negative economic, environmental and public health externalities.

There is a call for a paradigm shift in urban and transport planning in which the pursuit for more sustainable travel patterns requires changing the focus from mobility to accessibility (Banister, 2008; Cervero, 2005; Levine, Grengs and Merlin, 2019).

Policies that aim to increase traffic speed and road capacity, for example, could be replaced by policies that bring people and activities closer together and that encourage a more diverse land use mix, promoting the integration between transport and land use planning. Thus, the focus shift from mobility to accessibility opens up a wider range of possible public policy instruments and actions that aim to contribute to an urban development based on sustainability and social inclusion principles (Banister, 2011; Levine, Grengs and Merlin, 2019).

CHAPTER 2

2 ACCESSIBILITY MEASURES

Promoting a paradigm shift in urban and transport planning towards accessibility-oriented planning entails a few challenges. Among them, there is the need to develop and apply methods to measure the urban accessibility conditions in cities. The search for accessibility metrics that are easy to communicate, methodologically robust and computationally tractable lead researchers to develop a large number of different measures (Páez, Scott and Morency, 2012). These measures can be divided into two major groups: place-based measures and person-based measures (Dijst, Jong and Van Eck, 2002).

2.1 Place-based measures

Place-based metrics measure accessibility as a characteristic of a particular location. By simplification, these indicators assume that all people who are in the same place can equally access the activities distributed throughout the city. That is, if an accessibility analysis uses a place-based metric to calculate accessibility and divides the study area into a hexagonal grid, each cell of this grid (a hexagon) will have an accessibility value associated with it, which is equally assigned to all individuals residing within the cell. These measures are sensitive to land use and transport factors related to the spatial distribution of activities and to the configuration and performance of the transport network, but do not take into account people's individual characteristics.

These measures are the most widely used by transport agencies and researchers (Boisjoly and El-Geneidy, 2017; Papa et al., 2015). This is largely because they require less data and tend to be considerably easier to calculate and interpret than person-based measures. For this reason, the examples and case studies presented in this chapter and in the rest of the book focus only on place-based measures.

Place-based accessibility measures account for trip costs, usually expressed in terms of travel time (El-Geneidy et al., 2016; Venter, 2016) – i.e., if one location can be reached from another in half an hour, the cost to make this trip is 30 minutes. However, it is possible to consider other types of costs, such as the distance of the trip, its monetary cost and the passengers' perception of comfort (Arbex and Cunha, 2020; Herszenhut et al., 2022). We present below some of the place-based accessibility metrics most commonly used in the scientific literature and by transport agencies. Here, the term "cost" is used broadly, and can refer to any type of cost unit used to quantify the impedance of a trip, be it travel time, monetary cost or other alternatives.

2.1.1 Minimum travel cost

One of the simplest accessibility metrics, indicating the lowest cost required to reach the nearest opportunity from a given origin. It allows one to estimate, for example, the travel time from each block of the city to the closest health center. The indicator is calculated with the following formula:

$$A_i = min\left(c_{i1}, c_{i2}, \dots, c_{ij}, \dots, c_{i(n-1)}, c_{in}\right) \Leftrightarrow O_j \geq 1 \tag{1}$$

In which A_i is the accessibility at origin i, c_{ij} is the travel cost between origin i and destination j, n is the total number of destinations in the study area and O_j is the number of opportunities at destination j.

Advantages and disadvantages: the advantages of this measure are that it requires little data and it is easy to calculate and to communicate. Two disadvantages, however, are that it does not consider the amount of accessible opportunities at destinations and it does not take competition for opportunities into account. For example, even if a person lives very close to a hospital, this proximity does not necessarily guarantee good access to health services if that is the only hospital is subject to high demand peaks that overload the services beyond their capacities.

2.1.2 Cumulative opportunity measures

Computes the number of opportunities that can be reached within a given travel cost limit. For example, this indicator can be used to measure the number of jobs accessible by public transport in up to 60 minutes, or the number of schools accessible within 30 minutes of walking. It is calculated using the following formula:

$$A_i = \sum_{j=1}^{n} O_j \times f\left(c_{ij}\right) \tag{2}$$

$$f\left(c_{ij}\right) = \begin{cases} 1 & \text{if}\, c_{ij} \leq C \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

In which A_i is accessibility at origin i, O_j is the number of opportunities at destination j, n is the total number of destinations in the study area, $f\left(c_{ij}\right)$ is a binary function that assumes the values 0 or 1, depending on the travel cost c_{ij} between origin i and destination j and C is the travel cost threshold.

Advantages and disadvantages: the cumulative opportunities measure also requires little data and is easy to calculate and communicate. This helps explain why this is one of the indicators most commonly used by transport and funding agencies in accessibility analyses (Papa et al., 2015; Boisjoly and El-Geneidy, 2017).

Among its disadvantages are the fact that this indicator does not consider the competition for opportunities and that it requires the choice of a single cut-off point as a travel cost limit. Moreover, this measure assumes that all opportunities that can be reached within the travel cost limit are equally desirable and accessible. For example, if we consider a 60-minute travel time limit, an opportunity that is 40 minutes away from an origin is considered as accessible as another one that is just 10 minutes away.

2.1.3 Gravity measures

More than a specific type of accessibility metric, we can understand gravity-based accessibility as a family of measures. As in the case of the cumulative opportunities measure, gravity-based metrics consider the sum of opportunities that can be reached from a given location. However, the number of opportunities in each destination is gradually discounted as travel costs become higher. In other words, opportunities that are easier to access are considered to be more valuable, and the weight of each opportunity decreases as it gets more difficult to reach it from the trip origin.

The rate at which this weight decreases is determined by a *decay function*. For example, the *linear* decay function considers that the weight of each opportunity decreases linearly up to a certain cost limit, after which the weight becomes zero. The *negative exponential* function, on the other hand, considers that the weight of each opportunity is divided by a factor that grows exponentially, causing the weight to decrease rapidly at low travel costs and to approach 0 at high costs. The equations below present the generic formulation of a gravitational measure, as well as the linear and negative exponential decay functions mentioned above.

$$A_i = \sum_{j=1}^{n} O_j \times f(c_{ij}) \tag{4}$$

$$f_{lin}(c_{ij}) = \begin{cases} 1 - c_{ij}/C & \text{if } c_{ij} \leq C \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$f_{exp}(c_{ij}) = e^{-\beta c_{ij}} \tag{6}$$

In which A_i is the accessibility at origin i, O_j is the number of opportunities at destination j, n is the total number of destinations in the study area, $f(c_{ij})$ is a decay function whose result varies with the travel cost c_{ij} between origin i and destination j, $f_{lin}(c_{ij})$ is the linear decay function, C is travel cost limit, $f_{exp}(c_{ij})$ is the negative exponential decay function and β is a parameter that dictates the decay speed.

There are numerous types of decay functions that can be used when calculating gravity-based accessibility measures. The cumulative opportunities measure, for example, can be understood as a special case of a gravity-based measure in which the weight of each opportunity is set by a binary function, rather than a function that decays gradually. Levinson and King (2020, p. 49) present a list of decay functions often used by transport agencies and researchers in analyses involving gravity measures.

Advantages and disadvantages: the main advantage of gravity-based accessibility measures is that, by discounting the weight of opportunities by travel cost, these measures reflect to some extent how people perceive access to opportunities: services and activities that are closer to them tend to be perceived as more valuable, all else equal. This indicator, however, has at least two disadvantages. The first is that the estimated accessibility levels are difficult to interpret because of the way in which the number of opportunities is discounted by travel costs. Additionally, the decay rate of the impedance function (the β parameter of the negative exponential function, for example) needs to be calibrated if one wants the accessibility estimates to be representative of people's travel behavior. Therefore, gravity-based metrics require additional travel behavior data to be used in the calibration process, coming, for example, from household travel surveys or mobile phone services.

2.1.4 Accessibility measures with competition: floating catchment area

In many cases, access to opportunities is affected not only by geographical proximity and transportation costs, but also by the competition of many people trying to access the same opportunity. This is very common, for example, in the cases of access to health services, schools and jobs. A job opening can only be occupied by one person at a time, and the same goes for an Intensive Care Unit (ICU) bed or a school seat.

There are various measures that seek to account for competition effects in accessibility estimates. Some of the most widely used are those in the floating catchment area (FCA) family of indicators. For example, these indicators try to take into account how the same person can potentially access multiple ICU beds and, simultaneously, how each ICU bed can potentially be accessed by multiple people. Thus, a person's access to ICU beds is influenced both by transportation costs and by the availability of beds, given the potential competing demand for them.

Within the FCA measures' family, the most commonly used is the 2-step floating catchment area (2SFCA), originally proposed by Luo and Wang (2003). One limitation of 2SFCA is that it considers that the same person can demand multiple services/opportunities at the same time and that the same service can be used by multiple people at the same time. These issues are known as the demand and supply inflation problems, respectively, and can generate biased or inaccurate accessibility estimates (Páez, Higgins and Vivona, 2019). To deal with these problems, Páez, Higgins and Vivona (2019) proposed the balanced floating catchment area (BFCA), one of the most recent measures of the FCA family.

Advantages and disadvantages: different FCA measures have different advantages and disadvantages, to a greater or lesser extent. However, in general, the main advantage of measures from this family is their ability to incorporate aspects of competition into accessibility estimates. The main disadvantage, on the other hand, is the difficulty to interpret and communicate their results.

2.2 Person-based measures

Person-based accessibility measures are sensitive not only to the spatial distribution of activities and to the configuration and performance of transportation networks. Indicators in this group also take into account how the individual characteristics of each person (such as gender, age, physical disability etc.), and even the participation in certain activities and personal commitments, can affect people's ability to access opportunities. This category includes, for example, activity-based indicators (Dong et al., 2006) and space-time measures (Kim and Kwan, 2003; Neutens et al., 2012).

Advantages and disadvantages: although person-based accessibility measures are more sophisticated, they often require large amounts of data, such as travel diary records, household travel surveys, etc. Therefore, the calculation of these measures is computationally more intensive, which makes them less frequently used than place-based measures (Neutens et al., 2010; Miller, 2018). In contrast to place-based measures, which yield a single accessibility estimate for all individuals in the same place, person-based measures results associate one accessibility estimate to each person in the study area. While this allows for more nuanced accessibility analyses, as the resultant accessibility estimates take the particularities of each individual into account, this also makes the communication and interpretation of results more complex.

SECTION 2

HOW TO MEASURE URBAN ACCESSIBILITY

The purpose of this section is to show how to calculate urban accessibility estimates in R using the `{r5r}` and `{accessibility}` packages.

Calculating accessibility levels in a study area involves two major steps: first, we need to calculate a travel cost matrix between the origins and destinations of this area; next, we calculate the accessibility from each origin, taking into consideration the transport costs between origin-destination pairs and the number of opportunities in each destination. In this section, we will learn how to execute both of these steps using the R programming language. We will also learn about the data required in each step and the pros and cons of the different methods that can be used to generate accessibility estimates.

CHAPTER 3

3 CALCULATING ACCESSIBILITY ESTIMATES IN R

3.1 Calculating a travel time matrix

The first step to estimate the accessibility levels in a study area is to calculate the travel cost matrix between the various origins and destinations that make up this area. As previously mentioned, in the scientific literature and in transport planning practice this cost is generally represented by the travel time between two points (El-Geneidy et al., 2016; Venter, 2016), although recent studies have also considered other factors, such as trip monetary costs and the comfort of a trip between origins and destinations (Arbex and Cunha, 2020; Herszenhut et al., 2022). In this book, we will focus on travel time matrices as they are the most widely used in the literature and in practice, and we will cover other types of costs in a future book on advanced accessibility analysis in R.

Currently, the easiest and fastest way to generate a travel time matrix in R is using the `{r5r}` package (Pereira et al., 2021), developed by the AOP team. The package provides a simple and friendly interface to the R5 multimodal transport routing engine, developed by Conveyal.[5]

3.1.1 Installing `{r5r}`

Installing `{r5r}` works the same as installing any other R package (all code snippets from this point onward must be run in an R session).

```r
install.packages("r5r")
```

In addition to R, `{r5r}` also requires installing Java 11.[6] Use the command below to check the version of Java installed in your computer.

```r
cat(processx::run("java", args = "-version")$stderr)
```

```
openjdk version "11.0.19" 2023-04-18
OpenJDK Runtime Environment (build 11.0.19+7-post-Ubuntu-
0ubuntu120.04.1)
OpenJDK 64-Bit Server VM (build 11.0.19+7-post-Ubuntu-
0ubuntu120.04.1, mixed mode, sharing)
```

[5] Available at: https://github.com/conveyal/r5.

[6] Java 11 is available at: https://www.oracle.com/java/technologies/downloads/#java11; or at https://jdk.java.net/java-se-ri/11.

As we can see, the version installed in the book is compatible with `{r5r}`. If the version installed in your machine is not compatible (i.e. if running the code above results in an output mentioning version 12 or 1.8.0, for example), please install Java 11.

3.1.2 Required data

Using `{r5r}` requires different types of data. The list below describes the required and optional data sets and indicates where you could obtain these data.

1) Street network (required): a file in `.pbf` format containing the street network and pedestrian infrastructure as described by OpenStreetMap (OSM). It can be downloaded using: `{osmextract}` (R package); Geofabrik; HOT Export Tool; or BBBike Extract Service.
2) Public transport network (optional): one or more GTFS files describing the public transport network of the study area. If absent, public transport trips are not considered in the travel time matrix. This type of data can be downloaded using: `{tidytransit}` (R package); or Transitland. In chapter 4 (table 9) we also show where to download the GTFS files of Brazilian cities that share their data publicly.
3) Topography (optional): a raster file containing the digital elevation model data of the study area in `.tif`/`.tiff` format. This data set is required if you wish to consider the effects of topography on walking and cycling travel times. It can be downloaded using: `{elevatr}` (R package); or National Aeronautics and Space Administration (Nasa) SRTMGL1.

BOX 1
OSM data quality

OSM is a geographic database that provides information about street networks, buildings, parks etc. OSM is maintained by a community of volunteers, so the coverage and quality of its data can widely vary between regions (Barrington-Leigh and Millard-Ball, 2017). OSM data tend to have better coverage and quality in more developed regions and in urban areas with large populations (Barrington-Leigh and Millard-Ball, 2017; Camboim, Bravo and Sluter, 2015).

Authors' elaboration.

These data sets should be saved in a single directory, which should preferably not contain any other files. As we will see below, `{r5r}` combines all the data saved in this directory to create a multimodal transport network that is used for routing trips between origin-destination pairs and, consequently, for calculating travel time matrices. Please note that you can have more than one GTFS file in the same directory, in which case `{r5r}` considers the public transport networks of all feeds together. The street network and the topography of the study area, however, must be each one described by a single file. Assuming that R scripts are saved in a directory called `R`, a possible file structure is shown below:

```
/tmp/RtmpaPPJmV/accessibility_project
├── R
│   ├── script1.R
│   └── script2.R
└── r5
    ├── public_transport_network.zip
    ├── street_network.osm.pbf
    └── topography.tif
```

To illustrate the features of `{r5r}`, we will use a small data sample from the city of Porto Alegre (Brazil). This data is available within `{r5r}` itself, and its path can be accessed with the following command:

```
data_path <- system.file("extdata/poa", package = "r5r")
data_path
```

```
[1] "/home/runner/work/intro_access_book/renv/library/R-4.3/
x86_64-pc-linux-gnu/r5r/extdata/poa"
```

```
fs::dir_tree(data_path)
```

```
/home/runner/work/intro_access_book/renv/library/R-4.3/x86_64-
pc-linux-gnu/r5r/extdata/poa

├── poa_elevation.tif
├── poa_eptc.zip
├── poa_hexgrid.csv
├── poa_osm.pbf
├── poa_points_of_interest.csv
└── poa_trensurb.zip
```

This directory contains five files that we will use throughout this chapter and are listed below.

1) An OSM street network: `poa_osm.pbf`.
2) A GTFS feed describing some of the city's bus routes: `poa_eptc.zip`.
3) A GTFS feed describing some of the city's train routes: `poa_trensurb.zip`.
4) The digital elevation model of the city: `poa_elevation.tif`.
5) The `poa_hexgrid.csv` file, containing the geographic coordinates of the centroids of a regular hexagonal grid covering the entire study area

and information about the number of residents, jobs, hospitals and schools in each grid cell. These points will be used as the origins and destinations when calculating the travel time matrix.

3.1.3 Computing the travel time matrix

Before calculating the travel time matrix, we need to increase the memory available to run Java processes, used by the underlying R5 routing engine. This is necessary because R allocates only 512 MB of memory to Java processes by default, which is often not enough to calculate large matrices with {r5r}. To increase the available memory to 2 GB, for example, we use the following command at the beginning of the script, even before loading the R packages that will be used in our analysis:

```r
options(java.parameters = "-Xmx2G")
```

We can then proceed with the travel time matrix calculation, which we carry out in two steps. First, we need to generate a multimodal transport network that will be used to route trips between origin-destination pairs. To do this, we load {r5r} and use the setup_r5() function, which downloads the R5 routing engine and uses it to build the network. This function receives the path to the directory where the input data is saved and saves some files describing the routing network to this directory. It also outputs a connection to R5, which we named r5r_core in this example. This connection is responsible for linking the function calls with the transport network and is used to calculate the travel time matrix.

```r
library(r5r)

r5r_core <- setup_r5(data_path, verbose = FALSE)

fs::dir_tree(data_path)
```

```
/home/runner/work/intro_access_book/renv/library/R-4.3/x86_64-
pc-linux-gnu/r5r/extdata/poa

├── fares
│   └── fares_poa.zip
├── network.dat
├── network_settings.json
├── poa_elevation.tif
├── poa_eptc.zip
├── poa_hexgrid.csv
├── poa_osm.pbf
├── poa_osm.pbf.mapdb
├── poa_osm.pbf.mapdb.p
├── poa_points_of_interest.csv
└── poa_trensurb.zip
```

The second and final step is to actually calculate the travel time matrix with the `travel_time_matrix()` function. As basic inputs, the function receives the connection with R5 created above, origin and destination points as `data.frames` with columns `id`, `lon` and `lat`, the mode of transport considered, the departure time, the maximum walking time allowed when accessing public transport stations from the origin and when egressing from the last stop to the destination, and the maximum travel time allowed for trips. The function also accepts several other inputs, such as the walking speed and the maximum number of public transport legs allowed, among others.[7]

```
# read data.frame with grid centroids
points <- data.table::fread(file.path(data_path,
"poa_hexgrid.csv"))

ttm <- travel_time_matrix(
  r5r_core,
  origins = points,
  destinations = points,
  mode = c("WALK", "TRANSIT"),
  departure_datetime = as.POSIXct(
    "13-05-2019 14:00:00",
    format = "%d-%m-%Y %H:%M:%S"
  ),
  max_walk_time = 30,
  max_trip_duration = 120,
  verbose = FALSE,
  progress = FALSE
)

head(ttm)
```

```
            from_id            to_id travel_time_p50
1: 89a901291abffff 89a901291abffff               1
2: 89a901291abffff 89a9012a3cfffff              71
3: 89a901291abffff 89a901295b7ffff              41
4: 89a901291abffff 89a901284a3ffff              57
5: 89a901291abffff 89a9012809bffff              43
6: 89a901291abffff 89a901285cfffff              35
```

[7] For more information on each parameter, please refer to the function documentation in an R session (with the commands `?travel_time_matrix()` or `help("travel_time_matrix")`) or on `{r5r}` website, available at: https://ipeagit.github.io/r5r/reference/travel_time_matrix.html.

In practice, `travel_time_matrix()` finds the fastest route from each origin to all possible destinations taking into account the transport mode, the departure time and the other inputs set by the user. For this, `{r5r}` considers door-to-door travel times: in the case of a public transport trip, for example, the total travel time includes: i) the walking time from the origin to the public transport stop; ii) the waiting time at the stop; iii) the in-vehicle time; and iv) the walking time from the last public transport stop to the destination. When more than one public transport route is used, `{r5r}` also considers the time spent on transfers, which consists of walking between stops and waiting for the next vehicle.

BOX 2

Routing speed with `travel_time_matrix()`

The `travel_time_matrix()` function uses an extension of the RAPTOR routing algorithm (Conway, Byrd and Van Der Linden, 2017), making R5 extremely fast. Depending on the number of origin-destination pairs, `{r5r}` can calculate travel time matrices between 6 and 200 times faster than other multimodal routing softwares (Higgins et al., 2022).

Authors' elaboration.

3.2 Calculating accessibility

Having calculated the travel time matrix between the origins and destinations, we need to use it to calculate accessibility levels in the study area. For this, we will use the `{accessibility}`[8] package, also developed by the AOP/Ipea team, which provides several functions to calculate many accessibility measures. As basic inputs, all functions require a pre-calculated cost matrix (in our case, the travel time matrix calculated in the previous section) and some land use data, such as the number of opportunities in each cell that covers the study area.

3.2.1 Cumulative opportunities measure

The `cumulative_cutoff()` function is used to calculate the number of opportunities that can be reached within a given travel cost limit. In the example below, we first load the `{accessibility}` package and then calculate the number of schools that can be reached in 30 minutes from each origin.

```r
library(accessibility)

# rename column to use {accessibility} package
data.table::setnames(ttm, "travel_time_p50", "travel_time")

cum_opportunities <- cumulative_cutoff(
```

[8] Available at: https://github.com/ipeaGIT/accessibility.

```
  ttm,
  land_use_data = points,
  opportunity = "schools",
  travel_cost = "travel_time",
  cutoff = 30
)

head(cum_opportunities)
```

```
                id schools
1: 89a901291abffff      23
2: 89a9012a3cfffff       0
3: 89a901295b7ffff      18
4: 89a901284a3ffff       4
5: 89a9012809bffff      20
6: 89a901285cfffff      84
```

3.2.2 Minimum travel cost

The `cost_to_closest()` function, on the other hand, calculates the minimum travel cost required to reach a certain number of opportunities. With the code below, for example, we calculate the minimum travel time to reach the nearest hospital from each origin.

```
min_time <- cost_to_closest(
  ttm,
  land_use_data = points,
  opportunity = "healthcare",
  travel_cost = "travel_time"
)

head(min_time)
```

```
                id travel_time
1: 89a9012124fffff         Inf
2: 89a9012126bffff          19
3: 89a9012127bffff          16
4: 89a90128003ffff          14
5: 89a90128007ffff          11
6: 89a9012800bffff          13
```

3.2.3 Gravity measures

The `gravity()` function calculates gravity-based accessibility measures, those in which the weight of each opportunity gradually decreases as the travel cost increases. Since many different decay functions can be used to calculate gravity measures, such as the negative exponential, inverse power etc. This function receives an additional input: the decay function that should be used to calculate the opportunity weights. The example below calculates accessibility to schools using a negative exponential gravity measure with a decay parameter of 0.2.

```r
negative_exp_grav <- gravity(
  ttm,
  land_use_data = points,
  opportunity = "schools",
  travel_cost = "travel_time",
  decay_function = decay_exponential(0.2)
)

head(negative_exp_grav)
```

```
                id     schools
1: 89a901291abffff 0.428108826
2: 89a9012a3cfffff 0.003987477
3: 89a901295b7ffff 0.606786304
4: 89a901284a3ffff 0.079661746
5: 89a9012809bffff 0.494632773
6: 89a901285cfffff 1.987657134
```

3.2.4 Competitive measures

Finally, the `floating_catchment_area()` function calculates accessibility levels considering the competition for opportunities using different FCA methods. Because several FCA methods can be used, the function requires users to indicate which method to use. In addition, just like the `gravity()` function, the decay function must also be defined by the user. The following code shows an example of how to calculate accessibility to schools using the BFCA method (Páez, Higgins and Vivona, 2019) and an exponential decay function with a decay parameter of 0.05.

```r
bfca_competition <- floating_catchment_area(
  ttm,
  land_use_data = points,
  opportunity = "schools",
  travel_cost = "travel_time",
  demand = "population",
  method = "bfca",
  decay_function = decay_exponential(0.05)
)

head(bfca_competition)
```

```
                 id      schools
1: 89a901291abffff 2.628973e-04
2: 89a9012a3cfffff 5.875302e-05
3: 89a901295b7ffff 2.123543e-04
4: 89a901284a3ffff 1.414356e-04
5: 89a9012809bffff 2.254543e-04
6: 89a901285cfffff 3.901031e-04
```

The functions presented in this section can also receive other inputs not explicitly mentioned here. For more information on each parameter, please refer to the documentation of the `{accessibility}` package on its website.

3.2.5 Calculating accessibility with `{r5r}`

In the previous two sections, we learned how to calculate accessibility levels step-by-step. For didactic purposes, it is important to understand that calculating accessibility first requires calculating a travel cost matrix which is then used to estimate accessibility levels. However, `{r5r}` also includes an `accessibility()` function that calculates accessibility levels in a single call, which is much faster and does not require intermediate steps.

Similar to the travel time matrix function, `accessibility()` receives as inputs a connection to R5, origins, destinations, transport mode, departure time, among other arguments. Additionally, it also requires users to list which opportunities and which decay function should be considered, as well as the value of the cost threshold/decay parameter, depending on the decay function. The example below shows how to use this function to calculate a cumulative opportunity metric (`decay_function = "step"`). In this example, we calculate the number of schools accessible by walk and public transport in 30 minutes (`cutoffs = 30`).

```r
r5r_access <- accessibility(
  r5r_core,
  origins = points,
  destinations = points,
  opportunities_colname = "schools",
  decay_function = "step",
  cutoffs = 30,
  mode = c("WALK", "TRANSIT"),
  departure_datetime = as.POSIXct(
    "13-05-2019 14:00:00",
    format = "%d-%m-%Y %H:%M:%S"
  ),
  max_walk_time = 30,
  max_trip_duration = 120,
  verbose = FALSE,
  progress = FALSE
)

head(r5r_access)
```

```
                id opportunity percentile cutoff accessibility
1: 89a901291abffff     schools         50     30            21
2: 89a9012a3cfffff     schools         50     30             0
3: 89a901295b7ffff     schools         50     30            16
4: 89a901284a3ffff     schools         50     30             4
5: 89a9012809bffff     schools         50     30            17
6: 89a901285cfffff     schools         50     30            78
```

There is one small difference between r5r::accessibility() and accessibility::cumulative_cutoff(). In r5r::accessibility(), the function only considers trips below the travel time threshold, while accessibility::cumulative_cutoff() considers trips with costs equal or below the maximum threshold. That is, to perform the same operation above but with cumulative_cutoff() we need to set a cutoff of 29 minutes, not 30. We compare the results of the two functions with the code below.

```
cum_cutoff_29 <- cumulative_cutoff(
  ttm,
  land_use_data = points,
  opportunity = "schools",
  travel_cost = "travel_time",
  cutoff = 29
)

access_comparison <- merge(
  r5r_access,
  cum_cutoff_29,
  by = "id"
)

data.table::setnames(
  access_comparison,
  old = c("accessibility", "schools"),
  new = c("r5r_access", "accessibility_access")
)

head(access_comparison[, .(id, r5r_access, accessibility_access)])
```

```
                id r5r_access accessibility_access
1: 89a9012124fffff          1                    1
2: 89a9012126bffff         12                   12
3: 89a9012127bffff         14                   14
4: 89a90128003ffff         30                   30
5: 89a90128007ffff         21                   21
6: 89a9012800bffff         29                   29
```

As we can see, the results of the two functions are identical after this small adjustment. The main difference between the two methods, however, is that the “intermediate” information of travel time between origins and destinations is not available when using `r5r::accessibility()`. Still, this workflow can be a good alternative for people who are solely interested in the accessibility levels and do not require the travel time between points in their analyses. Also, note that the `{accessibility}` package has a wider range of accessibility indicators and provides more flexibility for users to define custom decay functions.

BOX 3

Considering other types of travel costs when calculating accessibility

Another difference between `{r5r}` accessibility function and `{accessibility}` functions is the fact that the latter can work with various types of travel costs, such as time, monetary cost, comfort etc. `{r5r}` function, however, is less flexible, and only considers travel time constraints.

Authors’ elaboration.

3.3 Accessibility analyses

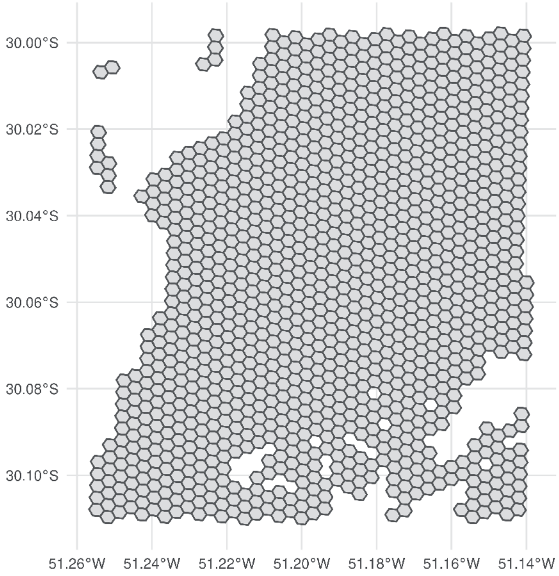
Having calculated accessibility levels, we can now analyze them. There is a wide variety of analyses that can be performed using this data: diagnosis of urban accessibility conditions in different neighborhoods, analyses of inequalities in access to opportunities between different social groups, analyses of social exclusion and accessibility poverty etc. In this section, however, we will present only two relatively simple and easy-to-communicate analyses: the spatial distribution of accessibility and its distribution among different income groups.

3.3.1 The spatial distribution of urban accessibility

To understand how urban accessibility is spatially distributed in a given city or region, we first need to obtain the spatial information of the points we have used as origins and destinations in our travel cost matrix. The points we used in the previous examples correspond to the centroids of a hexagonal grid based on the H3 index, developed by Uber (Brodsky, 2018). The grid of Porto Alegre, as well as some sociodemographic and land use data aggregated to it, is made available by the AOP team through the `{aopdata}` R package (Pereira et al., 2022). The package and its functions are presented in detail in section 5. With the code below, we load the data visualization package, download the grid and present it in a map.

```
library(ggplot2)

# download spatial grid
poa_grid <- aopdata::read_grid("Porto Alegre")

# keeps only the cells used in the travel time matrix
poa_grid <- subset(poa_grid, id_hex %in% points$id)

# plot map
ggplot(poa_grid) + geom_sf() + theme_minimal()
```

FIGURE 1
Hexagonal grid covering central Porto Alegre

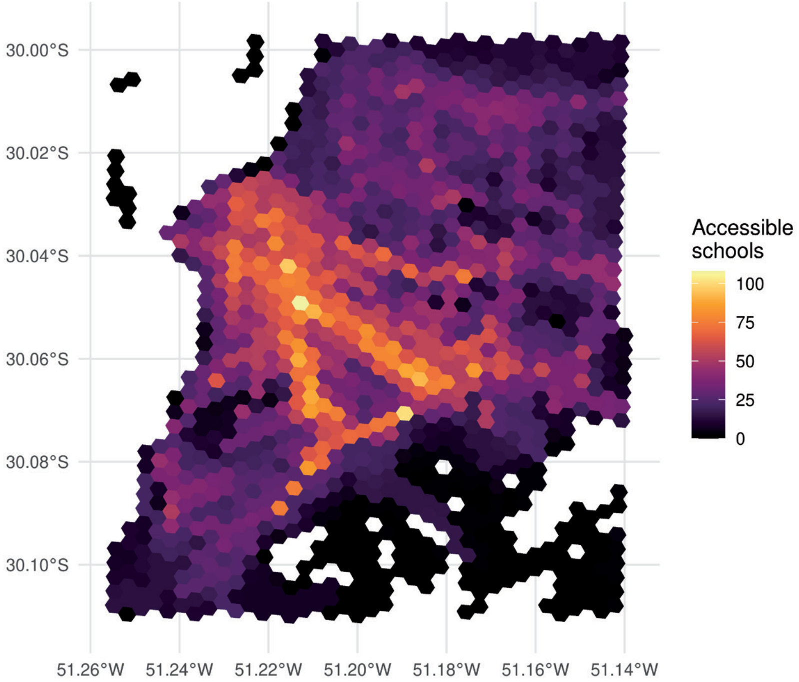


Source: Figure generated by the code snippet above.

To spatially visualize accessibility data, we need to combine the table of accessibility estimates with the table that contains the spatial grid, using the hexagon identification columns as key columns. The code below shows how to merge the two tables and plot the map. In this example, we are going to use the cumulative access to schools in 30 minutes that we have calculated previously.

```
spatial_access <- merge(
  poa_grid,
  cum_opportunities,
  by.x = "id_hex",
  by.y = "id"
)

ggplot(spatial_access) +
  geom_sf(aes(fill = schools), color = NA) +
  scale_fill_viridis_c(option = "inferno") +
  labs(fill = "Accessible\nschools") +
  theme_minimal()
```

FIGURE 2
Spatial distribution of accessibility to schools in central Porto Alegre



Source: Figure generated by the code snippet above.

As the map above shows, accessibility levels tend to be higher in the city center, where there is a greater concentration of schools, and close to the major transport corridors, as the people who live closer to these high-capacity and speed corridors tend to access distant locations relatively fast. In contrast, people who live farther away from these corridors depend on lower-frequency and speed modes (such as municipal buses, for example) and need to spend more time to reach the medium/high-capacity corridors. As a result, the accessibility levels of those living far from the city center and from high-capacity corridors tend to be relatively lower.

3.3.2 The distribution of urban accessibility across socioeconomic groups

Figure 2, although useful to reveal the places with the highest and lowest levels of accessibility, says nothing about the socioeconomic conditions of the people who have better or worse accessibility conditions in the region. To understand this, we need to join the previously calculated accessibility data with the demographic and economic information of the people living in each grid cell.

In the example below, we merge the accessibility estimates with the information of average income decile of the population in each cell (data also made available through the `{aopdata}` package).

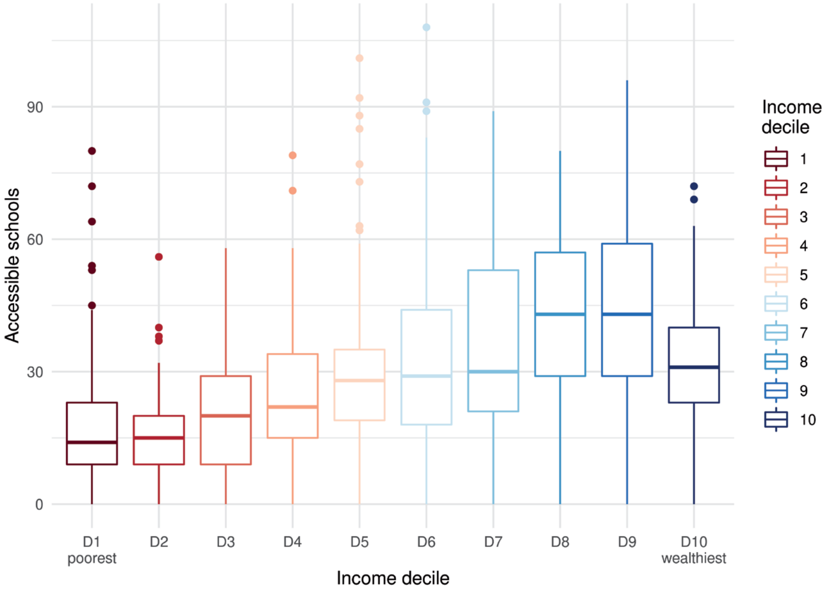
```
# download population and socioeconomic data
poa_population <- aopdata::read_population("Porto Alegre",
showProgress = FALSE)

# renames the columns with population count and income decile
data
data.table::setnames(
  poa_population,
  old = c("P001", "R003"),
  new = c("pop_count", "decile")
)

# merge accessibility and population tables
sociodemographic_access <- merge(
  spatial_access,
  poa_population,
  by = "id_hex"
)

head(sociodemographic_access[, c("id_hex", "schools",
"pop_count", "decile")])
```

```
Simple feature collection with 6 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -51.25678 ymin: -30.1111 xmax: -51.19031 ymax: -30.06699
Geodetic CRS:  WGS 84

           id_hex schools pop_count decile                       geometry
1 89a9012124fffff       1       733      9 POLYGON ((-51.25083 -30.111...
2 89a9012126bffff      13       355      9 POLYGON ((-51.25369 -30.106...
3 89a9012127bffff      14       996     10 POLYGON ((-51.2538 -30.1094...
4 89a90128003ffff      34      1742      4 POLYGON ((-51.19446 -30.071...
5 89a90128007ffff      23       477      5 POLYGON ((-51.19744 -30.069...
6 89a9012800bffff      34       501      4 POLYGON ((-51.19137 -30.070...
```

With the information on the income decile of each hexagon, we can analyze the distribution of accessibility levels by income groups. For this, we need to weigh the accessibility level of each origin by the number of people who reside there. This will tell us the accessibility distribution of the people located in cells with a given income decile. If we do not weigh the accessibility estimate by the population, we will have the accessibility distribution of the cells per se, and not of the people located in each cell. Since our analysis focuses on people, and not on the spatial units in which they are aggregated, weighting the accessibility levels by the population count is an essential part of it. The accessibility distribution of each decile is shown below.

```r
ggplot(subset(sociodemographic_access, !is.na(decile))) +
  geom_boxplot(
    aes(
      x = as.factor(decile),
      y = schools,
      color = as.factor(decile),
      weight = pop_count
    )
  ) +
  labs(
    color = "Income\ndecile",
    x = "Income decile",
    y = "Accessible schools"
  ) +
  scale_color_brewer(palette = "RdBu") +
  scale_x_discrete(
    labels = c("D1\npoorest", paste0("D", 2:9), "D10\nwealthiest")
  ) +
  theme_minimal()
```

FIGURE 3
Distribution of accessibility to schools in central Porto Alegre by income decile



Source: Figure generated by the code snippet above.

Figure 3 is very clear: lower-income residents tend to have considerably lower accessibility levels than high-income residents. This is a common pattern in virtually all Brazilian cities (Pereira et al., 2020) and which results, to a large extent, from the spatial distribution patterns of low- and high-income communities: the wealthiest usually live in high-valued areas, close to large employment hubs (and opportunities for education, health, leisure etc.) and with relatively better public transport services. The poorest, on the other hand, tend to live in the cities' outskirts, where the land value is lower and the distances from the large concentrations of opportunities are larger. Additionally, in most cases the provision of mid- and high-capacity public transport services is worse in regions with high concentrations of low-income residents. As a result, low-income communities have, on average, much lower accessibility levels than wealthier communities, as the chart illustrates.

SECTION 3

PUBLIC TRANSPORT DATA

The purpose of this section is: i) to introduce the GTFS public transport data specification; ii) to show where to download GTFS data for Brazilian cities and for other cities around the globe; and iii) to show how to manipulate and analyze GTFS data using R.

Public transport data is a key element of transport planning in general, and of accessibility analyses in particular. To be used with confidence, this data needs to be reliable and of simple inspection and interpretation.

To meet these criteria, transport agencies, decision makers and researchers have been trying to use data structured according to open and collaborative specifications – that is, whose format is decided upon by a community of different actors, including data producers (e.g. public transport agencies) and consumers (e.g. researchers and software developers). Although an open specification does not necessarily improve the quality and reliability of the data it describes, it brings many advantages that promote knowledge-sharing and transparency of analyses and applications that depend on it – factors that can substantially improve data quality and reliability.

The widespread usage of a standard data format to represent public transport systems promotes the development of computational tools and softwares that analyze and make use of this data, which helps creating a space in which actors from different cities and countries can learn from and support each other. Thus, an application developed by a Brazilian transport agency can easily be used by a researcher in the United States, a Japanese developer or another transport agency in South Africa – as long as, of course, they organize their data in the same format. Moreover, the more widely this format is used, the greater the reliability of the specification itself, as multiple actors tend to expand their ability to use, interpret and inspect this data.

The open and collaborative data specification most widely used in public transport planning and operation is the GTFS format, short for General Transit Feed Specification. As shown in chapter 3, GTFS feeds are also important pieces of data when estimating urban accessibility levels by public transport. In this section, we will learn more about GTFS data, how it is structured and how to work with it in R.

CHAPTER 4

4 GTFS DATA

The GTFS format is an open and collaborative specification that aims to describe the main components of a public transport network. Originally created in the mid-2000s by a partnership between Google and TriMet, the transport agency of Portland, Oregon, in the United States, the GTFS specification is now used by transport agencies in thousands of cities, spread across all continents of the globe (McHugh, 2013). Currently, the specification is divided in two distinct components:

- the GTFS Schedule, or GTFS Static, which contains the planned schedule of public transport trips, information about their fares and spatial information about their itineraries; and
- the GTFS Realtime, which is used to inform, in real-time, vehicle location information, alerts for possible delays, itinerary changes and events that may interfere with the planned schedule.

Throughout this section, we will focus on *GTFS Schedule*, the most widely used GTFS format in accessibility analyses and by transport agencies.[9]

Being an open and collaborative specification, the GTFS format attempts to enable several distinct uses that transport agencies and tool developers might find for it. However, agencies and applications may still depend on information that is not included in the official specification. As a result, different specification extensions have been created, and some of them may eventually be incorporated into the official specification if this is agreed upon by the GTFS community. In this section, we will focus on a subset of information available in the basic GTFS Schedule format, thus not covering its extensions.

4.1 GTFS structure

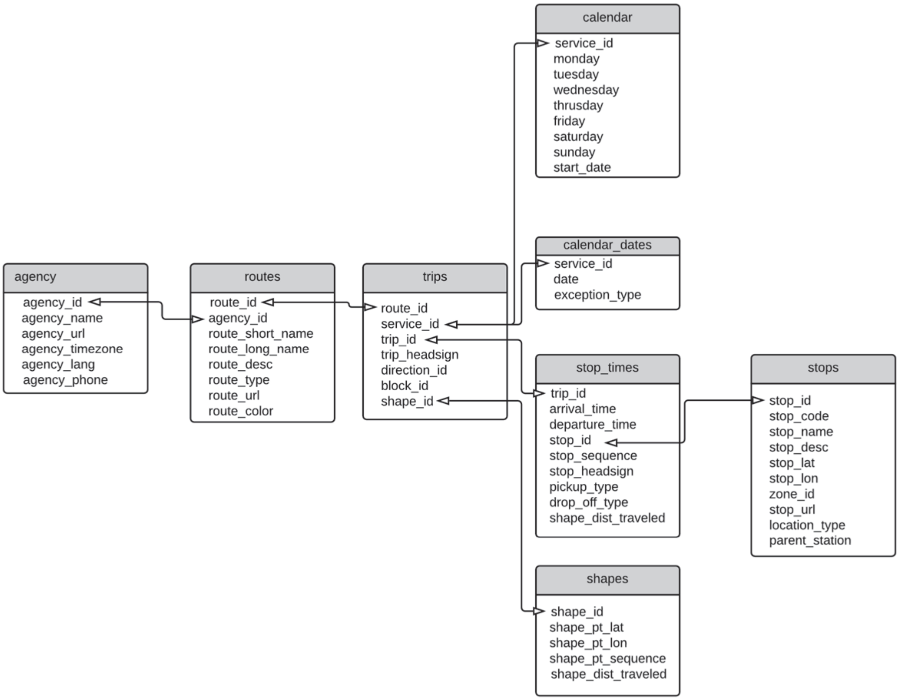
Files in the GTFS Schedule format (from this point onwards referred to as GTFS) are also known as feeds.[10] A feed is nothing more than a compressed `.zip` file that contains a set of tables, saved in separate `.txt` files, describing some aspects of the public transport network (stops/stations location, trip frequency, itineraries paths etc.). Just like in a relational database, tables in a feed have key columns that allow one to link information described in one table to the data described in another one. An example of the GTFS scheme is presented in figure 4, which

[9] More information on GFTS realtime is available at: https://gtfs.org/realtime/.

[10] In this book, we will use the terms feed, GTFS file and GTFS data as synonyms.

shows some of the the most important tables that make up the specification and highlights the key columns that link the tables together.

FIGURE 4
GTFS format scheme



Source: Pereira, Andrade and Vieira (2022).

In total, the GTFS format can be made of up to 22 tables.[11] Some of them, however, are optional, meaning that they don't need to be present for the feed to be considered valid. The specification classifies the presence of a table into the following categories: required, optional and conditionally required (when the requirement of the table depends on the existence of another particular table, column or value). For simplicity, we will consider only the first two categories in this book and will indicate whether a table is required whenever appropriate. Using our simplified convention, tables are classified as follows.

[11] According to the official specification as of May 9th 2022.

1) Required: `agency.txt`; `stops.txt`; `routes.txt`; `trips.txt`; `stop_times.txt`; `calendar.txt`.
2) Optional: `calendar_dates.txt`; `fare_attributes.txt`; `fare_rules.txt`; `fare_products.txt`; `fare_leg_rules.txt`; `fare_transfer_rules.txt`; `areas.txt`; `stop_areas.txt`; `shapes.txt`; `frequencies.txt`; `transfers.txt`; `pathways.txt`; `levels.txt`; `translations.txt`; `feed_info.txt`; `attributions.txt`.

Throughout this chapter, we will learn about the basic structure of a GTFS file and its tables. We will focus only on the required tables and the optional tables most often used by producers and consumers of these files.[12]

In this demonstration, we use a subset of a feed describing the public transport network of São Paulo, Brazil, produced by São Paulo Transporte (SPTrans)[13] and downloaded in October 2019. The feed contains the six required tables plus two widely used optional tables, `shapes.txt` and `frequencies.txt`, which gives a good overview of the GTFS format.

4.1.1 `agency.txt`

File used to list the transport operators/agencies running the system described by the feed. Although the term agency, instead of operators, is used, it is up to the feed producer to choose which institutions are listed in the table.

For example, imagine that multiple bus companies operate in a given location, but all schedule and fare planning is carried out by a single institution, either a transport agency or a specific public entity, which is also recognized by public transport users as the system operator. In this case, we should probably list the planning institution in the table.

Now imagine a scenario in which a local public transport agency transfers the operation of a multimodal system to several companies (using concession contracts, for example). Each one of these companies is responsible for planning the schedules and fares of trips/routes they operate, provided that certain pre-established parameters are followed. In this case, we would probably be better off listing the operators in the table, instead of the public transport agency.

Table 1 shows the `agency.txt` file of SPTrans' feed. We can see that the feed producers decided to list the company itself in the table, instead of the operators of buses and subway routes.

[12] For more information on the tables and columns not covered in this section, please check the official specification.

[13] Available at: https://www.sptrans.com.br/desenvolvedores/.

TABLE 1
`agency.txt` example

agency_id	agency_name	agency_url	agency_timezone	agency_lang
1	SPTRANS	http://www.sptrans.com.br/?versao=011019	America/Sao_Paulo	pt

Source: SPTrans.

It is important to note that, although we are presenting `agency.txt` in table format, the data should be formatted as a `.csv` file. That is, the values of each cell must be separated by commas, and the contents of each table row must be listed in a different row of the `.csv` file. The table above, for example, is formatted as follows:

```
agency_id,agency_name,agency_url,agency_timezone,agency_lang
1,SPTRANS,http://www.sptrans.com.br/?versao=011019,America/
Sao_Paulo,pt
```

For the sake of communicability and interpretability, the next examples in this chapter are also presented as tables. It is important to keep in mind, however, that these tables are structured as shown above.

4.1.2 `stops.txt`

File used to describe the stops in a public transport system. The points listed in this file may reference simple stops (such as bus stops), stations, platforms, station entrances and exits etc. Table 2 shows the `stops.txt` of SPTrans' feed.

TABLE 2
`stops.txt` example

stop_id	stop_name	stop_desc	stop_lat	stop_lon
706325	Parada 14 Bis B/C	Viad. Dr. Plínio De Queiroz, 901	-23.55593	-46.65011
810602	R. Sta. Rita, 56	Ref.: R. Bresser / R. João Boemer	-23.53337	-46.61229
910776	Av. Do Estado, 5854	Ref.: Rua Dona Ana Néri	-23.55896	-46.61520
1010092	Parada Caetano Pinto	Av. Rangel Pestana, 1249 Ref.: Rua Caetano Pinto/rua Prof. Batista De Andrade	-23.54615	-46.62218
1010093	Parada Piratininga	Av. Rangel Pestana, 1479 Ref.: Rua Monsenhor Andrade	-23.54509	-46.62006
1010099	R. Xavantes, 612	Ref.: Rua Joli	-23.53545	-46.61368

Source: SPTrans.

The columns stop_id and stop_name identify each stop, but fulfill different roles. The purpose of stop_id is to identify relationships between this table and other tables that compose the feed (as we will later see in the stop_times.txt file, for example). Meanwhile, the column stop_name serves as an identifier that should be easily recognized by the passengers, thus usually assuming values of station names, points of interest or addresses (as in the case of SPTrans' feed).

The stop_desc column, present in SPTrans' feed, is optional and allows feed producers to add a description of each stop and its surroundings. Finally, stop_lat and stop_lon associate each stop to a point in space with its latitude and longitude geographic coordinates.

Two of the optional columns not present in this stops.txt table are location_type and parent_station. The location_type column is used to indicate the type of location that each point refers to. When not explicitly set, all points are interpreted as public transport stops, but distinct values can be used to distinguish a stop (location_type = 0) from a station (location_type = 1) or a boarding area (location_type = 2), for example. The parent_station column, on the other hand, is used to describe hierarchical relationships between two points. When describing a boarding area, for example, the feed producer must list the stop/platform that this area refers to, and when describing a stop/platform the producer can optionally list the station that it belongs to.

4.1.3 routes.txt

File used to describe the routes that run in a public transport system. Table 3 shows the routes.txt of SPTrans' feed.

TABLE 3
routes.txt example

route_id	agency_id	route_short_name	route_long_name	route_type
CPTM L07	1	CPTM L07	JUNDIAI - LUZ	2
CPTM L08	1	CPTM L08	AMADOR BUENO - JULIO PRESTES	2
CPTM L09	1	CPTM L09	GRAJAU - OSASCO	2
CPTM L10	1	CPTM L10	RIO GRANDE DA SERRA - BRÁS	2
CPTM L11	1	CPTM L11	ESTUDANTES - LUZ	2
CPTM L12	1	CPTM L12	CALMON VIANA - BRAS	2

Source: SPTrans.

As in the case of stops.txt, the routes.txt table also includes different columns to distinguish between the identifier of each route (route_id) and their names. In this case, however, there are two distinct name columns:

route_short_name and route_long_name. The first refers to the name of the route commonly recognized by passengers, while the second tends to be a more descriptive name. SPTrans, for example, has chosen to highlight the start and endpoints of each route in the latter column. We can also note that the same values are repeated in both route_id and route_short_name, which is neither required nor forbidden – in this case, the feed producer decided that the route names could satisfactorily work as identifiers because they are reasonably short and unique.

The agency_id column works as the key column that links the routes to the data described in agency.txt, and it indicates the agency responsible for operating each route – in this case the agency with id 1 (SPTrans itself). This column is optional in the case of feeds containing a single agency, but required otherwise. Using a feed describing a multimodal system with a subway corridor and several bus lines as an example, a possible configuration of routes.txt could associate the subway routes to the subway operator and the bus routes to the agency/company responsible for planning the bus schedules.

The route_type column is used to describe the transport mode of each route. The above example lists rail lines, whose corresponding numeric value is 2. The corresponding values of other transport modes are listed in the specification.

4.1.4 trips.txt

File used to describe the trips that compose the system. The trip is the basic unit of movement in the GTFS format: each trip is associated with a public transport route (route_id), with a service that operates on certain days of the week (as we will later cover in calendar.txt) and with a spatial trajectory (as we will later cover in shapes.txt). Table 4 shows the trips.txt of SPTrans' feed.

TABLE 4
trips.txt example

trip_id	route_id	service_id	trip_headsign	direction_id	shape_id
CPTM L07-0	CPTM L07	USD	JUNDIAI	0	17846
CPTM L07-1	CPTM L07	USD	LUZ	1	17847
CPTM L08-0	CPTM L08	USD	AMADOR BUENO	0	17848
CPTM L08-1	CPTM L08	USD	JULIO PRESTES	1	17849
CPTM L09-0	CPTM L09	USD	GRAJAU	0	17850
CPTM L09-1	CPTM L09	USD	OSASCO	1	17851

Source: SPTrans.

The `trip_id` column identifies the trips described in the table, just as the `route_id` references a route described in `routes.txt`. The `service_id` column identifies the services that determine the days of the week that each trip runs on (weekdays, weekends, a mix of both etc.), described in detail in `calendar.txt`. The rightmost column in the example above is `shape_id`, which identifies the spatial trajectory of each trip, described in detail in the `shapes.txt` file.

The two remaining columns, `trip_headsign` and `direction_id`, are optional and should be used to describe the direction/destination of the trip. The first, `trip_headsign`, is used to report the text that appears on the vehicle headsign (in the case of buses, for example) or on information panels (such as in subway and rail stations) highlighting the destination of the trip. The `direction_id` column is often used in conjunction with `trip_headsign` to distinguish the direction of each trip, where `0` represents one direction and `1` the opposite one. In our example, the first two rows describe trips that refer to the same public transport route (`CPTM L07`), but in opposite directions: one runs towards Jundiaí, and the other towards Luz.

4.1.5 `calendar.txt`

File used to describe the different service calendars in a public transport system, listing the set of days of the week in which trips may occur. Each service is also associated to an interval, with a start and an end date, within which the service operates. Table 5 shows the `calendar.txt` of SPTrans' feed.

TABLE 5
`calendar.txt` example

service_id	monday	tuesday	wednesday	thursday	friday	saturday	sunday	start_date	end_date
USD	1	1	1	1	1	1	1	20080101	20200501
U__	1	1	1	1	1	0	0	20080101	20200501
US_	1	1	1	1	1	1	0	20080101	20200501
_SD	0	0	0	0	0	1	1	20080101	20200501
__D	0	0	0	0	0	0	1	20080101	20200501
_S_	0	0	0	0	0	1	0	20080101	20200501

Source: SPTrans.

The column `service_id` identifies each service described in the table. As shown earlier, this identifier is also used in the `trips.txt`, where it associates each trip to a particular service.

The `monday, tuesday, wednesday, thursday, friday, saturday` and `sunday` columns are used to list the days of the week in which each service operates. A value of 1 means that the service operates on that day, while a value of 0 means that it does not. In the example above, the USD service operates on every day of the week and the service U__ operates only on business days.

Finally, the columns start_date and end_date delimit the interval within which the services are valid. Dates in GTFS files must always be formatted using the YYYYMMDD format: the first four numbers define the year, the subsequent two define the month and the last two, the day. The value 20220428, for example, represents the 28th of April 2022.

4.1.6 shapes.txt

File used to describe the spatial trajectory of each trip in the system. This file is optional, but feed producers are strongly encouraged to include it in their GTFS files. Table 6 shows the shapes.txt of SPTrans' feed.

TABLE 6
shapes.txt example

shape_id	shape_pt_lat	shape_pt_lon	shape_pt_sequence
17846	-23.53517	-46.63535	1
17846	-23.53513	-46.63548	2
17846	-23.53494	-46.63626	3
17846	-23.53473	-46.63710	4
17846	-23.53466	-46.63735	5
17846	-23.53416	-46.63866	6

Source: SPTrans.

The column shape_id identifies each shape and links each trip to its spatial trajectory in the trips.txt table. Unlike all the other identifiers we have seen so far, however, shape_id is repeated in several table rows. This is because each shape is defined by a sequence of spatial points, whose geographic coordinates are described with the shape_pt_lat and shape_pt_lon columns. The shape_pt_sequence column lists the sequence in which the points connect to form the shape. Values listed in this column must increase along the path.

4.1.7 stop_times.txt

File used to describe the timetable of each trip, including the arrival and departure times at each stop. How this table should be formatted depends on whether the feed contains a frequencies.txt table or not, a detail that we will cover later. For now, we will look at the stop_times.txt of SPTrans' feed, which also includes a frequencies.txt, in table 7.

TABLE 7
stop_times.txt example

trip_id	arrival_time	departure_time	stop_id	stop_sequence
CPTM L07-0	04:00:00	04:00:00	18940	1
CPTM L07-0	04:08:00	04:08:00	18920	2
CPTM L07-0	04:16:00	04:16:00	18919	3
CPTM L07-0	04:24:00	04:24:00	18917	4
CPTM L07-0	04:32:00	04:32:00	18916	5
CPTM L07-0	04:40:00	04:40:00	18965	6

Source: SPTrans.

The trip whose timetable is being described is identified by the trip_id column. Similarly to what happens in shapes.txt, the same trip_id appears in several rows. This is because, just like a trip trajectory is composed of a sequence of spatial points, a timetable consists of a sequence of departure/arrival times at various public transport stops.

The next columns, arrival_time, departure_time and stop_id, describe the schedule of each trip, associating an arrival and a departure time to each visited stop. The time columns must be formatted using the HH:MM:SS format, with the first two numbers defining the hour, the subsequent two the minutes and the last two, the seconds. This format also accepts hour values greater than 24: for example, if a trip departs at 11 pm but it only arrives at a given station at 1 am of the next day, the arrival time must be listed as 25:00:00, not 01:00:00. The stop_id column associates the arrival and departure times with a stop described in stops.txt and the stop_sequence column lists the sequence in which the stops connect to form the trip schedule. The values of this last column must always increase along the trip.

It is worth highlighting here the difference between shapes.txt and stop_times.txt. Although both tables present some spatial information of the trips, they do it in different ways. The stop_times.txt table lists the sequence of stops and times that make up a schedule, but says nothing about the trajectory traveled between the stops. shapes.txt, on the other hand, describes the detailed trajectory of a trip, but does not specify where the public transport stops are located. Combined, the information from the two tables allows one to understand both the schedule of each trip and the spatial trajectory between stops.

4.1.8 frequencies.txt

Optional file used to describe the frequency of each trip within different time intervals of a day. Table 8 shows the frequency.txt of SPTrans' feed.

TABLE 8
`frequencies.txt` example

trip_id	start_time	end_time	headway_secs
CPTM L07-0	04:00:00	04:59:00	720
CPTM L07-0	05:00:00	05:59:00	360
CPTM L07-0	06:00:00	06:59:00	360
CPTM L07-0	07:00:00	07:59:00	360
CPTM L07-0	08:00:00	08:59:00	360
CPTM L07-0	09:00:00	09:59:00	480

Source: SPTrans.

The trip whose frequency is being described is identified by the `trip_id` column. Again, the same identifier may appear in multiple observations. This is because the specification allows the same trip to have different frequencies throughout the day, such as at peak and off-peak hours, for example. Thus, each row refers to the frequency of a given trip within the time interval specified by the `start_time` and `end_time` columns.

Within this interval, the trip operates on regular headways specified in `headway_secs`. The headway is the time between trips that operate the same route. In the case of this table, this time must be specified in seconds. In the example above, we see a headway of `720` between 4 and 5 am, which indicates that the `CPTM L07-0` trip departs every 12 minutes within this interval.

Using `frequencies.txt` and `stop_times.txt` together

It is important to understand how the presence of a `frequencies.txt` table changes the specification of `stop_times.txt`. As we can see in the `stop_times.txt` example, the `CPTM L07-0` trip departs from the first stop at 4 am and arrives at the second at 4:08 am. The arrival and departure times at a given stop, however, cannot be specified more than once for each trip, even though the headway set in `frequencies.txt` defines that this trip departs every 12 minutes from 4 am to 5 am. If that's the case, how can we set the schedule of trips departing at 4:12 am, 4:24 am, 4:36 am etc.?

If the frequency of a trip is specified in `frequencies.txt`, the timetable of this trip defined in `stop_times.txt` should be understood as a reference that describes the time between stops. In other words, the times defined in the `stop_times.txt` file should not be interpreted "as is". For example, the timetable of trip `CPTM L07-0` establishes that the journey between the first and second stop takes 8 minutes to complete, which is the same travel time between the second and third stops as well. Thus, a trip departing from the first stop at 4 am arrives at the second at 4:08 am and at the third at 4:16 am. The next trip, which departs from the first stop at 4:12 am, arrives at the second stop at 4:20 am and at the third at 4:28 am.

To describe the same trips in stop_times.txt without making a frequencies.txt table, one could add a suffix that would identify each trip of route CPTM L07 in direction 0 throughout the day. The trip with id CPTM L07-0_1, for example, would be the first trip of the day heading towards direction 0 and would depart from the first stop at 4 am and arrive at the second at 4:08 am. The CPTM L07-0_2 trip, on the other hand, would be the second trip of the day and would depart from the first stop at 04:12 am and arrive at the second at 4:20 am. The rest of the trips would follow the same pattern. Each one of these trips would also need to be added to trips.txt, as well as to any other tables that use trip_id.

Another variable that changes how frequencies.txt affects the timetables in stop_times.txt is the optional column exact_times. When it assumes the value of 0 (or when it is missing from the feed, as in the case of the SPTrans' GTFS file) it indicates that the trip does not necessarily follow a fixed schedule over the time interval. Instead, operators try to maintain a predetermined headway during the interval. Using the same example of a trip whose headway is 12 minutes between 4 am and 5 am, this would mean that the first departure does not necessarily happen at 4 am, the second at 4:12 am, and so on. The first trip can, for example, leave at 4:02 am. The second, at 4:14 am or 4:13 am, etc. Meanwhile, an exact_times value of 1 must be used to define a schedule that always follows the exact same headway. This is an equivalent and more concise way of defining several similar trips departing at different times in stop_times.txt (as shown in the previous paragraph).

4.2 Finding GTFS data for Brazilian cities

GTFS data from cities all over the world can be downloaded with the {tidytransit} R package or on the Transitland website. In Brazil, several cities use GTFS data to plan and operate their transport systems. In many cases, however, the data is owned by private companies and operators and is not publicly available. As a result, GTFS data in Brazil is seldom openly available, which goes against the public interest and against good practices of data management and governance. Table 9 lists some of the few Brazilian cities that make their GTFS feeds openly available to the public.[14]

[14] As of the publication date of this book.

TABLE 9
Openly available GTFS data in Brazil

City	Source	Information
Belo Horizonte	Belo Horizonte's Transport and Traffic Company (BHTrans).	Open data: conventional transport network and supplementary network.
Fortaleza	Fortaleza's Urban Transport Company (Etufor).	Open data, available at: https://dados.fortaleza.ce.gov.br/dataset/gtfs.
Fortaleza	Fortaleza's Subway (Metrofor).	Open data, available at: https://www.metrofor.ce.gov.br/gtfs/.
Porto Alegre	Porto Alegre's Transport and Traffic Public Company (EPTC).	Open data, available at: https://dadosabertos.poa.br/dataset/gtfs.
Rio de Janeiro	Municipal Department of Transport (SMTR).	Open data, available at: https://www.data.rio/datasets/gtfs-do-rio-de-janeiro/about.
São Paulo	São Paulo's Metropolitan Urban Transport Company (EMTU).	Download available at: https://www.emtu.sp.gov.br/emtu/dados-abertos/dados-abertosprincipal/gtfs.fss.
São Paulo	SPTrans.	Download available at: https://www.sptrans.com.br/desenvolvedores/perfil-desenvolvedor/. Registration required.

Authors' elaboration.
Obs.: The GTFS data provided by SMTR does not include train and subway data.

CHAPTER 5

5 GTFS DATA MANIPULATION AND VISUALIZATION

GTFS data is frequently used in various types of analyses that involve a few common elements. The AOP team has developed the `{gtfstools}` R package, which provides several functions that help tackling repetitive tasks and operations and facilitate feed manipulation and exploration.

In this chapter, we'll go through some of the most frequently used package features. To do this, we will use a sample of the SPTrans feed presented in the previous chapter, and which is included in the package installation.

5.1 Reading and manipulating GTFS files

Reading GTFS files with `{gtfstools}` is done with the `read_gtfs()` function, which receives a string with the file path. The package represents a feed as a list of `data.tables`, a high-performance version of `data.frames`. Throughout this chapter, we will refer to this list of tables as a *GTFS object*. By default, the function reads all `.txt` tables in the feed:

```r
# loads the package
library(gtfstools)

# points to path of the sample gtfs data installed in {gtfstools}
path <- system.file("extdata/spo_gtfs.zip", package = "gtfstools")

# reads the gtfs
gtfs <- read_gtfs(path)

# checks the tables inside the gtfs object
names(gtfs)
```

```
[1] "agency"      "calendar"    "frequencies" "routes"      "shapes"
[6] "stop_times"  "stops"       "trips"
```

We can see that each `data.table` within the GTFS object is named according to the table it represents, without the `.txt` extension. This configuration allows us to select and manipulate each table individually. The code below, for example, lists the first 6 rows of the `trips` table:

```
head(gtfs$trips)
```

```
   route_id  service_id       trip_id  trip_headsign  direction_id  shape_id
1: CPTM L07         USD   CPTM   L07-0        JUNDIAI             0     17846
2: CPTM L07         USD   CPTM   L07-1            LUZ             1     17847
3: CPTM L08         USD   CPTM   L08-0   AMADOR BUENO             0     17848
4: CPTM L08         USD   CPTM   L08-1  JULIO PRESTES             1     17849
5: CPTM L09         USD   CPTM   L09-0         GRAJAU             0     17850
6: CPTM L09         USD   CPTM   L09-1         OSASCO             1     17851
```

Tables within a GTFS object can be easily manipulated using the {dplyr} or {data.table} packages, for example. In this book, we opted to use the {data.table} syntax. This package offers several useful features, primarily for manipulating tables with a large number of records, such as updating columns by reference, very fast row subsets and efficient data aggregation.[15] For example, we can use the code below to add 100 seconds to all the headways listed in the frequencies table and later reverse this change:

```
# saves original headways
original_headway <- gtfs$frequencies$headway_secs
head(gtfs$frequencies, 3)
```

```
      trip_id start_time end_time headway_secs
1: CPTM L07-0   04:00:00 04:59:00          720
2: CPTM L07-0   05:00:00 05:59:00          360
3: CPTM L07-0   06:00:00 06:59:00          360
```

```
# updates the headways
gtfs$frequencies[, headway_secs := headway_secs + 100]
head(gtfs$frequencies, 3)
```

```
      trip_id start_time end_time headway_secs
1: CPTM L07-0   04:00:00 04:59:00          820
2: CPTM L07-0   05:00:00 05:59:00          460
3: CPTM L07-0   06:00:00 06:59:00          460
```

```
# restores the original headway
gtfs$frequencies[, headway_secs := original_headway]
head(gtfs$frequencies, 3)
```

```
      trip_id start_time end_time headway_secs
1: CPTM L07-0   04:00:00 04:59:00          720
2: CPTM L07-0   05:00:00 05:59:00          360
3: CPTM L07-0   06:00:00 06:59:00          360
```

[15] More details on {data.table} usage and syntax are available at: https://rdatatable.gitlab.io/data.table/index.html.

After editing a GTFS object in R, we often want to use the processed GTFS to perform different analyses. In order to do this, we frequently need the GTFS file in `.zip` format again, and not as a list of tables in an R session. To transform GTFS objects that exist in an R session into GTFS files saved to disk, `{gtfstools}` includes the `write_gtfs()` function. To use this function, we only need to pass the object that should be written to disk and the file path where it should be written to:

```
# points to the path where the GTFS should be written to
export_path <- tempfile("new_gtfs", fileext = ".zip")

# writes the GTFS to the path
write_gtfs(gtfs, path = export_path)

# lists files within the feed
zip::zip_list(export_path)[, c("filename", "compressed_size",
"timestamp")]
```

```
         filename compressed_size           timestamp
1      agency.txt             112 2023-06-16 15:38:14
2    calendar.txt             129 2023-06-16 15:38:14
3 frequencies.txt            2381 2023-06-16 15:38:14
4      routes.txt             659 2023-06-16 15:38:14
5      shapes.txt          160470 2023-06-16 15:38:14
6  stop_times.txt            7907 2023-06-16 15:38:14
7       stops.txt           18797 2023-06-16 15:38:14
8       trips.txt             717 2023-06-16 15:38:14
```

5.2 Calculating trip speed

GTFS files are often used in public transport routing applications and to inform the timetable of different routes in a given region to potential passengers. Feeds must, therefore, accurately describe the schedule and the operational speed of public transport trips.

To calculate the average speed of the trips described in a feed, `{gtfstools}` package includes the function `get_trip_speed()`. By default, the function returns the speed (in km/h) of all trips included in the feed, but one can choose to calculate the speed of selected trips with the `trip_id` parameter:

```
# calculates the speeds of all trips
speeds <- get_trip_speed(gtfs)

head(speeds)
```

```
     trip_id origin_file      speed
1: 2002-10-0      shapes   8.952511
2: 2105-10-0      shapes  10.253365
3: 2105-10-1      shapes   9.795292
4: 2161-10-0      shapes  11.182534
5: 2161-10-1      shapes  11.784458
6: 4491-10-0      shapes  13.203560
```

```
nrow(speeds)
```

```
[1] 36
```

```
# calculates the speeds of two specific trips
speeds <- get_trip_speed(gtfs, trip_id = c("CPTM L07-0",
"2002-10-0"))

speeds
```

```
      trip_id origin_file     speed
1:  2002-10-0      shapes  8.952511
2: CPTM L07-0      shapes 26.787768
```

To calculate the speed of a trip, we need to know its length and how long it takes to travel from its first to its last stop. Behind the scenes, `get_trip_speed()` uses two other functions from `{gtfstools}` toolset: `get_trip_length()` and `get_trip_duration()`. The usage of both is very similar to what has been shown before, returning the length/duration of all trips by default or of a few selected trips if desired. Below, we show their default behavior:

```
# calculates the length of all trips
lengths <- get_trip_length(gtfs, file = "shapes")

head(lengths)
```

```
      trip_id   length origin_file
1: CPTM L07-0 60.71894      shapes
2: CPTM L07-1 60.71894      shapes
3: CPTM L08-0 41.79037      shapes
4: CPTM L08-1 41.79037      shapes
5: CPTM L09-0 31.88906      shapes
6: CPTM L09-1 31.88906      shapes
```

```
# calculates the duration of all trips
durations <- get_trip_duration(gtfs)

head(durations)
```

```
     trip_id duration
1: 2002-10-0       48
2: 2105-10-0      108
3: 2105-10-1      111
4: 2161-10-0       94
5: 2161-10-1       93
6: 4491-10-0       69
```

Just as `get_trip_speed()` returns speeds in km/h by default, `get_trip_length()` returns lengths in km and `get_trip_duration()` returns the duration in minutes. These units can be adjusted with the `unit` parameter, present in all three functions.

5.3 Combining and filtering feeds

The tasks of processing and manipulating GTFS files are often performed manually, which may increase the chances of leaving minor inconsistencies or errors in the data. A common issue in some GTFS feeds is the presence of duplicate records in the same table. SPTrans' feed, for example, contains duplicate records both in `agency.txt` and in `calendar.txt`:

```
gtfs$agency
```

```
   agency_id agency_name                               agency_url
1:         1     SPTRANS http://www.sptrans.com.br/?versao=011019
2:         1     SPTRANS http://www.sptrans.com.br/?versao=011019
     agency_timezone agency_lang
1: America/Sao_Paulo          pt
2: America/Sao_Paulo          pt
```

```
gtfs$calendar
```

```
   service_id monday tuesday wednesday thursday friday saturday sunday
1:        USD      1       1         1        1      1        1      1
2:        U__      1       1         1        1      1        0      0
3:        US_      1       1         1        1      1        1      0
4:        _SD      0       0         0        0      0        1      1
5:        __D      0       0         0        0      0        0      1
6:        _S_      0       0         0        0      0        1      0
```

```
 7:        USD      1       1         1        1      1        1      1
 8:        U__      1       1         1        1      1        0      0
 9:        US_      1       1         1        1      1        1      0
10:        _SD      0       0         0        0      0        1      1
11:        __D      0       0         0        0      0        0      1
12:        _S_      0       0         0        0      0        1      0

    start_date   end_date
 1: 2008-01-01 2020-05-01
 2: 2008-01-01 2020-05-01
 3: 2008-01-01 2020-05-01
 4: 2008-01-01 2020-05-01
 5: 2008-01-01 2020-05-01
 6: 2008-01-01 2020-05-01
 7: 2008-01-01 2020-05-01
 8: 2008-01-01 2020-05-01
 9: 2008-01-01 2020-05-01
10: 2008-01-01 2020-05-01
11: 2008-01-01 2020-05-01
12: 2008-01-01 2020-05-01
```

{gtfstools} includes the remove_duplicates() function to keep only unique entries in all tables of the feed. This function takes a GTFS object as input and returns the same object without duplicates:

```
no_dups_gtfs <- remove_duplicates(gtfs)

no_dups_gtfs$agency
```

```
   agency_id agency_name                              agency_url
1:         1     SPTRANS http://www.sptrans.com.br/?versao=011019
     agency_timezone agency_lang
1: America/Sao_Paulo          pt
```

```
no_dups_gtfs$calendar
```

```
   service_id monday tuesday wednesday thursday friday saturday sunday
1:        USD      1       1         1        1      1        1      1
2:        U__      1       1         1        1      1        0      0
3:        US_      1       1         1        1      1        1      0
4:        _SD      0       0         0        0      0        1      1
5:        __D      0       0         0        0      0        0      1
6:        _S_      0       0         0        0      0        1      0
```

```
   start_date   end_date
1: 2008-01-01 2020-05-01
2: 2008-01-01 2020-05-01
3: 2008-01-01 2020-05-01
4: 2008-01-01 2020-05-01
5: 2008-01-01 2020-05-01
6: 2008-01-01 2020-05-01
```

We often have to deal with multiple feeds describing the same study area. For example, when the bus and the rail systems of a single city are described in separate GTFS files. In such cases, we may want to merge both files into a single feed to reduce the data processing effort. To help us with that, `{gtfstools}` includes the `merge_gtfs()` function. The example below shows the output of merging SPtrans' feed (without duplicate entries) with EPTC's feed:

```
# reads Porto Alegre’s GTFS
poa_path <- system.file("extdata/poa_gtfs.zip", package =
"gtfstools")
poa_gtfs <- read_gtfs(poa_path)

poa_gtfs$agency
```

```
   agency_id
1:      EPTC
                                    agency_name
1: Empresa Publica de Transportes e Circulação
               agency_url       agency_timezone
1: http://www.eptc.com.br    America/Sao_Paulo
   agency_lang agency_phone
1:          pt          156
                                                agency_fare_url
1: http://www2.portoalegre.rs.gov.br/eptc/default.php?p_secao=155
```

```
no_dups_gtfs$agency
```

```
   agency_id agency_name                               agency_url
1:         1     SPTRANS http://www.sptrans.com.br/?versao=011019
     agency_timezone agency_lang
1: America/Sao_Paulo          pt
```

```
# combines Porto Alegre’s and São Paulo’s GTFS objects
combined_gtfs <- merge_gtfs(no_dups_gtfs, poa_gtfs)

# check results
combined_gtfs$agency
```

```
   agency_id                                agency_name
1:         1                                    SPTRANS
2:      EPTC Empresa Publica de Transportes e Circulação
                                 agency_url   agency_timezone agency_lang
1: http://www.sptrans.com.br/?versao=011019 America/Sao_Paulo          pt
2:                   http://www.eptc.com.br America/Sao_Paulo          pt
   agency_phone                                                agency_fare_url
1:
2:          156 http://www2.portoalegre.rs.gov.br/eptc/default.php?p_secao=155
```

We can see that the tables of both feeds are combined into a single one. This is the case when two (or more) GTFS objects contain the same table (`agency`, in the example). When a particular table is present in only one of the feeds, the function copies this table to the output. That's the case of the `frequencies` table, in our example, which exists only in SPTrans' feed:

```
names(poa_gtfs)
```

```
[1] "agency"    "calendar"    "routes"     "shapes"    "stop_times"
[6] "stops"     "trips"
```

```
names(no_dups_gtfs)
```

```
[1] "agency"      "calendar"    "frequencies" "routes"   "shapes"
[6] "stop_times"  "stops"       "trips"
```

```
names(combined_gtfs)
```

```
[1] "agency"      "calendar"    "frequencies" "routes"    "shapes"
[6] "stop_times"  "stops"       "trips"
```

```
identical(no_dups_gtfs$frequencies, combined_gtfs$frequencies)
```

```
[1] TRUE
```

Filtering feeds to keep only a few entries within each table is another operation that frequently comes up when dealing with GTFS data. Feeds are often used to describe large-scale public transport networks, which may result in complex and slow data manipulation, analysis and sharing. Thus, planners and researchers often work with feeds' subsets. If we want to measure the performance of a transport network during the morning peak, for example, we can filter our GTFS data to keep only the observations related to trips that run within this period.

{gtfstools} includes lots of functions to filter GTFS data. They are:

- filter_by_agency_id();
- filter_by_route_id();
- filter_by_service_id();
- filter_by_shape_id();
- filter_by_stop_id();
- filter_by_trip_id();
- filter_by_route_type();
- filter_by_weekday();
- filter_by_time_of_day(); and
- filter_by_sf().

5.3.1 Filtering by identifiers

The seven first functions from the above list work very similarly. They take as input a vector of identifiers and return a GTFS object whose table entries are related to the specified ids. The example below demonstrates this functionality with filter_by_trip_id():

```
# checks pre-filter object size
utils::object.size(gtfs)
```

```
864568 bytes
```

```
head(gtfs$trips[, .(trip_id, trip_headsign, shape_id)])
```

```
      trip_id trip_headsign shape_id
1: CPTM L07-0       JUNDIAI    17846
2: CPTM L07-1           LUZ    17847
3: CPTM L08-0  AMADOR BUENO    17848
4: CPTM L08-1 JULIO PRESTES    17849
5: CPTM L09-0        GRAJAU    17850
6: CPTM L09-1        OSASCO    17851
```

```
# keeps entries related to the two specified ids
filtered_gtfs <- filter_by_trip_id(
  gtfs,
  trip_id = c("CPTM L07-0", "CPTM L07-1")
)

# checks post-filter object size
utils::object.size(filtered_gtfs)
```

```
71592 bytes
```

```r
head(filtered_gtfs$trips[, .(trip_id, trip_headsign, shape_id)])
```

```
       trip_id trip_headsign shape_id
1: CPTM L07-0        JUNDIAI    17846
2: CPTM L07-1            LUZ    17847
```

```r
unique(filtered_gtfs$shapes$shape_id)
```

```
[1] "17846" "17847"
```

We can see from the code snippet above that the function not only filters trips, but all other tables containing a column that relates to `trip_id` in any way. The shapes of trips `CPTM L07-0` and `CPTM L07-1`, for example, are respectively described by `shape_ids 17846` and `17847`. Therefore, these are the only shape identifiers kept in the filtered GTFS.

The function also supports the opposite behavior: instead of keeping the entries related to the specified identifiers, we can drop them. To do this, we need to set the `keep` argument to `FALSE`:

```r
# removes entries related to two trips from the feed
filtered_gtfs <- filter_by_trip_id(
  gtfs,
  trip_id = c("CPTM L07-0", "CPTM L07-1"),
  keep = FALSE
)

head(filtered_gtfs$trips[, .(trip_id, trip_headsign, shape_id)])
```

```
       trip_id       trip_headsign shape_id
1: CPTM L08-0        AMADOR BUENO    17848
2: CPTM L08-1       JULIO PRESTES    17849
3: CPTM L09-0              GRAJAU    17850
4: CPTM L09-1              OSASCO    17851
5: CPTM L10-0 RIO GRANDE DA SERRA    17852
6: CPTM L10-1                BRÁS    17853
```

```r
head(unique(filtered_gtfs$shapes$shape_id))
```

```
[1] "17848" "17849" "17850" "17851" "17852" "17853"
```

We can see that the specified trips, as well as their shapes, are not present in the filtered GTFS anymore. The same logic, demonstrated here with `filter_by_trip_id()`, applies to the functions that filter GTFS objects by `agency_id`, `route_id`, `service_id`, `shape_id`, `stop_id` and `route_type`.

5.3.2 Filtering by day of the week and time of the day

Another common operation when dealing with GTFS data is subsetting feeds to keep services that only happen during certain times of the day or days of the week. To do this, the package includes the `filter_by_weekday()` and `filter_by_time_of_day()` functions.

`filter_by_weekday()` takes as input the days of the week whose services that operate on them should be kept (or dropped). The function also includes a `combine` parameter, which defines how multi-days filters should work. When this argument receives the value `"and"`, only services that operate on every single specified day are kept. When it receives the value `"or"`, services that operate on at least one of the days are kept:

```
# keeps services that operate on both saturday AND sunday
filtered_gtfs <- filter_by_weekday(
  no_dups_gtfs,
  weekday = c("saturday", "sunday"),
  combine = "and"
)

filtered_gtfs$calendar[, c("service_id", "sunday", "saturday")]
```

```
   service_id sunday saturday
1:        USD      1        1
2:        _SD      1        1
```

```
# keeps services that operate EITHER on saturday OR on sunday
filtered_gtfs <- filter_by_weekday(
  no_dups_gtfs,
  weekday = c("sunday", "saturday"),
  combine = "or"
)

filtered_gtfs$calendar[, c("service_id", "sunday", "saturday")]
```

```
   service_id sunday saturday
1:        USD      1        1
2:        US_      0        1
3:        _SD      1        1
4:        __D      1        0
5:        _S_      0        1
```

filter_by_time_of_day(), on the other hand, takes the beginning and the end of a time window and keeps (or drops) the entries related to the trips that run within this window. The behavior of this function depends on whether a frequencies table is included in the feed or not: the stop_times timetable of trips listed in frequencies must not be filtered, because, as previously mentioned, it works as a reference that describes the time between consecutive stops, and the departure and arrival times listed there should not be considered rigorously. If a trip is not listed in frequencies, however, its stop_times entries are filtered according to the specified time window. Let's see how the function works with some examples:

```
# keeps trips that run within the 5am to 6am window
filtered_gtfs <- filter_by_time_of_day(gtfs, from = "05:00:00",
to = "06:00:00")

head(filtered_gtfs$frequencies)
```

```
       trip_id start_time end_time headway_secs
1: CPTM L07-0   05:00:00 05:59:00          360
2: CPTM L07-1   05:00:00 05:59:00          360
3: CPTM L08-0   05:00:00 05:59:00          480
4: CPTM L08-1   05:00:00 05:59:00          480
5: CPTM L09-0   05:00:00 05:59:00          480
6: CPTM L09-1   05:00:00 05:59:00          480
```

```
head(filtered_gtfs$stop_times[, c("trip_id", "departure_time",
"arrival_time")])
```

```
       trip_id departure_time arrival_time
1: CPTM L07-0       04:00:00     04:00:00
2: CPTM L07-0       04:08:00     04:08:00
3: CPTM L07-0       04:16:00     04:16:00
4: CPTM L07-0       04:24:00     04:24:00
5: CPTM L07-0       04:32:00     04:32:00
6: CPTM L07-0       04:40:00     04:40:00
```

```
# save the frequencies table and remove it from the original gtfs
frequencies <- gtfs$frequencies
gtfs$frequencies <- NULL

filtered_gtfs <- filter_by_time_of_day(gtfs, from = "05:00:00",
to = "06:00:00")
```

```
head(filtered_gtfs$stop_times[, c("trip_id", "departure_time",
"arrival_time")])
```

```
     trip_id departure_time arrival_time
1: CPTM L07-0       05:04:00     05:04:00
2: CPTM L07-0       05:12:00     05:12:00
3: CPTM L07-0       05:20:00     05:20:00
4: CPTM L07-0       05:28:00     05:28:00
5: CPTM L07-0       05:36:00     05:36:00
6: CPTM L07-0       05:44:00     05:44:00
```

Filtering the `stop_times` table can work in two different ways. One is to keep trips that cross the specified time window intact. The other is to keep only the timetable entries that take place *inside* this window (default behavior). This behavior is controlled by the `full_trips` parameter, as shown below (please pay attention to the times and stops present in each example):

```
# keeps any trips that cross the 5am to 6am window intact
filtered_gtfs <- filter_by_time_of_day(
  gtfs,
  from = "05:00:00",
  to = "06:00:00",
  full_trips = TRUE
)

head(
  filtered_gtfs$stop_times[
    ,
    c("trip_id", "departure_time", "arrival_time", "stop_sequence")
  ]
)
```

```
     trip_id departure_time arrival_time stop_sequence
1: CPTM L07-0       04:00:00     04:00:00             1
2: CPTM L07-0       04:08:00     04:08:00             2
3: CPTM L07-0       04:16:00     04:16:00             3
4: CPTM L07-0       04:24:00     04:24:00             4
5: CPTM L07-0       04:32:00     04:32:00             5
6: CPTM L07-0       04:40:00     04:40:00             6
```

```
# keeps only the timetable entries that happen inside the 5am
# to 6am window
filtered_gtfs <- filter_by_time_of_day(
  gtfs,
  from = "05:00:00",
  to = "06:00:00",
  full_trips = FALSE
)

head(
  filtered_gtfs $stop_times[
    ,
    c("trip_id", "departure_time", "arrival_time", "stop_sequence")
  ]
)
```

```
      trip_id departure_time arrival_time stop_sequence
1: CPTM L07-0       05:04:00     05:04:00             9
2: CPTM L07-0       05:12:00     05:12:00            10
3: CPTM L07-0       05:20:00     05:20:00            11
4: CPTM L07-0       05:28:00     05:28:00            12
5: CPTM L07-0       05:36:00     05:36:00            13
6: CPTM L07-0       05:44:00     05:44:00            14
```

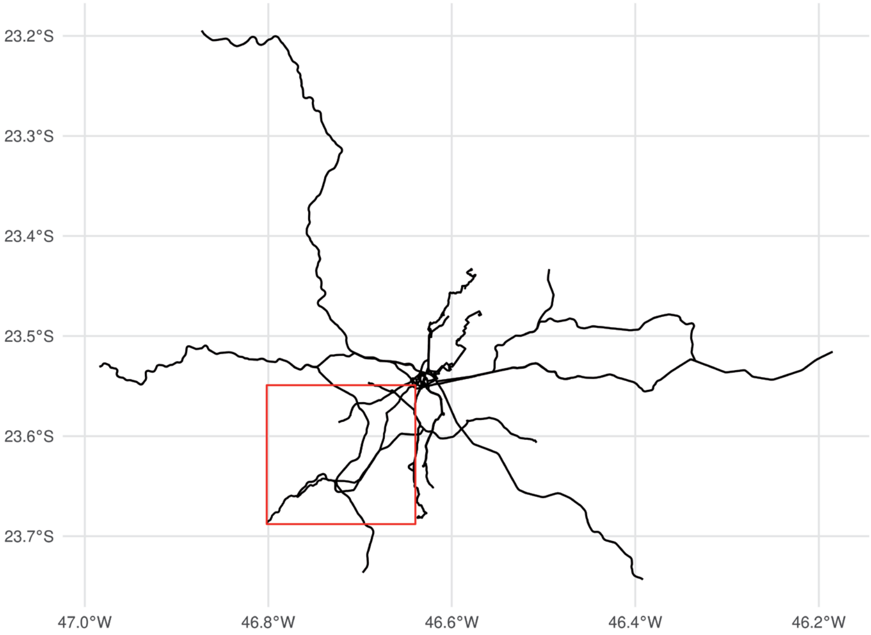
5.3.3 Filtering using a spatial extent

Finally, {gtfstools} also includes a function that allows one to filter a GTFS object using a spatial polygon. filter_by_sf() takes an sf/sfc object (spatial representation created by the {sf} package), or its bounding box, and keeps the entries related to trips selected by their position in relation to that spatial polygon. Although this might seem complicated, this filtering process is fairly easy to grasp once we illustrate it with an example. To demonstrate this function, we are going to filter SPTrans' feed using the bounding box of shape 68962. With the code snippet below we show the spatial distribution of unfiltered data along with the bounding box in red:

```
library(ggplot2)

# creates a polygon with the bounding box of shape 68962
shape_68962 <- convert_shapes_to_sf(gtfs, shape_id = "68962")
bbox <- sf::st_bbox(shape_68962)
bbox_geometry <- sf::st_as_sfc(bbox)

# creates a geometry with all the shapes described in the gtfs
all_shapes <- convert_shapes_to_sf(gtfs)
```

```
ggplot() +
  geom_sf(data = all_shapes) +
  geom_sf(data = bbox_geometry, fill = NA, color = "red") +
  theme_minimal()
```

FIGURE 5
Shapes spatial distribution overlayed by the bounding box of shape 68962

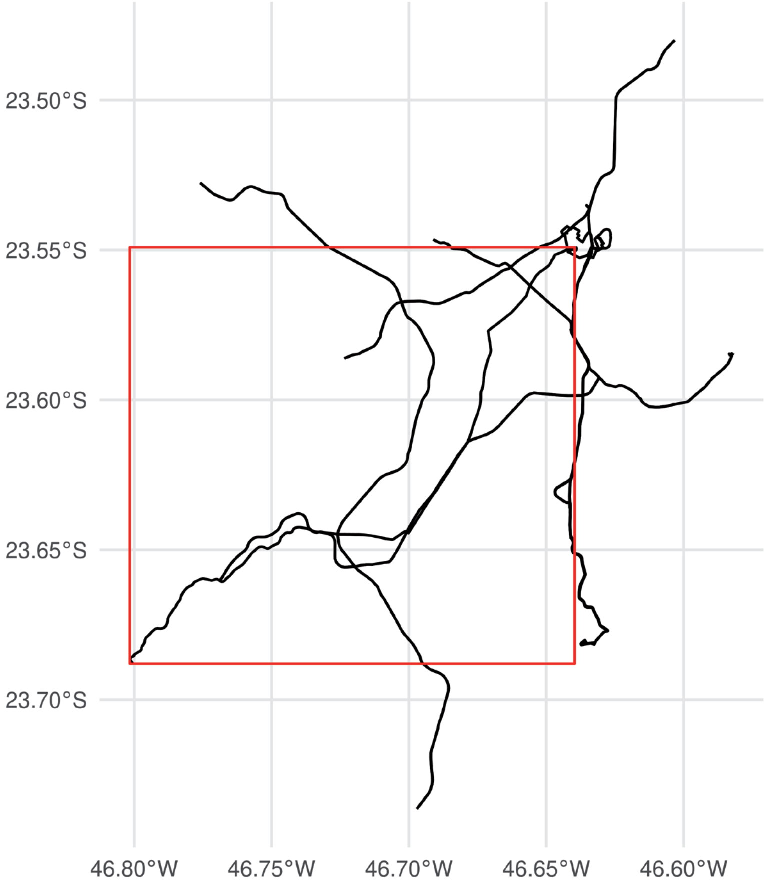


Source: Figure generated by the code snippet above.

Please note that we have used the `convert_shapes_to_sf()` function, also included in `{gtfstools}`, to convert the shapes described in the feed into a `sf` spatial object. By default, `filter_by_sf()` keeps all entries related to trips that intersect with the specified polygon:

```
filtered_gtfs <- filter_by_sf(gtfs, bbox)
filtered_shapes <- convert_shapes_to_sf(filtered_gtfs)

ggplot() +
  geom_sf(data = filtered_shapes) +
  geom_sf(data = bbox_geometry, fill = NA, color = "red") +
  theme_minimal()
```

FIGURE 6
Spatial distribution of shapes that intersect with the bounding box of shape 68962

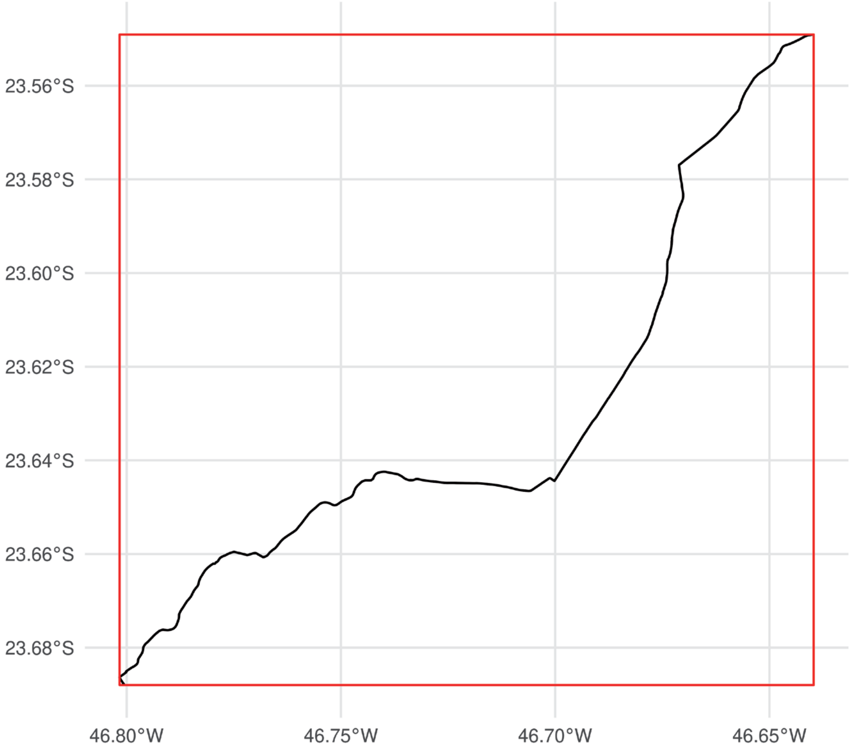


Source: Figure generated by the code snippet above.

We can, however, specify different spatial operations to filter the feed. The code below shows how we can keep the entries related to trips that are contained by the specified polygon:

```
filtered_gtfs <- filter_by_sf(gtfs, bbox, spatial_operation =
sf::st_contains)
filtered_shapes <- convert_shapes_to_sf(filtered_gtfs)

ggplot() +
  geom_sf(data = filtered_shapes) +
  geom_sf(data = bbox_geometry, fill = NA, color = "red") +
  theme_minimal()
```

FIGURE 7
Spatial distribution of shapes contained by the bounding box of shape 68962



Source: Figure generated by the code snippet above.

5.4 Validating GTFS data

Transport planners and researchers often want to assess the quality of the GTFS data they are producing or using in their analyses. Are feeds structured following the best practices adopted by the larger GTFS community? Are tables and columns adequately formatted? Is the information described by the feed reasonable (trip speeds, stop locations etc.)? These are some of the questions that may arise when dealing with GTFS data.

To answer these and other questions, `{gtfstools}` includes the `validate_gtfs()` function. This function works as a wrapper to MobilityData's Canonical GTFS Validator, which requires Java version 11 or higher to run.[16]

[16] For more information on how to check the installed version of Java in your computer and on how to install the required version, please check chapter 3.

Using `validate_gtfs()` is very simple. First, we need to download the validator. To do this, we use the `download_validator()` function, included in the package, which receives the path to the directory where the validator should be saved to and the version of the validator that should be downloaded (defaults to the latest available). The function returns the path to the downloaded validator:

```
tmpdir <- tempdir()

validator_path <- download_validator(tmpdir)
validator_path
```

```
[1] "/tmp/Rtmpf3LrBZ/gtfs-validator-v4.0.0.jar"
```

The second (and final) step consists in actually validating the GTFS data with `validate_gtfs()`. This function supports GTFS data in different formats: i) as a GTFS object in an R session; ii) as a path to a local GTFS file in `.zip` format; iii) as an URL pointing to a feed; or iv) as a directory containing unzipped GTFS tables. The function also takes a path to a directory where the validation result should be saved to and the path to the validator that should be used in the process. In the example below, we validate SPTrans' feed from its path:

```
output_dir <- tempfile("gtfs_validation")

validate_gtfs(
  path,
  output_path = output_dir,
  validator_path = validator_path
)

list.files(output_dir)
```

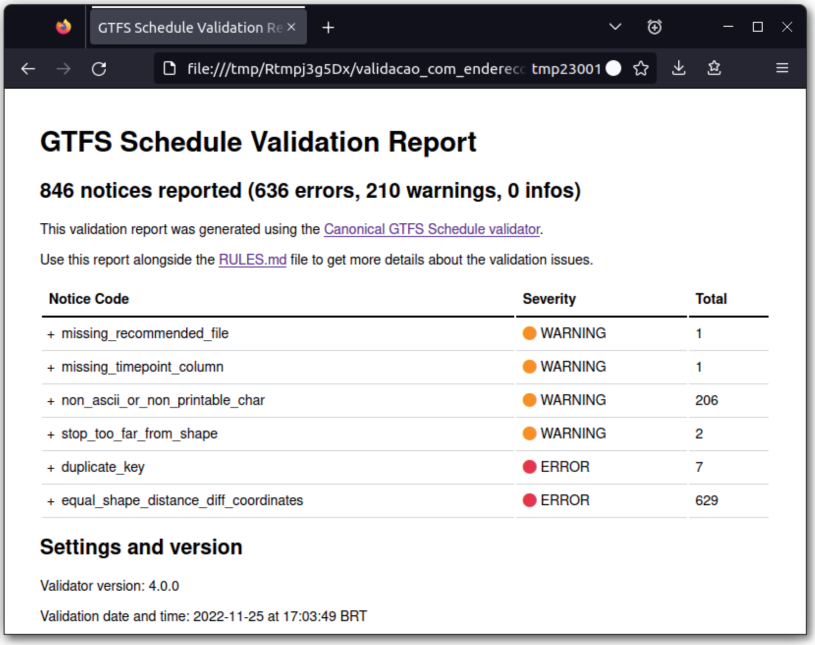
```
[1] "report.html"            "report.json"    "system_errors.json"
[4] "validation_stderr.txt"
```

We can see that the validation process generates a few output files:

- `report.html`, shown in figure 8, which summarizes the validation result in a nicely formatted HTML page (only available with validator version 3.1.0 or higher);
- `report.json`, which summarizes the same information, but in JSON format, which can be used to programatically parse and process the results;

- `system_errors.json`, which summarizes eventual system errors that may have happened during the validation process and may compromise the results; and
- `validation_stderr.txt`, which lists informative messages sent by the validator tool, including a list of the tests conducted, eventual error messages etc.[17]

FIGURE 8
Validation report example


GTFS Schedule Validation Re ×
+
file:///tmp/Rtmpj3g5Dx/validacao_com_enderecc tmp23001

GTFS Schedule Validation Report

846 notices reported (636 errors, 210 warnings, 0 infos)

This validation report was generated using the Canonical GTFS Schedule validator.

Use this report alongside the RULES.md file to get more details about the validation issues.

Notice Code	Severity	Total
+ missing_recommended_file	● WARNING	1
+ missing_timepoint_column	● WARNING	1
+ non_ascii_or_non_printable_char	● WARNING	206
+ stop_too_far_from_shape	● WARNING	2
+ duplicate_key	● ERROR	7
+ equal_shape_distance_diff_coordinates	● ERROR	629

Settings and version

Validator version: 4.0.0

Validation date and time: 2022-11-25 at 17:03:49 BRT

Authors' elaboration.
Obs.: Figure whose layout and texts could not be formatted due to the technical characteristics of the original files (Publiser's note).

5.5 `{gtfstools}` workflow example: spatial visualization of headways

We have shown in previous sections that `{gtfstools}` offers a large toolset to process and analyze GTFS files. The package, however, also includes many other functions that could not be shown in this book due to space constraints.[18]

[17] Informative messages may also be listed in the `validation_stdout.txt` file. Whether messages are listed in this file or in `validation_stderr.txt` depends on the validator version.

[18] The complete list of functions available in `{gtfstools}` can be checked at: https://ipeagit.github.io/gtfstools/reference/index.html.

In this final section of the chapter, we illustrate how to use the package to make more complex analyses. To do this, we present a workflow that combines various functions of `{gtfstools}` together to answer the following question: how are the times between vehicles operating the same route (the headways) spatially distributed in SPTrans' GTFS?

First, we need to define the scope of our analysis. In this example, we are only going to consider the services operating during the morning peak, between 7 am and 9 am, on a typical tuesday. Thus, we need to filter our feed:

```
gtfs <- read_gtfs(path)

# filters the GTFS
filtered_gtfs <- gtfs |>
  remove_duplicates() |>
  filter_by_weekday("tuesday") |>
  filter_by_time_of_day(from = "07:00:00", to = "09:00:00")

# checking the result
filtered_gtfs$frequencies[trip_id == "2105-10-0"]
```

```
     trip_id start_time end_time headway_secs
1: 2105-10-0   07:00:00 07:59:00          900
2: 2105-10-0   08:00:00 08:59:00         1200
```

```
filtered_gtfs$calendar
```

```
   service_id monday tuesday wednesday thursday friday saturday sunday
1:        USD      1       1         1        1      1        1      1
2:        U__      1       1         1        1      1        0      0
   start_date   end_date
1: 2008-01-01 2020-05-01
2: 2008-01-01 2020-05-01
```

Next, we need to calculate the headways within this time interval. This information can be found at the `frequencies` table, though there is a factor we have to pay attention to: each trip is associated to more than one headway, as shown above (one entry for the 7 am to 7:59 am interval and another for the 8 am to 8:59 am interval). To solve this, we are going to calculate the *average* headway from 7 am to 9 am.

The first few `frequencies` rows in SPTrans' feed seem to suggest that the headways are always associated to one-hour intervals, but this is neither a rule set in the official specification nor necessarily a practice adopted by other feed producers. Thus, we have to calculate the average headways weighted by the time duration of each headway. To do this, we need to multiply each headway by the size of the time interval during which it is valid, sum these multiplication results for each trip, and then divide this amount by the total time interval (two hours, in our case). To calculate the time intervals within which the headways are valid, we first use the `convert_time_to_seconds()` function to calculate the start and end time of the time interval in seconds and then subtract the latter by the former:

```r
filtered_gtfs <- convert_time_to_seconds(filtered_gtfs)

# check how the results look like for a particular trip id
filtered_gtfs$frequencies[trip_id == "2105-10-0"]
```

```
    trip_id start_time end_time headway_secs start_time_secs end_time_secs
1: 2105-10-0   07:00:00 07:59:00          900           25200         28740
2: 2105-10-0   08:00:00 08:59:00         1200           28800         32340
```

```r
filtered_gtfs$frequencies[, time_interval := end_time_secs -
start_time_secs]
```

Then we calculate the average headway:

```r
average_headway <- filtered_gtfs$frequencies[,
  .(average_headway = weighted.mean(x = headway_secs,
  w = time_interval)),
  by = trip_id
]

average_headway[trip_id == "2105-10-0"]
```

```
     trip_id average_headway
1: 2105-10-0            1050
```

```r
head(average_headway)
```

```
      trip_id average_headway
1: CPTM L07-0             360
2: CPTM L07-1             360
3: CPTM L08-0             300
4: CPTM L08-1             300
5: CPTM L09-0             240
6: CPTM L09-1             240
```

Now we need to generate each trip geometry and join this data to the average headways. To do this, we will use the get_trip_geometry() function, which returns the spatial geometries of the trips in the feed. This function allows us to specify which trips we want to generate the geometries of, so we are only going to apply the procedure to the trips present in the average headways table:

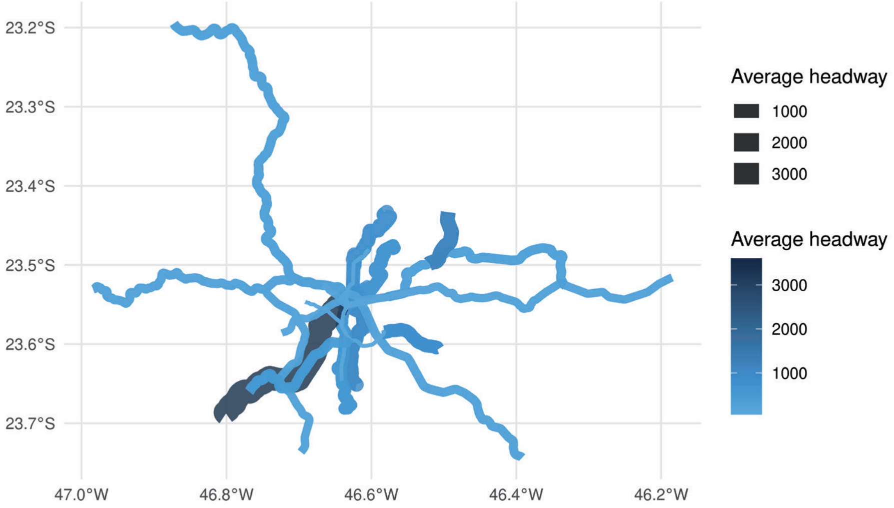
```
selected_trips <- average_headway$trip_id

geometries <- get_trip_geometry(
  filtered_gtfs,
  trip_id = selected_trips,
  file = "shapes"
)

head(geometries)
```

```
Simple feature collection with 6 features and 2 fields
Geometry type: LINESTRING
Dimension:     XY
Bounding box:  xmin: -46.98404 ymin: -23.73644 xmax: -46.63535
               ymax: -23.19474
Geodetic CRS:  WGS 84
     trip_id origin_file                       geometry
1 CPTM L07-0      shapes LINESTRING (-46.63535 -23.5...
2 CPTM L07-1      shapes LINESTRING (-46.87255 -23.1...
3 CPTM L08-0      shapes LINESTRING (-46.64073 -23.5...
4 CPTM L08-1      shapes LINESTRING (-46.98404 -23.5...
5 CPTM L09-0      shapes LINESTRING (-46.77604 -23.5...
6 CPTM L09-1      shapes LINESTRING (-46.69711 -23.7...
```

Finally, we need to join the average headway data to the geometries and then configure the map as wished. In the example below, the color and line width of each trip geometry varies with its headway:

```
geoms_with_headways <- merge(
  geometries,
  average_headway,
  by = "trip_id"
)

ggplot(geoms_with_headways) +
  geom_sf(aes(color = average_headway, size = average_headway),
  alpha = 0.8) +
  scale_color_gradient(high = "#132B43", low = "#56B1F7") +
  labs(color = "Average headway", size = "Average headway") +
  theme_minimal()
```

FIGURE 9
Headways spatial distribution in SPTrans' GTFS



Source: Figure generated by the code snippet above.

As we can see, `{gtfstools}` makes the analysis of GTFS feeds a simple task that requires only basic knowledge of table manipulation packages (such as `{data.table}` or `{dplyr}`). The example shown in this section illustrates how one could use many of the package's functions together to reveal important aspects of public transport systems specified in the GTFS format.

SECTION 4

IMPACT ASSESSMENT OF TRANSPORTATION PROJECTS

The purpose of this section is to present a step-by-step example of how to use the methods introduced in this book to assess the impact of a transport infrastructure project on urban accessibility conditions using R.

Although accessibility analyses have been frequently used in scientific literature for more than two decades, only recently transport agencies and decision makers have begun to pay more attention to urban accessibility issues in their daily planning and operation of transport systems (Papa et al., 2015; Boisjoly and El-Geneidy, 2017). Much of this is due to the difficulty of incorporating accessibility analyses to project evaluation methodologies and to planning activities (Silva et al., 2017; Büttner, 2021).

In this section, we use a subway expansion project in Fortaleza (Brazil) as a case study to illustrate how to use the methods and R packages presented in the previous chapters to assess the accessibility impacts of transport projects. Chapter 6 presents a method to evaluate the effects of transportation investments not only on the average accessibility levels of the population, but also on the geographic and socioeconomic distribution of these impacts, which ultimately affect accessibility disparities. Applying the method involves using and manipulating different GTFS files, calculating travel time matrices, making decisions such as which accessibility measure to use, estimating accessibility levels, creating spatial visualizations of these estimates and calculating and analyzing inequality indicators. Therefore, this case study covers several topics discussed in the book and serves as a practical example of the concepts presented thus far.

It is important to mention that the evaluation of transportation projects, investments and policies should ideally encompass many different criteria. These criteria range from the degree of social participation in the policy development and decision-making process to their environmental, economic and social impacts. While assessing accessibility impacts is very important to identify the potential benefits of an intervention and to evaluate the performance of a transport network, it offers only a limited perspective of the effects of a given policy. Accessibility impact assessments should, therefore, complement and be accompanied by other analyses that investigate other potential impacts of transportation projects.

CHAPTER 6

6 COMPARING ACCESSIBILITY BETWEEN TWO TRANSPORT SCENARIOS

In this chapter, we will illustrate how to combine the material taught in previous chapters to assess the impact of a transport infrastructure project on urban accessibility conditions. To measure the impact of a transport project, we need to compare the accessibility levels both before and after the project implementation. We need, therefore:

- to use different sets of GTFS feeds and to edit them to represent the scenarios before and after the project implementation;
- to calculate two travel time matrices, one before and one after the project;
- to measure the accessibility levels both before and after the project; and
- to compare the accessibility conditions in both scenarios, looking at how the impacts are distributed both spatially and between socioeconomic groups.

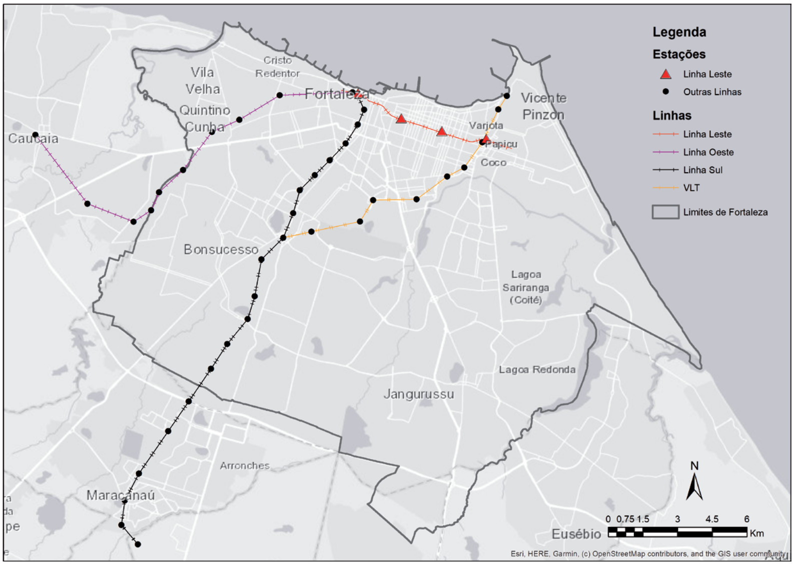
In this chapter, we will look at each one of these steps in detail. First, though, a brief presentation of our case study.

6.1 Case study

As a case study, we will assess Fortaleza's subway East line project (figure 10). The city of Fortaleza is the capital of Ceará state, located in Northeast Brazil. With an estimated population of 2.7 million inhabitants, Fortaleza is the fifth most populous city in the country.

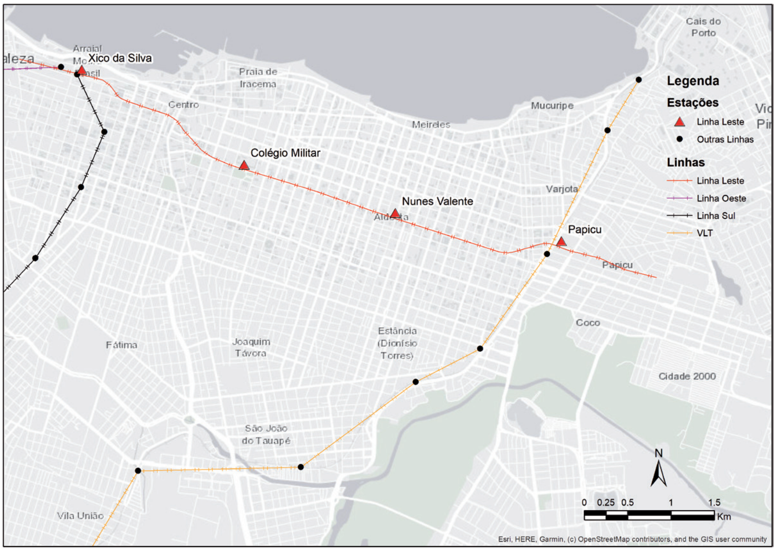
The East line is one of the biggest recent investments in Fortaleza's transport system. The corridor extends for 7.3 km and connects the city center to the Papicu neighborhood, connecting the South and West subway lines to the light rail (in portuguese, veículo leve sobre trilhos – VLT) corridor and Papicu's bus terminal (figure 11). The East line is still under construction as of the publication of this book, so we will be conducting an ex-ante analysis in this chapter – i.e. one in which we assess the future impacts of a project on urban accessibility conditions. This type of analysis differs from ex-post analyses, which are used to assess the impact of projects that have already been implemented.

FIGURE 10
Fortaleza's rapid transit network



Source: Braga et al. (2022).
Obs.: Figure whose layout and texts could not be formatted and proofread due to the technical characteristics of the original files (Publisher's note).

FIGURE 11
East line in detail



Source: Braga et al. (2022).
Obs.: Figure whose layout and texts could not be formatted and proofread due to the technical characteristics of the original files (Publisher's note).

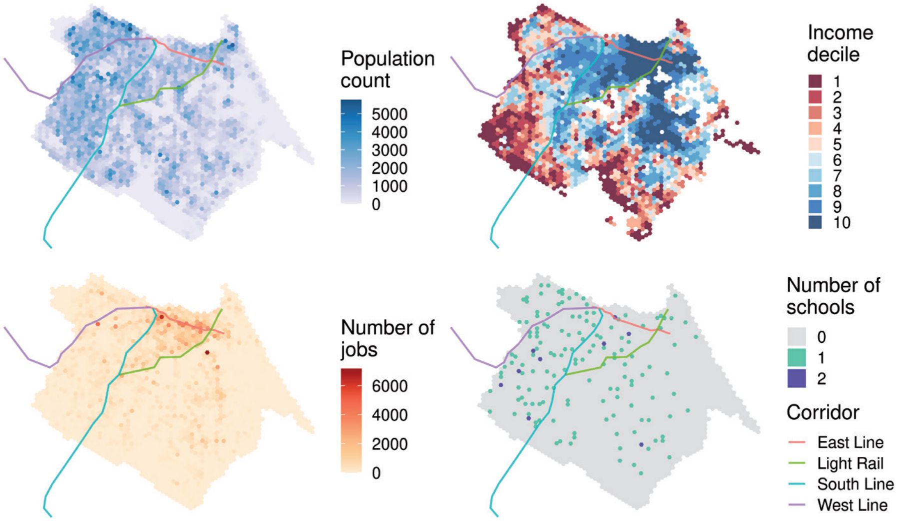
BOX 4
The scenario analyzed

It's important to note that East line's implementation will also be followed by changes to the frequency of subway's South and West lines and of Parangaba-Mucuripe light rail, in addition to cuts and adjustments on the municipal bus system, as detailed in Fortaleza's Sustainable Accessibility Plan (Pasfor).[1] For didactic purposes, however, the bus system changes were not incorporated into our analyses in this book. Therefore, the case study showcased in this chapter looks at simplified scenarios, in which only East line's implementation and the changes to other subway lines and VLT are considered.[2]

Authors' elaboration.
Notes: [1] Available at: https://www.pasfor.com.br/.
[2] For a more complete assessment that considers all the changes foreseen in Pasfor, please check Braga et al. (2022).

Figure 12 shows that Fortaleza's population is mainly distributed in the central and western parts of the city, although some relatively high density neighborhoods can also be seen in the southeastern region. Generally, wealthier groups (shown in blue in the income decile distribution map) tend to reside in the expanded city center, extending towards the southeast, while low-income groups (in red) are mainly located in the western and southern peripheries. Most of the formal jobs are distributed along key avenues, with higher concentrations in the city center. In contrast, public highschools are more equally distributed throughout the city.

FIGURE 12
Distribution of population, formal jobs, schools and rapid transit corridors in Fortaleza



Authors' elaboration.
Obs.: Figure whose layout and texts could not be formatted and proofread due to the technical characteristics of the original files (Publisher's note).

6.2 GTFS data used in the analysis

In this analysis, we will use the GTFS files made available by Etufor and Metrofor. These feeds describe the public transport network that operated in Fortaleza in October 2019. To access these data, we use the code below, in which we download the feeds using the {httr} package:

```
metrofor_path <- tempfile("metrofor", fileext = ".zip")
etufor_path <- tempfile("etufor", fileext = ".zip")

# downloads metrofor data
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/gtfs_for_metrofor_2021-01.zip",
  httr::write_disk(metrofor_path)
)

# downloads etufor data
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/gtfs_for_etufor_2019-10.zip",
  httr::write_disk(etufor_path)
)
```

To simulate the implementation of subway's East line, we also need a feed that describes its operation. This feed must contain some key information, such as the shape of the corridor, the stop locations, the travel time between stations and the frequency of trips. In this example, we will use a GTFS file previously created by the Access to Opportunities team for a more detailed assessment of the accessibility impacts caused by this project (Braga et al., 2022). Just like Etufor's and Metrofor's feeds, this feed has been published in the book GitHub repository and can be downloaded with the code below:

```
east_line_path <- tempfile("east_line", fileext = ".zip")

httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/gtfs_linha_leste.zip",
  httr::write_disk(east_line_path)
)
```

Etufor's and Metrofor's feeds, however, do not include the changes to the public transport system foreseen in Pasfor. Therefore, we have to edit the feeds using the {gtfstools} package to take these changes into consideration in the post-implementation scenario.

In our case study, we will consider the changes to the frequencies of the subway and light rail services listed in Braga et al. (2022), based on Pasfor: i) an increase in the South line subway frequency from four to ten trips per hour; ii) an increase in the West line subway frequency from two to five trips per hour; and iii) an increase in the Parangaba-Mucuripe light rail frequency from two to eight trips per hour. As we are only considering changes to the subway and light rail services, we only need to edit Metrofor's GTFS. First, we need to read this feed with `read_gtfs()` and understand how the trips are described. To do so, we are going to look at how the `routes`, `trips` and `calendar` tables are structured.

```
library(gtfstools)

metrofor_gtfs <- read_gtfs(metrofor_path)

metrofor_gtfs$routes[, .(route_id, route_long_name)]
```

```
   route_id       route_long_name
1:        8 VLT Parangaba Papicu
2:        6             Linha Sul
3:        7           Linha Oeste
```

```
metrofor_gtfs$trips[, .N, by = .(route_id, direction_id,
service_id)]
```

```
   route_id direction_id service_id  N
1:        7            0          4 15
2:        7            1          4 15
3:        6            0          4 63
4:        6            1          4 64
5:        8            0          4 29
6:        8            1          4 29
```

```
metrofor_gtfs$calendar
```

```
   service_id monday tuesday wednesday thursday friday saturday sunday
1:          4      1       1         1        1      1        1      0
   start_date   end_date
1: 2020-01-01 2021-12-31
```

The feed describes three distinct routes: the two subway corridors and the light rail corridor. Since the feed does not include a `frequencies` table, each route is described by many trips that depart at different times of the day. There is information for trips in both directions, and they are all associated with the same service that operates on business days and saturdays.

The strategy we are going to adopt to make the necessary changes to the feed include three steps, as follows.

1) First, we are going to filter the Metrofor feed to keep only one trip per direction for each route. This trip will tell us the travel time each trip takes between its stops.

2) Then, we are going to add a `frequencies` table to the GTFS object, in which we are going to describe the frequency of each trip.

3) Finally, we are going to "convert" the recently-added `frequencies` entries to timetables described in `stop_times`. This conversion will be used to maintain the original feed's characteristic of describing trips using only the `stop_times` table.

To keep only one trip per direction for each route, we need to filter the feed using `filter_by_trip_id()`. To do so, we are going to identify the first trip entry per route and per direction and use the function to keep only these trips in the feed.

```r
# identifies the table index in which the first entries per
# route and per direction are located at
index <- metrofor_gtfs$trips[, .I[1], by = .(route_id,
direction_id)]$V1

# selects the id of each row
selected_trips <- metrofor_gtfs$trips[index]$trip_id

# filters the gtfs to keep only the trips above
filtered_gtfs <- filter_by_trip_id(metrofor_gtfs,
trip_id = selected_trips)

filtered_gtfs$trips
```

```
   trip_id     trip_headsign direction_id block_id shape_id service_id route_id
1:       4           Caucaia            0                            4        7
2:      19      Moura Brasil            1                            4        7
3:      34 Carlito Benevides            0                            4        6
4:      96    Chico da Silva            1                            4        6
5:     159              Iate            0                            4        8
6:     181         Parangaba            1                            4        8
```

To facilitate the data manipulation, we are going to change the trip ids, identifying the corridor and the direction in which they operate. We need to make this change both in the trips and in the `stop_times` tables.

```
filtered_gtfs$stop_times[
  ,
  trip_id := data.table::fcase(
    trip_id == "4", "west_subway_0",
    trip_id == "19", "west_subway_1",
    trip_id == "34", "south_subway_0",
    trip_id == "96", "south_subway_1",
    trip_id == "159", "light_rail_0",
    trip_id == "181", "light_rail_1"
  )
]

filtered_gtfs$trips[
  ,
  trip_id := data.table::fcase(
    trip_id == "4", "west_subway_0",
    trip_id == "19", "west_subway_1",
    trip_id == "34", "south_subway_0",
    trip_id == "96", "south_subway_1",
    trip_id == "159", "light_rail_0",
    trip_id == "181", "light_rail_1"
  )
]

filtered_gtfs$trips
```

```
          trip_id     trip_headsign direction_id block_id shape_id service_id
1:  west_subway_0           Caucaia            0                            4
2:  west_subway_1      Moura Brasil            1                            4
3: south_subway_0 Carlito Benevides            0                            4
4: south_subway_1    Chico da Silva            1                            4
5:   light_rail_0              Iate            0                            4
6:   light_rail_1         Parangaba            1                            4

   route_id
1:        7
2:        7
3:        6
4:        6
5:        8
6:        8
```

Now we need to add a frequencies table describing the frequency of each trip. Note, however, that the GTFS specification requires us to list the *headway* of each trip, and not its frequency. The headway is the inverse of the *frequency*, so we need to divide the interval of one hour (3,600 seconds) by the frequency of each route (10 trips/hour for the South line, 5 trips/hours for the West line and 8 trips/hours for the light rail). As a result, we have that the headway of the South line, West line and the light rail will be, respectively, 360, 720 and 450 seconds. With the code below, we create a frequencies table using the {tibble} and {data.table} packages.

```
frequencies <- tibble::tribble(
  ~trip_id,         ~start_time, ~end_time,  ~headway_secs, ~exact_times,
  "west_subway_0",  "06:00:00",  "09:00:00", 720L,           1,
  "west_subway_1",  "06:00:00",  "09:00:00", 720L,           1,
  "south_subway_0", "06:00:00",  "09:00:00", 360L,           1,
  "south_subway_1", "06:00:00",  "09:00:00", 360L,           1,
  "light_rail_0",   "06:00:00",  "09:00:00", 450L,           1,
  "light_rail_1",   "06:00:00",  "09:00:00", 450L,           1
)

# converts the table to data.table
data.table::setDT(frequencies)

# assigns table to gtfs object
filtered_gtfs$frequencies <- frequencies
```

To keep things simple in this case study, we assume that these headways are valid between 6 am and 9 am. This assumption works in our case because we are only going to calculate the travel time matrix during the morning peak. If we wanted to calculate travel times in other periods of the day or to use this GTFS to examine operation of these corridors throughout the day, however, we would have to list the headways for the rest of the day as well. The value 1 in the exact_times column determines that the trips' timetables during the specified period must follow the headway exactly, not approximately.[19]

The GTFS object that results from the modifications done up until this stage can already be used to calculate travel time matrices. However, in order to restore the original feed's characteristic of not having a frequencies table, we "convert" this table's entries into timetables described in stop_times. To do so, we use the frequencies_to_stop_times() function. Since all trips in the feed are converted, the frequencies table is removed from the GTFS object.

[19] For more details, please refer to the frequencies table description in chapter 4.

```
filtered_gtfs <- frequencies_to_stop_times(filtered_gtfs)

filtered_gtfs$frequencies
```

```
NULL
```

To check if the data manipulation worked as intended, we look at the West line trips that head towards Caucaia (whose `direction_id` is `0`). With a frequency of 5 trips/hour between 6 am and 9 am, the `trips` table must contain exactly 16 entries related to this route (5 trips/hour during 3 hours plus a trip starting at 9 am).

```
west_line_subway <- filtered_gtfs$trips[grepl("west_subway_0",
trip_id)]

nrow(west_line_subway)
```

```
[1] 16
```

```
west_line_subway$trip_id
```

```
 [1] "west_subway_0_1"  "west_subway_0_2"  "west_subway_0_3"  "west_subway_0_4"
 [5] "west_subway_0_5"  "west_subway_0_6"  "west_subway_0_7"  "west_subway_0_8"
 [9] "west_subway_0_9"  "west_subway_0_10" "west_subway_0_11" "west_subway_0_12"
[13] "west_subway_0_13" "west_subway_0_14" "west_subway_0_15" "west_subway_0_16"
```

The `stop_times` table, in turn, must list these trips departing every 12 minutes (equivalent to a 450-second headway). Thus, we need to check the first entry of the timetable of each one of the trips listed above.

```
west_subway_trips <- west_line_subway$trip_id

# identifies above trips' first entries in stop_times
trip_indices <- filtered_gtfs$stop_times[
  trip_id %in% west_subway_trips,
  .I[1],
  by = trip_id
]$V1

filtered_gtfs$stop_times[trip_indices, .(trip_id, departure_time)]
```

```
            trip_id  departure_time
 1:  west_subway_0_1         06:00:00
 2:  west_subway_0_2         06:12:00
 3:  west_subway_0_3         06:24:00
 4:  west_subway_0_4         06:36:00
 5:  west_subway_0_5         06:48:00
 6:  west_subway_0_6         07:00:00
 7:  west_subway_0_7         07:12:00
 8:  west_subway_0_8         07:24:00
 9:  west_subway_0_9         07:36:00
10: west_subway_0_10         07:48:00
11: west_subway_0_11         08:00:00
12: west_subway_0_12         08:12:00
13: west_subway_0_13         08:24:00
14: west_subway_0_14         08:36:00
15: west_subway_0_15         08:48:00
16: west_subway_0_16         09:00:00
```

We can see that the "conversion" from `frequencies` to `stop_times` worked correctly, allowing us to use this modified feed to calculate the travel time matrix in the post-implementation scenario. To do this, we need to save this GTFS object to disk in `.zip` format, just like the rest of the feeds we are going to use. We use the `write_gtfs()` function for that.

```
modified_metrofor_path <- tempfile("modified_metrofor",
fileext = ".zip")

write_gtfs(filtered_gtfs, modified_metrofor_path)
```

Now, we have four distinct GTFS files:

- Etufor's feed, describing the bus system that operated in October 2019;
- Metrofor's feed, describing the subway's (South and West lines) and the light rail's operation in October 2019;
- Metrofor's modified feed, describing the South and West subway lines' and the light rail's future operation, as foreseen in Pasfor; and
- East line's feed, describing the future operation of the subway East line.

These four GTFS files will be used to calculate the accessibility conditions in Fortaleza before and after the implementation of the East line. In the pre-implementation scenario, we are going to calculate the travel time matrices using only the October 2019 feeds from Metrofor and Etufor. In the post-implementation scenario, we are going to use Etufor's feed, Metrofor's modified feed with updated frequencies and the feed of the new East line.

6.3 Calculating the travel time matrices

After making the necessary changes to the GTFS files and defining which feeds we are going to use in each scenario, we need to calculate the travel time matrices that we are going to use to estimate the accessibility levels. To do this, we are going to use the `travel_time_matrix()` function from `{r5r}`, previously presented in chapter 3.

Before calculating the travel matrices, however, we need to organize our data as required by `{r5r}`. With the code below, we create a separate directory for each scenario (before and after implementation) in which we save the files used in the routing process:

```
# creates root analysis directory
analysis_dir <- "impact_analysis"
dir.create(analysis_dir)

# creates scenarios directories
before_dir <- file.path(analysis_dir, "before")
after_dir <- file.path(analysis_dir, "after")

dir.create(before_dir)
dir.create(after_dir)

# copy relevant files to "before" scenario directory
file.copy(from = etufor_path, to = file.path(before_dir,
"etufor.zip"))
file.copy(from = metrofor_path, to = file.path(before_dir,
"metrofor.zip"))

# copy relevant files to "after" scenario directory
file.copy(from = etufor_path, to = file.path(after_dir,
"etufor.zip"))
file.copy(
  from = modified_metrofor_path,
  to = file.path(after_dir, "modified_metrofor.zip")
)
file.copy(
  from = east_line_path,
  to = file.path(after_dir, "east_line.zip")
)

# visualizes file structure
fs::dir_tree(analysis_dir)
```

```
impact_analysis
├── after
│   ├── east_line.zip
│   ├── etufor.zip
│   ├── linha_leste.zip
│   ├── metrofor.zip
│   ├── modified_metrofor.zip
│   └── network_settings.json
└── before
    ├── etufor.zip
    ├── metrofor.zip
    └── network_settings.json
```

To estimate the travel times in our study area, we also need a file representing the local street network extracted from OSM in `.pbf` format. Optionally, we are also going to use a file representing the local topography, in `.tif` format. These data sets, just like the GTFS files, can be downloaded from the book repository. Assuming that the implementation of East line will not affect the street network, the pedestrian infrastructure and the topography in the region, we can use the same files to calculate both travel time matrices. With the code below, we download these data sets and copy the files to both scenarios' directories.

```r
# creates temporary files to save data
pbf_path <- tempfile("street_network", fileext = ".osm.pbf")
tif_path <- tempfile("topography", fileext = ".tif")

# downloads OSM data
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/fortaleza.osm.pbf",
   httr::write_disk(pbf_path)
)

# downloads topography data
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/topografia3_for.tif",
  httr::write_disk(tif_path)
)

# copies files to both scenarios’ directories
file.copy(from = pbf_path, to = file.path(before_dir,
"street_network.osm.pbf"))
```

```
file.copy(from = pbf_path, to = file.path(after_dir,
"street_network.osm.pbf"))

file.copy(from = tif_path, to = file.path(before_dir,
"topography.tif"))
file.copy(from = tif_path, to = file.path(after_dir,
"topography.tif"))
```

```
fs::dir_tree(analysis_dir)
```

```
impact_analysis
├── after
│   ├── east_line.zip
│   ├── etufor.zip
│   ├── linha_leste.zip
│   ├── metrofor.zip
│   ├── modified_metrofor.zip
│   ├── network_settings.json
│   ├── street_network.osm.pbf
│   └── topography.tif
└── before
    ├── etufor.zip
    ├── metrofor.zip
    ├── network_settings.json
    ├── street_network.osm.pbf
    └── topography.tif
```

With the data properly organized, we can now start calculating the travel time matrices. The first step is to use the street network, public transport and topography data to build the transport network used by {r5r} in the routing process. To do this, we use the setup_r5() function, which also returns a connection to R5. With the code below, we build two networks, one for each scenario:

```
# allocates memory to be used by Java Virtual Machine
options(java.parameters = "-Xmx4G")

library(r5r)

r5r_core_before <- setup_r5(before_dir, verbose = FALSE)
r5r_core_after <- setup_r5(after_dir, verbose = FALSE)
```

Having built the transport networks, we can now proceed to the actual travel time matrices calculation. In this step, we are going to use the centroids of a hexagonal grid covering Fortaleza as our origins and destinations. We are going to use the hexagonal grid made available by `{aopdata}`.[20] Each grid hexagon covers an area of 0.11 km², similar to a city block, which produces results at a fine spatial resolution.

For a proper comparison between both scenarios, we need to calculate the two travel matrices using the same parameters. We consider trips by foot or by public transport, allow walking trips of at most 30 minutes to access or egress from public transport stops and limit the maximum trip duration to 60 minutes. We also consider a departure time of 7 am, during the morning peak of a typical monday:

```r
# downloads spatial grid data
fortaleza_grid <- aopdata::read_grid("Fortaleza")

# gets cells' centroids
points <- sf::st_centroid(fortaleza_grid)

# renames the column holding the cell ids
names(points)[1] <- "id"

# calculates the "before" scenario travel time matrix
ttm_before <- travel_time_matrix(
  r5r_core_before,
  origins = points,
  destinations = points,
  mode = c("WALK", "TRANSIT"),
  departure_datetime = as.POSIXct(
    "02-03-2020 07:00:00",
    format = "%d-%m-%Y %H:%M:%S"
  ),
  max_walk_time = 30,
  max_trip_duration = 60,
  verbose = FALSE,
  progress = FALSE
)

# calculates the "after" scenario travel time matrix
ttm_after <- travel_time_matrix(
  r5r_core_after,
  origins = points,
  destinations = points,
  mode = c("WALK", "TRANSIT"),
  departure_datetime = as.POSIXct(
```

[20] For more details on the package, please refer to section 5.

```
    "02-03-2020 07:00:00",
    format = "%d-%m-%Y %H:%M:%S"
  ),
  max_walk_time = 30,
  max_trip_duration = 60,
  verbose = FALSE,
  progress = FALSE
)

head(ttm_before)
```

```
           from_id           to_id travel_time_p50
1: 89801040323ffff 89801040323ffff               2
2: 89801040323ffff 89801040327ffff              22
3: 89801040323ffff 8980104032bffff              32
4: 89801040323ffff 8980104032fffff              15
5: 89801040323ffff 89801040333ffff              10
6: 89801040323ffff 89801040337ffff              19
```

```
head(ttm_after)
```

```
           from_id           to_id travel_time_p50
1: 89801040323ffff 89801040323ffff               2
2: 89801040323ffff 89801040327ffff              22
3: 89801040323ffff 8980104032bffff              32
4: 89801040323ffff 8980104032fffff              15
5: 89801040323ffff 89801040333ffff              10
6: 89801040323ffff 89801040337ffff              19
```

At first sight, our matrices look exactly the same: all travel times shown in the samples above are identical. This happens because the subway expansion project is limited to a relatively small area near Fortaleza's city center, and the changes to the frequencies of the other subway and light rail corridors mainly affect these corridors' immediate surroundings. Thus, many trips that take place in the city are not affected by these transport interventions. However, the travel time between many origin-destination pairs are, in fact, impacted:

```
# joins both scenarios' travel times in the same data set
comparison <- merge(
  ttm_before,
  ttm_after,
  by = c("from_id", "to_id"),
  suffixes = c("_before", "_after")
)
```

```r
# shows the OD pairs whose travel times got faster
comparison[travel_time_p50_before < travel_time_p50_after]
```

```
               from_id           to_id travel_time_p50_before
    1: 8980104096fffff 8980104e803ffff                     48
    2: 8980104096fffff 8980104e807ffff                     57
    3: 8980104096fffff 8980104e80bffff                     53
    4: 8980104096fffff 8980104e80fffff                     55
    5: 8980104096fffff 8980104e863ffff                     56
   ---
12889: 8980104eecbffff 8980104ea5bffff                     52
12890: 8980104eecbffff 8980104eac3ffff                     49
12891: 8980104eecbffff 8980104ead3ffff                     44
12892: 8980104eecbffff 8980104eadbffff                     49
12893: 8980104eecbffff 8980104ee6bffff                     41
       travel_time_p50_after
    1:                    50
    2:                    59
    3:                    55
    4:                    57
    5:                    57
   ---
12889:                    57
12890:                    53
12891:                    47
12892:                    50
12893:                    42
```

6.4 Calculating accessibility levels in both scenarios

Calculating the accessibility levels in both scenarios is really simple, requiring only some basic data processing before we apply one of the functions from the {accessibility} package. To facilitate the data manipulation, we merge the travel time matrices of both scenarios into a single table and identify each scenario with a column named scenario:

```r
ttm <- rbind(ttm_before, ttm_after, idcol = "scenario")
ttm[, scenario := factor(scenario, labels = c("before", "after"))]

ttm
```

```
         scenario         from_id           to_id travel_time_p50
      1:   before 89801040323ffff 89801040323ffff               2
      2:   before 89801040323ffff 89801040327ffff              22
      3:   before 89801040323ffff 8980104032bffff              32
      4:   before 89801040323ffff 8980104032fffff         15
      5:   before 89801040323ffff 89801040333ffff         10
    ---
3775198:    after 8980107b6dbffff 8980107b6cbffff          8
3775199:    after 8980107b6dbffff 8980107b6cfffff         15
3775200:    after 8980107b6dbffff 8980107b6d3ffff          9
3775201:    after 8980107b6dbffff 8980107b6d7ffff         16
3775202:    after 8980107b6dbffff 8980107b6dbffff          0
```

To calculate the accessibility levels, we need a table with some land used data for Fortaleza. We can download such data using the `read_landuse()` function from the `{aopdata}` package, which returns a table containing the population and opportunities count in each one of the hexagons that compose the previously downloaded spatial grid.

```r
fortaleza_data <- aopdata::read_landuse(
  "Fortaleza",
  showProgress = FALSE
)
```

For demonstration purposes, we calculate the accessibility to jobs and public highschools in our study area. The information on the total number of jobs and public highschools in each hexagon is listed in the columns `T001` and `E004`, respectively. We rename them to facilitate their identification. We also keep in the land use dataset only the columns that we are going to use later, including the columns `P001`, which lists the total population in each hexagon, and `R003`, which contains the income decile:

```r
cols_to_keep <- c("id", "jobs", "schools", "population",
"decile")
data.table::setnames(
  fortaleza_data,
  old = c("id_hex", "T001", "E004", "P001", "R003"),
  new = cols_to_keep
)

# deletes the columns that won't be used
fortaleza_data[, setdiff(names(fortaleza_data), cols_to_keep)
:= NULL]

fortaleza_data
```

```
                     id population  decile  jobs  schools
   1: 89801040323ffff         30       1     0        0
   2: 89801040327ffff        318       1     7        0
   3: 8980104032bffff          0      NA     0        0
   4: 8980104032fffff        103       1    98        0
   5: 89801040333ffff         43       1     0        0
  ---
2558: 8980107b6cbffff       2575       4   124        0
2559: 8980107b6cfffff       2997       3     4        0
2560: 8980107b6d3ffff       1751       8    14        0
2561: 8980107b6d7ffff       2032       4   134        0
2562: 8980107b6dbffff       1896       9   193        0
```

A key decision in any accessibility analysis is which accessibility measure to use. It's extremely important to weigh the pros and cons of each measure and to comprehend which metrics are more adequate for the type of opportunities we are looking at. In this example, we use two distinct measures.

1) To calculate accessibility to jobs, we use a cumulative opportunities measure. This metric allows us to understand how many jobs are accessible within a given time frame. Despite its limitations discussed in chapter 2, this is one of the most commonly used accessibility metrics. This is to a large extent because the results from this accessibility indicator are extremely easy to communicate and interpret. In this example, we set a travel time threshold of 60 minutes, which is close to average commuting time by public transport in Fortaleza (approximately 58 minutes, according to Pasfor).

2) To calculate accessibility to public highschools, we use a minimum travel cost measure. This metric is particularly useful to assess the coverage of essential public services, such as basic health and education facilities. We can use this measure, for example, to identify population groups that are further from these opportunities than a time/distance limit deemed reasonable.

As previously shown in chapter 3, we can calculate this measures using the `cumulative_cutoff()` and `cost_to_closest()` functions, respectively, from the `{accessibility}` package:

```r
library(accessibility)

access_to_jobs <- cumulative_cutoff(
  ttm,
  land_use_data = fortaleza_data,
  opportunity = "jobs",
  travel_cost = "travel_time_p50",
  cutoff = 60,
  group_by = "scenario"
)

access_to_jobs
```

```
                    id scenario   jobs
   1: 89801040323ffff   before  48049
   2: 89801040327ffff   before  26044
   3: 8980104032bffff   before  25862
   4: 8980104032fffff   before  69361
   5: 89801040333ffff   before  48049
  ---
5120: 8980107b6cbffff    after 378840
5121: 8980107b6cfffff    after 286878
5122: 8980107b6d3ffff    after 339878
5123: 8980107b6d7ffff    after 359648
5124: 8980107b6dbffff    after 372565
```

```
time_to_schools <- cost_to_closest(
  ttm,
  land_use_data = fortaleza_data,
  opportunity = "schools",
  travel_cost = "travel_time_p50",
  group_by = "scenario"
)

time_to_schools
```

```
                  id scenario travel_time_p50
   1: 89801040323ffff   before              36
   2: 89801040323ffff    after              36
   3: 89801040327ffff   before              41
   4: 89801040327ffff    after              41
   5: 8980104032bffff   before              41
  ---
5120: 8980107b6d3ffff    after              19
5121: 8980107b6d7ffff   before              14
5122: 8980107b6d7ffff    after              14
5123: 8980107b6dbffff   before              15
5124: 8980107b6dbffff    after              15
```

We can see that the minimum travel cost function output includes some `Inf` values, which are used to signal origins that cannot reach any opportunities given the trips that compose the travel time matrix. In our case, origins listed with this value cannot reach any public highschools within 60 minutes of travel (which is the travel time limit imposed when calculating the matrix). To simplify the process from this point onward, we consider that these regions are 80 minutes away from their nearest school:

```r
# substitutes Inf values by 80 minutes
time_to_schools[
  ,
  travel_time_p50 := ifelse(is.infinite(travel_time_p50),
  80, travel_time_p50)
]
```

Having done that, we can calculate the accessibility difference between the two scenarios with the code below. This information is useful to clearly communicate how the accessibility conditions in the city would be impacted by the future implementation of the East subway line and the frequency changes foreseen in Pasfor.

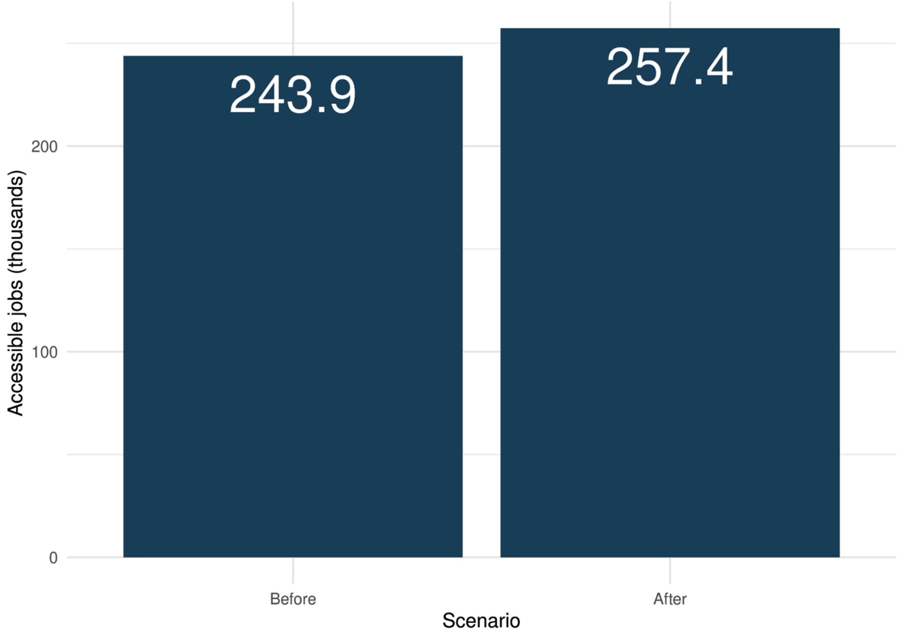
```r
access_to_jobs[
  ,
  difference := data.table::shift(jobs, type = "lead") - jobs,
  by = id
]

time_to_schools[
  ,
  difference := data.table::shift(travel_time_p50, type = "lead") -
      travel_time_p50,
  by = id
]
```

6.5 Analyzing accessibility levels

Now that we have calculated the accessibility levels in both scenarios and the difference between them, we can examine how the future implementation of the East line coupled with the changes to the frequencies of the subway and light rail services will impact the accessibility conditions in our study area. As a first exploratory analysis, we can investigate how these changes affect the average accessibility in the city. Looking at the accessibility to jobs first, we calculate the average number of accessible jobs in each scenario. Here, it’s important to weigh the accessibility levels by the population of each grid cell, as hexagons with larger populations contribute more to the city's average than hexagons with fewer residents.

```r
library(ggplot2)
library(patchwork)

# merges accessibility table with land use data (population
# count and income decile)
access_to_jobs <- merge(
  access_to_jobs,
  fortaleza_data,
  by = "id"
)

# renames columns with duplicated names
data.table::setnames(
  access_to_jobs,
  old = c("jobs.x", "jobs.y"),
  new = c("access_to_jobs", "job_count")
)

# calculates avg accessibility in each scenario
avg_access <- access_to_jobs[
  ,
  .(access  = weighted.mean(access_to_jobs,
  w = as.numeric(population))),
  by = scenario
]

ggplot(data = avg_access, aes(x = scenario, y = access / 1000)) +
  geom_col(fill = "#0f3c53") +
  geom_text(
    aes(label = round(access / 1000, digits = 1)),
    vjust = 1.5,
    color = "white",
    size = 10
  ) +
  ylab("Accessible jobs (thousands)") +
  scale_x_discrete(name = "Scenario", labels = c("Before", "After")) +
  theme_minimal()
```

FIGURE 13
Average accessibility to jobs in Fortaleza by transport scenario



Source: Figure generated by the code snippet above.

The results show that Fortaleza's population could reach on average 243,859 jobs by public transport in up to 60 minutes before the subway expansion, in 2019. The East line's implementation and the changes to the frequencies of subway and light rail services will result in an increase of 5.5%, to 257,369 jobs on average.

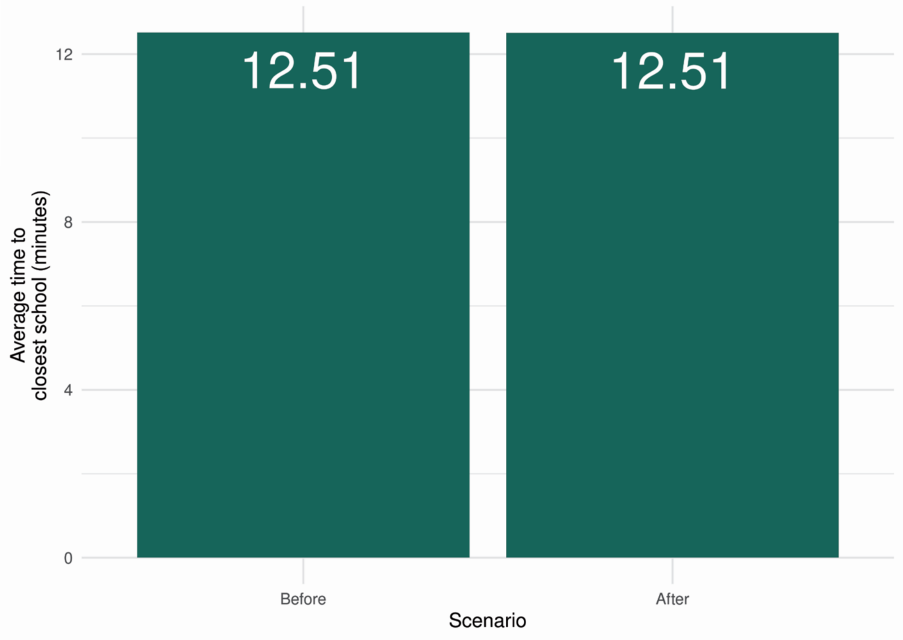
When we look at the average time to reach the closest public highschool, we see that the changes to the transport system barely affect the accessibility to these schools. On average, Fortaleza's population would take approximately 13 minutes to reach the nearest public highschool to their home in 2019. After the subway extension and the increase to the subway and light rail frequencies, this value will remain virtually unchanged.

```
# merges time to schools table with land use data
time_to_schools <- merge(
  time_to_schools,
  fortaleza_data,
  by = "id"
)
```

```
# calculates avg time to schools in each scenario
avg_time <- time_to_schools[
  ,
  .(time  = weighted.mean(travel_time_p50,
  w = as.numeric(population))),
  by = scenario
]

ggplot(data = avg_time, aes(x = scenario, y = time)) +
  geom_col(fill = "#0d6556") +
  geom_text(
    aes(label = round(time, digits = 2)),
    vjust = 1.5,
    color = "white",
    size = 10
  ) +
  ylab("Average time to\nclosest school (minutes)") +
  scale_x_discrete(name = "Scenario", labels = c("Before", "After")) +
  theme_minimal()
```

FIGURE 14

Average time to the closest public highschool in Fortaleza by transport scenario



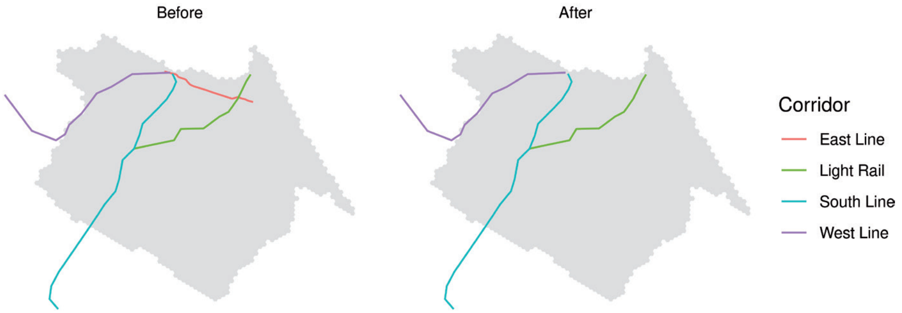
Source: Figure generated by the code snippet above.

In summary, the results show that the planned construction of the East line and the frequency adjustment of the other rail services in Fortaleza will affect accessibility to jobs much more significantly than the accessibility to public highschools. This is mainly a result of how these two types of opportunities are spatially distributed in Fortaleza: while jobs are much more concentrated in the city center, schools are better distributed throughout the city. The changes to the public transport system, therefore, could help the residents of regions far from the city center reach the jobs located there. On the other hand, public highschools are much more evenly distributed across the city, which results in relatively good accessibility conditions even before the changes to the public transport network. This helps us explain why the transport interventions will have such a low impact on the travel time necessary to reach the nearest schools.

These results can be more deeply understood when we observe their spatial distribution. Before doing so, however, we create a spatial object outlining the shapes of the public transport corridors in the city, which will help making the impact of the changes to the transport network even clearer.

```r
# reads the gtfs files required to create the geometries of
# each corridor
metrofor_gtfs <- read_gtfs(metrofor_path)
east_line_gtfs <- read_gtfs(east_line_path)

# metrofor's gtfs does not contain a shapes table, so we have
# to create the geometries from the stops and stop_times
# tables with
corridors_trips <- c("4", "34", "159")

# the stop sequence from one of the trips is not properly
# order, so we have to manually order them
metrofor_gtfs$stop_times <- metrofor_gtfs$stop_times[
  order(trip_id, stop_sequence)
]
metrofor_shape <- gtfstools::get_trip_geometry(
  metrofor_gtfs,
  trip_id = corridors_trips
)

# converts the east line shape in one of the directions to
# spatial geometry
east_line_shape <- gtfstools::convert_shapes_to_sf(
  east_line_gtfs,
  shape_id = "LL_0"
)

# names each route and bind the two tables together
east_line_shape$corridor <- "East Line"
metrofor_shape$corridor <- data.table::fcase(
  metrofor_shape$trip_id == 4, "West Line",
  metrofor_shape$trip_id == 34, "South Line",
  metrofor_shape$trip_id == 159, "Light Rail"
)

metrofor_shape$origin_file <- NULL
metrofor_shape$trip_id <- NULL
east_line_shape$shape_id <- NULL

corridors_shapes <- rbind(metrofor_shape, east_line_shape)
# duplicates the table, adds a column identifying each
# scenario and removes east line from the pre-implementation
# scenario
corridors_shapes <- rbind(corridors_shapes, corridors_shapes)
corridors_shapes$scenario <- rep(c("before", "after"), each = 4)
corridors_shapes <- subset(
  corridors_shapes,
  corridor != "East Line" | scenario != "before"
)
corridors_shapes$scenario <- factor(
  corridors_shapes$scenario,
  labels = c("before", "after")
)

ggplot() +
  geom_sf(data = fortaleza_grid, fill = "gray90", color = NA) +
  geom_sf(data = corridors_shapes, aes(color = corridor)) +
  scale_color_manual(
    name = "Corridor",
    values = c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF")
  ) +
  facet_wrap(
    ~ scenario,
    nrow = 1,
    labeller = as_labeller(c(before = "Before", after = "After"))
  ) +
  theme_void()
```

FIGURE 15
Spatial distribution of rapid transit corridors in Fortaleza by transport scenario

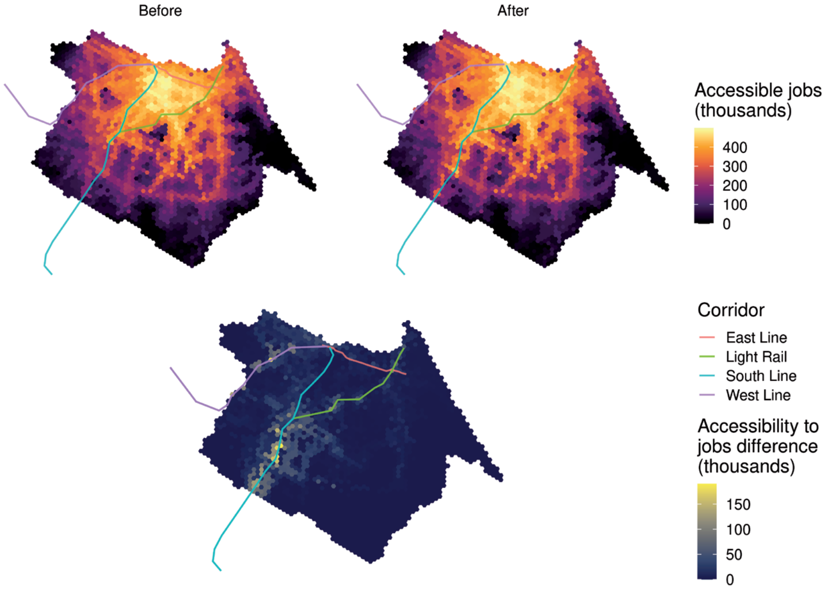


Source: Figure generated by the code snippet above.

Now we can analyze the spatial distribution of accessibility levels in both scenarios, as well as the accessibility difference between them. To do this, we need to merge the accessibility estimates with the spatial grid of our study area. We first look at access to jobs:

```r
# merges accessibility data with fortaleza's spatial grid and
# convert the result into a spatial object
access_to_jobs <- merge(
  access_to_jobs,
  fortaleza_grid,
  by.x = "id",
  by.y = "id_hex"
)
access_to_jobs_sf <- sf::st_sf(access_to_jobs)

# configures access distribution maps in both scenarios
access_dist <- ggplot() +
  geom_sf(
    data = access_to_jobs_sf,
    aes(fill = access_to_jobs),
    color = NA
  ) +
  facet_wrap(
    ~ scenario,
    nrow = 1,
    labeller = as_labeller(c(before = "Before", after = "After"))
  ) +
  scale_fill_viridis_c(
    option = "inferno",
    label = scales::label_number(scale = 1 / 1000)
  ) +
  labs(fill = "Accessible jobs\n(thousands)", color = "Corridor") +
  geom_sf(
    data = corridors_shapes,
    aes(color = corridor),
    alpha = 0.8,
    show.legend = FALSE
  ) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
  "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# configures difference map
difference_dist <- ggplot() +
  geom_sf(
    data = subset(access_to_jobs_sf, !is.na(difference)),
    aes(fill = difference),
    color = NA
  ) +
  scale_fill_viridis_c(
    option = "cividis",
    label = scales::label_number(scale = 1 / 1000)
  ) +
  labs(
    fill = "Accessibility to\njobs difference\n(thousands)",
    color = "Corridor"
  ) +
  geom_sf(data = corridors_shapes, aes(color = corridor),
  alpha = 0.8) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
  "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# combines both plots
access_dist / difference_dist + plot_layout(ncol = 1,
heights = c(1, 1))
```

FIGURE 16
Spatial distribution of accessibility to jobs by transport scenario and of the difference between scenarios



Source: Figure generated by the code snippet above.

The map shows that the regions that will benefit the most from the changes to the transport system are those distant from the city center, but which are still close to rapid transit stations. The job accessibility gains concentrate mainly around the South and West subway corridors, and, to a smaller extent, around some of the light rail stations. Even regions close to these corridors, although not immediately adjacent to them, display large accessibility gains, highlighting the importance of the transport network connectivity to guarantee good accessibility conditions. The region around the new East line, on the other hand, which already concentrated some of the highest accessibility levels in the city even before the implementation of the new corridor, shows only modest accessibility gains.

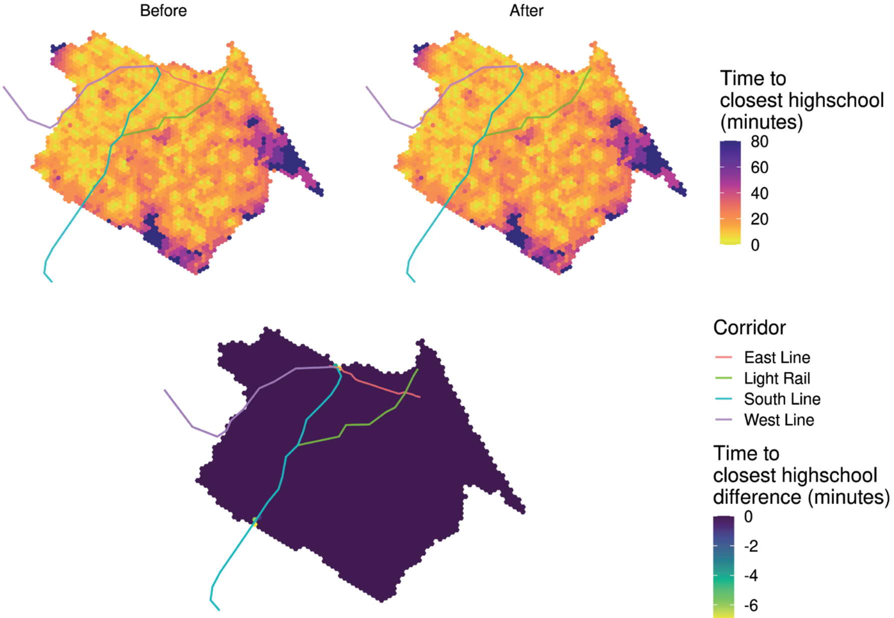
The maps of travel time to the nearest school, however, depict a different story.

```r
# merges time to schools data with Fortaleza's spatial grid
# and converts the result into a spatial object
time_to_schools <- merge(
  time_to_schools,
  fortaleza_grid,
  by.x = "id",
  by.y = "id_hex"
)
```

```
time_to_schools_sf <- sf::st_sf(time_to_schools)

# configures time to schools distribution maps in both scenarios
time_dist <- ggplot() +
  geom_sf(data = time_to_schools_sf, aes(fill = travel_time_p50),
  color = NA) +
  facet_wrap(
    ~ scenario,
    nrow = 1,
    labeller = as_labeller(c(before = "Before", after = "After"))
  ) +
  scale_fill_viridis_c(option = "plasma", direction = -1) +
  labs(fill = "Time to\nclosest highschool\n(minutes)",
  color = "Corridor") +
  geom_sf(
    data = corridors_shapes,
    aes(color = corridor),
    alpha = 0.8,
    show.legend = FALSE
  ) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
  "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# configures difference map
time_diff_dist <- ggplot() +
  geom_sf(
    data = subset(time_to_schools_sf, !is.na(difference)),
    aes(fill = difference),
    color = NA
  ) +
  scale_fill_viridis_c(option = "viridis", direction = -1) +
  labs(
    fill = "Time to\nclosest highschool\ndifference (minutes)",
    color = "Corridor"
  ) +
  geom_sf(data = corridors_shapes, aes(color = corridor), alpha =
  0.8) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
  "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# combines both plots
time_dist / time_diff_dist + plot_layout(ncol = 1, heights = c(1, 1))
```

FIGURE 17

Spatial distribution of travel time to the closest public highschool by transport scenario and of the difference between scenarios


Before
After
Time to closest highschool (minutes)
80
60
40
20
0
Corridor
East Line
Light Rail
South Line
West Line
Time to closest highschool difference (minutes)
0
-2
-4
-6

Source: Figure generated by the code snippet above.

The East line and the changes to the subway and light rail frequencies barely affect the accessibility to public highschools in Fortaleza. Very few hexagons present any accessibility gain between scenarios, with the exception of a small number of grid cells very close to subway stations. As we can see, the accessibility to schools is much more evenly distributed than the accessibility to jobs. Again, this is a consequence of how public highschools are distributed in the city: unlike the jobs distribution, which tends to follow economic criteria, the spatial planning of public schools in Brazil is guided by equity guidelines, aiming to increase the proximity between schools and vulnerable population groups. Nonetheless, the degree to which education policies successfully promote equitable accessibility greatly varies between cities and education levels (Saraiva et al., 2023).

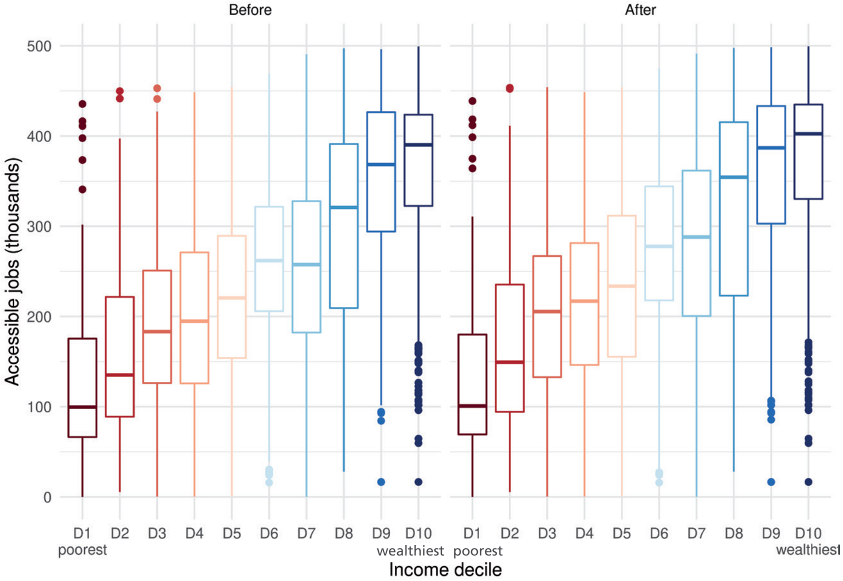
6.6 Accessibility inequality

A key dimension when assessing transport policies is related to their distributive aspects. Who are the winners and losers? From an equity perspective, we expect public policies to prioritize improvements on the accessibility conditions of those with worse socioeconomic conditions and who depend on public transport the most (Pereira, Schwanen and Banister, 2017; Van Wee, 2022).

In this section, we look at how the job accessibility gains that result from the East line implementation coupled with the changes to subway and light rail frequencies are distributed between different income groups. To do this, we need to understand how the accessibility levels were distributed among the population in 2019, before the transport intervention, and how they will be after the implementation of such changes. With the code below, we use the classification of each hexagon in terms of income decile to investigate the accessibility distribution between income groups before and after the changes to the transport system.

```r
ggplot(data = access_to_jobs[population > 0]) +
  geom_boxplot(
    aes(
      x = as.factor(decile),
      y = access_to_jobs / 1000,
      color = as.factor(decile),
      weight = population,
      group = decile
    ),
    show.legend = FALSE
  ) +
  facet_wrap(
    ~ scenario,
    nrow = 1,
    labeller = as_labeller(c(before = "Before", after = "After"))
  ) +
  scale_colour_brewer(palette = "RdBu") +
  labs(x = "Income decile", y = "Accessible jobs (thousands)") +
  scale_x_discrete(
    labels = c("D1\npoorest", paste0("D", 2:9), "D10\nwealthiest")
  ) +
  theme_minimal()
```

FIGURE 18
Job accessibility distribution between income deciles by transport scenario



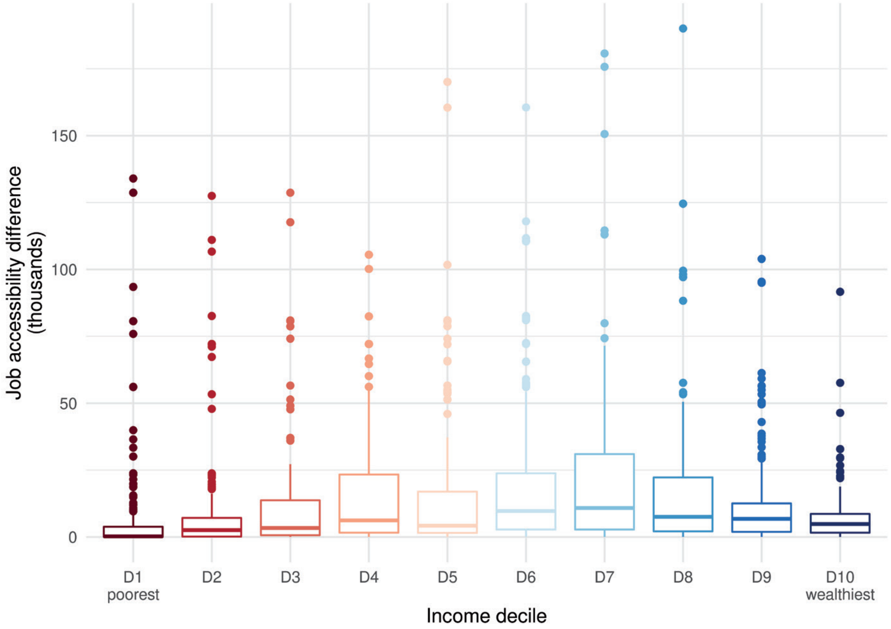
Source: Figure generated by the code snippet above.

This figure clearly shows that the wealthiest people in Fortaleza have higher job accessibility than their poorer counterparts, both before and after the changes to the transport system. In Fortaleza, as in most Brazilian cities, the wealthiest populations tend to live closer to the city center and areas with higher concentration of jobs, whereas the poorest tend to reside in the city's outskirts (Pereira et al., 2022b). Consequently, the wealthiest usually have better urban accessibility conditions than the poorest. Not only because they tend to live closer to their jobs, but also because these regions tend to be better served by public transport than the urban peripheries.

However, it is difficult to see in this figure the magnitude of the variation in accessibility between the two scenarios. Using the same strategy that we have previously used, we present in the following figure the distribution of accessibility *gains* between scenarios by income decile:

```
ggplot(subset(access_to_jobs, population > 0 & !is.na(difference))) +
  geom_boxplot(
    aes(
      x = as.factor(decile),
      y = difference / 1000,
      color = as.factor(decile),
      weight = population,
      group = decile
    ),
    show.legend = FALSE
  ) +
  scale_colour_brewer(palette = "RdBu") +
  labs(
    x = "Income decile",
    y = "Job accessibility difference\n(thousands)"
  ) +
  scale_x_discrete(
      labels = c("D1\npoorest", paste0("D", 2:9), "D10\nwealthiest")
  ) +
  theme_minimal()
```

FIGURE 19
Distribution of accessibility gains between transport scenarios by income decile



Source: Figure generated by the code snippet above.

As we can see, the distribution of accessibility gains follows an inverted-U shape, with middle-income groups concentrating larger gains than the poorest and wealthiest populations. The hexagon that gained the most accessibility appears as an outlier of the 8th decile category, with an accessibility excess between scenarios of almost 200,000 jobs.

Charts such as the ones shown in the last two figures contain lots of information, and that's why they are not the simplest to communicate. To facilitate this communication, summary measures are frequently used to assess the impact of transport policies on accessibility inequalities. This type of measure tries to summarize the distribution of accessibility levels among population groups (here, income deciles) into a single indicator that facilitates the understanding and interpretation of the results and is frequently used, for example, when developing plans and setting goals. In the accessibility literature, two of the most frequently used inequality measures are the Palma Ratio and the Gini Index (Lucas, Van Wee and Maat, 2016; Guzman and Oviedo, 2018; Pritchard et al., 2019).

In this example, we calculate the Palma Ratio before and after the interventions to the transport system. This measure is the result of dividing the average accessibility of the wealthiest 10% by the average accessibility of the poorest 40%:

$$P = \frac{\overline{A_{tp10}}}{\overline{A_{bt40}}} \tag{7}$$

In which P is the Palma Ratio, $\overline{A_{tp10}}$ is the average accessibility of the richest 10% and $\overline{A_{bt40}}$ is the average accessibility of the poorest 40%.

BOX 5

Why use the Palma Ratio?

One of the main advantages of the Palma Ratio over the Gini Index is how easy it is to communicate and interpret its results. Values higher than 1 indicate a scenario in which the wealthiest have higher average accessibility levels than the poorest, and values lower than 1 the opposite situation. Another advantage of the Palma Ratio is that it clearly reflects how the inequality varies between two groups of particular interest to us: the most privileged and the most vulnerable in a population. The Gini Index, on the other hand, estimates how much a distribution deviates from a hypothetical situation in which everyone has the exact same access level, but says nothing about the socioeconomic conditions of those with the highest and lowest accessibility levels. If a given policy increases the accessibility levels of wealthy people that live in low-accessibility regions, for example, the Gini Index would point to an inequality decrease, even if not a single vulnerable citizen had benefited from this policy. Such a policy can hardly be assessed as equitable, even if the summary measure (the Gini Index, in this case) suggests otherwise.

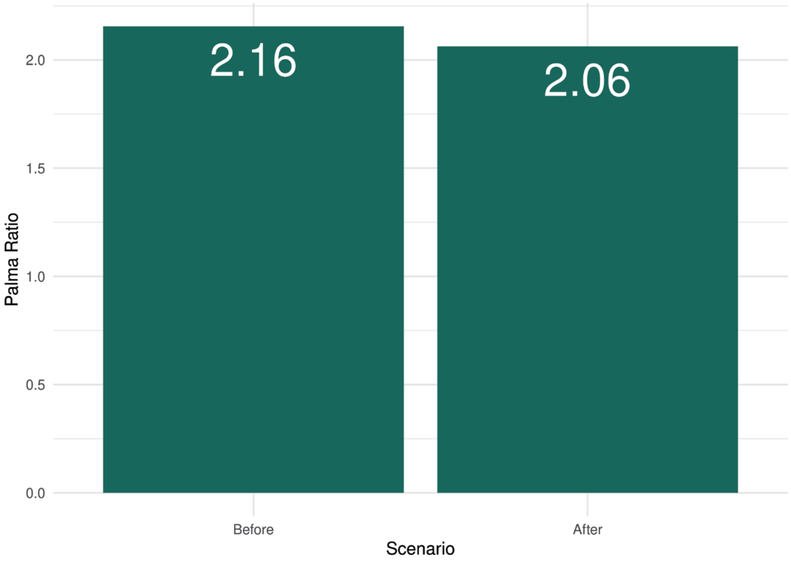
Authors' elaboration.

Calculating the Palma Ratio before and after the East line implementation and the changes to the subway and light rail frequencies allows us to understand how these policies will impact the job accessibility inequality in Fortaleza:

```r
# calculates the wealthiest’s average accessibility in both scenarios
wealthiest_access <- access_to_jobs[
  decile == 10,
  .(access = weighted.mean(access_to_jobs, w = as.numeric(population))),
  by = scenario
]

# calculates the poorest’s average accessibility in both scenarios
poorest_access <- access_to_jobs[
  decile %in% 1:4,
  .(access = weighted.mean(access_to_jobs, w = as.numeric(population))),
  by = scenario
]

# combines the wealthiest’s and the poorest’s accessibility
palma_ratio <- merge(
  wealthiest_access,
  poorest_access,
  by = "scenario",
  suffixes = c("_wealthiest", "_poorest")
)

# calculates the palma ratio
palma_ratio[, palma := access_wealthiest / access_poorest]

ggplot(data = palma_ratio, aes(x = scenario, y = palma)) +
  geom_col(fill = "#0d6556") +
  geom_text(
    aes(label = round(palma, digits = 2)),
    vjust = 1.5,
    color = "white",
    size = 10
  ) +
  ylab("Palma Ratio") +
  scale_x_discrete(name = "Scenario", labels = c("Before", "After")) +
  theme_minimal()
```

FIGURE 20

Job accessibility Palma Ratio in Fortaleza by transport scenario



Source: Figure generated by the code snippet above.

The figure above shows that, in 2019, the wealthiest groups in Fortaleza could access, on average, 2.16 times more jobs by public transport in 60 minutes than the poorest population. The chart also shows that the inequality, as measured by the Palma Ratio, slightly decreased between the pre- and post-intervention scenarios. Thus, we can say that, in this simplified case study, the proposed subway expansion combined with the changes to the subway and light rail frequencies will be slightly progressive. In other words, these interventions will reduce the job accessibility inequality between high- and low-income populations in Fortaleza.[21]

In this chapter, we have focused on assessing the accessibility impacts of a transport policy. It's worth noting, however, that a complete assessment of a public policy must also consider other criteria, such as community engagement with the policy development and decision-making process, as well as other environmental, economic and social impacts of the policy. Although an accessibility impact assessment is very important to determine who benefits from the transport policy and how such policy impacts the performance of the transport network, this type of analysis only looks at a single impact dimension, and should be complemented by other analyses.

[21] It's important to emphasize that the project assessment presented in this chapter looks at a simplified intervention scenario for didactic purposes. For a more complete assessment of the East line implementation and the changes foreseen in Pasfor, which also includes changes to the bus network, please see Braga et al. (2022).

SECTION 5

DATA FROM THE ACCESS TO OPPORTUNITY PROJECT

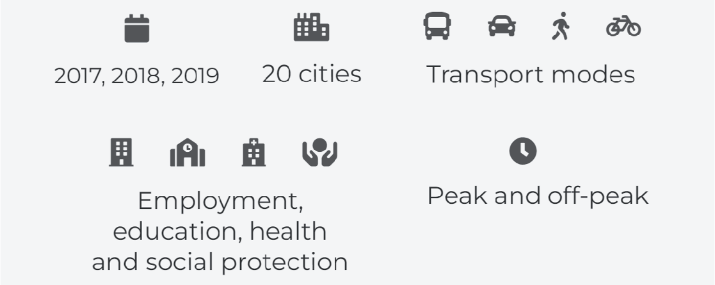
The purpose of this section is: i) to present the accessibility, land use and socioeconomic data made available by AOP; and ii) to teach how to download and use this data with the `{aopdata}` R package.

In the previous chapters, we learned about the concept of accessibility, how this concept is usually translated into quantitative measures and how to estimate accessibility levels using R. However, we often come across situations in which we do not want to calculate accessibility levels ourselves, either because we do not have the resources or data required for this or simply because they have already been calculated by other people. Throughout the next chapters, we will present the accessibility database created and made available by AOP.

AOP is a research initiative led by Ipea with the objective of investigating the urban accessibility conditions and the inequalities in access to opportunities in Brazilian cities. All data produced by the project is made publicly available, including not only accessibility estimates, but also information on the spatial distribution of population, economic activities and public services.[22] The data is spatially aggregated into a hexagonal grid indexed by the H3 geospatial indexing system, originally developed by Uber (Brodsky, 2018). Each hexagonal cell covers around 0.11 km², an area similar to that covered by a city block, resulting in high spatial resolution analyses and outputs. As shown in figure 21, accessibility estimates have been produced for the years 2017, 2018 and 2019 and for the 20 largest Brazilian cities, considering different transport modes (walking, cycling, public transport and automobile), times of day (peak and off-peak), population groups (aggregated by income, race, sex and age) and types of activity (jobs, schools, health services and social assistance centers).

[22] The methods used to generate these datasets are presented in detail in two separate publications, one for population and land use data (Pereira et al., 2022a) and another for accessibility data (Pereira et al., 2022b).

FIGURE 21
Scope of AOP data



2017, 2018, 2019

20 cities

Transport modes

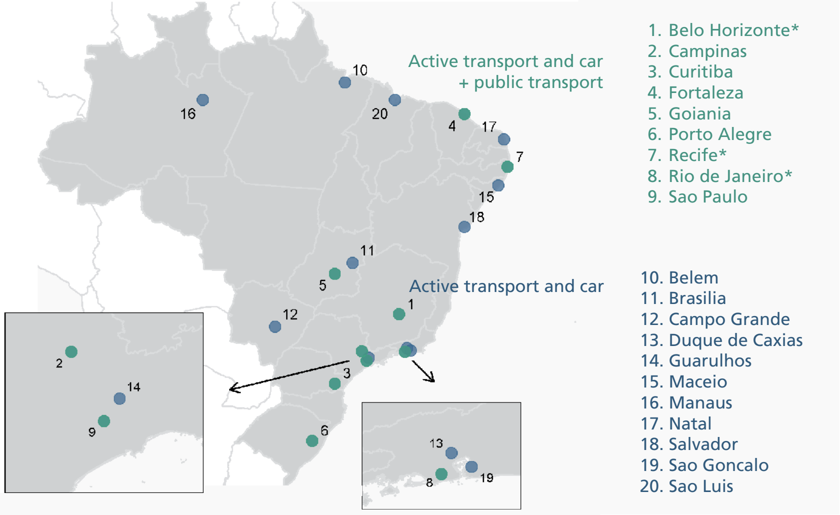
Employment, education, health and social protection

Peak and off-peak

Authors' elaboration.

Accessibility estimates by public transport were calculated only for cities with good quality GTFS data, which include Belo Horizonte, Campinas, Curitiba, Fortaleza, Goiânia,[23] Porto Alegre, Recife, Rio de Janeiro and São Paulo. Still, in some cases the feeds of some of these cities were either only available for a few years or had quality issues, not being representative of the public transport operations they should be describing. In such cases, accessibility estimates by public transport were not calculated. Figure 22 shows the cities included in the project and the transport modes considered in their accessibility estimates.

FIGURE 22
Geographic scope of AOP data



Authors' elaboration.
Obs.: Cities highlighted by an asterisk do not include accessibility estimates by public transport for all years.

[23] Goiânia's GTFS covers not only the municipality, but its entire metropolitan region.

The following tables summarize the data made available by the project. Table 10 describes the urban accessibility dataset.

TABLE 10
Accessibility indicators calculated in AOP

Indicator (code)	Description	Type of opportunities	Travel time thresholds
Minimum travel time (TMI)	Time to the nearest opportunity	Health, education and social assistance reference centers (CRAS)	Walk (60 minutes); bicycle, public transport and car (120 minutes)
Active cumulative accessibility measure (CMA)	Number of accessible opportunities within a given travel time threshold	Jobs, health, education and CRAS	Walk and bicycle (15, 30, 45 and 60 minutes); public transport and car (15, 30, 60, 90 and 120 minutes)
Passive cumulative accessibility measure (CMP)	Number of people that can access the grid cell within a given travel time threshold	-	Walk and bicycle (15, 30, 45 and 60 minutes); public transport and car (15, 30, 60, 90 and 120 minutes)

Authors' elaboration.

Table 11 describes the dataset containing the sociodemographic characteristics of the population and the spatial distribution of opportunities.

TABLE 11
Data on the sociodemographic characteristics of the population and the spatial distribution of activities aggregated by AOP, by year of reference and data source

Data	Information	Years	Source
Sociodemographic characteristics of the population	Number of people by sex, age and race; average income per capita	2010	Demographic Census from the Brazilian Institute for Geography and Statistics (IBGE)
Education services	Number of public schools by education level (early childhood, primary and secondary education)	2017 2018 2019	School Census from the Anísio Teixeira National Institute for Educational Studies and Research (Inep)
Health services	Number of health facilities that serve the Unified Health System (SUS) by complexity level (low, medium and high complexity)	2017 2018 2019	National Registry of Health Facilities (CNES) from the Ministry of Health

(Continues)

(Continued)

Data	Information	Years	Source
Economic activity	Number of formal jobs by education level of workers (primary, secondary and tertiary education)	2017 2018 2019	Annual Relation of Social Information (RAIS) from the Ministry of Economy
Social welfare services	Number of CRAS	2017 2018 2019	Unified Social Assistance System (SUAS) from the Ministry of Citizenship

Authors' elaboration.

All datasets created by AOP are available for download in the project website and through the `{aopdata}` R package. The data dictionary can be accessed online[24] or with the command `aopdata::aopdata_dictionary(lang = "en")` in an R session. The chapters in this section provide several examples illustrating how to download the datasets and create visualizations with them in R.

[24] Available at: https://ipeagit.github.io/aopdata/articles/data_dic_en.html.

7 POPULATION AND SOCIOECONOMIC DATA

The sociodemographic data used in AOP, including aggregate information on the spatial distribution of the population and of their characteristics in terms of income per capita, race, sex and age, comes from the 2010 Census. This dataset can be downloaded in R with the `read_population()` function from the `{aopdata}` package. This function takes a `city` parameter, used to indicate the city whose data must be downloaded. To include the spatial information of each grid cell when downloading the data, the `geometry` parameter, which defaults to `FALSE`, must take the value `TRUE`.

In the example below, we show how to download the population and socioeconomic data of Fortaleza:

```
data_fortaleza <- aopdata::read_population(
  city = "Fortaleza",
  year = 2010,
  geometry = TRUE,
  showProgress = FALSE
)
```

The output includes the Census reference year, columns identifying the grid cells and the municipality and socioeconomic data in multiple columns with encoded names:

```
names(data_fortaleza)
```

```
 [1] "year"       "id_hex"       "abbrev_muni" "name_muni"    "code_muni"
 [6] "P001"       "P002"         "P003"        "P004"         "P005"
[11] "P006"       "P007"         "P010"        "P011"         "P012"
[16] "P013"       "P014"         "P015"        "P016"         "R001"
[21] "R002"       "R003"         "geometry"
```

Table 12 presents the data dictionary with the description of each column, as well as observations about some of the values. This description can also be found in the documentation of the function, running the command `? read_population` in an R session.

TABLE 12
Description of the columns in the population and socioeconomic dataset

Column	Description	Observation
`year`	Reference year	-
`id_hex`	Unique hexagon identifier	-
`abbrev_muni`	3-letter abbreviation of municipality name	-
`name_muni`	Municipality name	-
`code_muni`	7-digit municipality IBGE code	-
`P001`	Total number of people	-
`P002`	Number of white people	-
`P003`	Number of black people	-
`P004`	Number of indigenous people	-
`P005`	Number of people of yellow color	-
`P006`	Number of men	-
`P007`	Number of women	-
`P010`	Number of people from 0 to 5 years old	-
`P011`	Number of people from 6 to 14 years old	-
`P012`	Number of people from 15 to 18 years old	-
`P013`	Number of people aged 19 to 24 years old	-
`P014`	Number of people aged 25 to 39 years old	-
`P015`	Number of people aged 40 to 69 years old	-
`P016`	Number of people aged 70 years old and over	-
`R001`	Average income per capita	Values from 2010, in Brazilian Reais (BRL)
`R002`	Income quintile	Values range from 1 (poorest) to 5 (richest)
`R003`	Income decile	Values range from 1 (poorest) to 10 (richest)
`geometry`	Spatial geometry	-

Authors' elaboration.

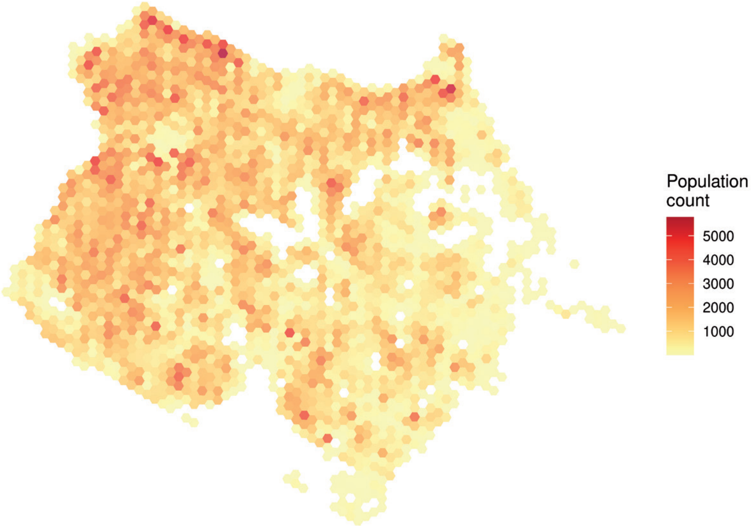
The following sections show a few examples illustrating how to create spatial visualizations out of this dataset.

7.1 Population spatial distribution

In the code below, we load a couple data visualization packages and configure the map. With a single command, we can visualize the population spatial distribution in Fortaleza. The figure shows a choropleth map in which the color of each grid cell represents the number of people that reside there (variable P001).

```
library(patchwork)
library(ggplot2)

ggplot(subset(data_fortaleza, P001 > 0)) +
  geom_sf(aes(fill = P001), color = NA, alpha = 0.8) +
  scale_fill_distiller(palette = "YlOrRd", direction = 1) +
  labs(fill = "Population\ncount") +
  theme_void()
```

FIGURE 23
Population spatial distribution in Fortaleza



Source: Figure generated by the code snippet above.

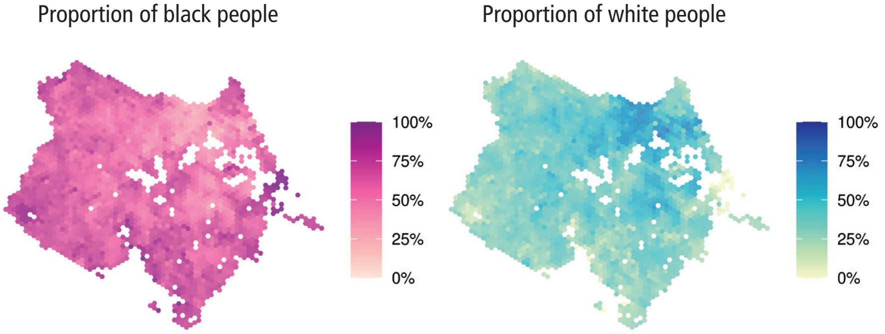
7.2 Population spatial distribution by race

Besides reporting the total population count in each cell, the dataset also includes information on population count by race (variables P002 to P005), sex (variables P006 and P007) and age (variables P010 to P016). The code below illustrates how simple it is to calculate the proportion of black and white people in each hexagon and visualize this information on a map.

```r
pop_black <- ggplot(subset(data_fortaleza, P001 > 0)) +
  geom_sf(aes(fill = P003 / P001), color = NA, alpha = 0.8) +
  scale_fill_distiller(
    name = NULL,
    palette = "RdPu",
    direction = 1,
    labels = scales::percent,
    limits = c(0, 1)
  ) +
  labs(title = "Proportion of black people") +
  theme_void()

pop_white <- ggplot(subset(data_fortaleza, P001 > 0)) +
  geom_sf(aes(fill = P002 / P001), color = NA, alpha = 0.8) +
  scale_fill_distiller(
    name = NULL,
    palette = "YlGnBu",
    direction = 1,
    labels = scales::percent,
    limits = c(0, 1)
  ) +
  labs(title = "Proportion of white people") +
  theme_void()

pop_black + pop_white
```

FIGURE 24

Proportion of black and white people in Fortaleza



Proportion of black people

Proportion of white people

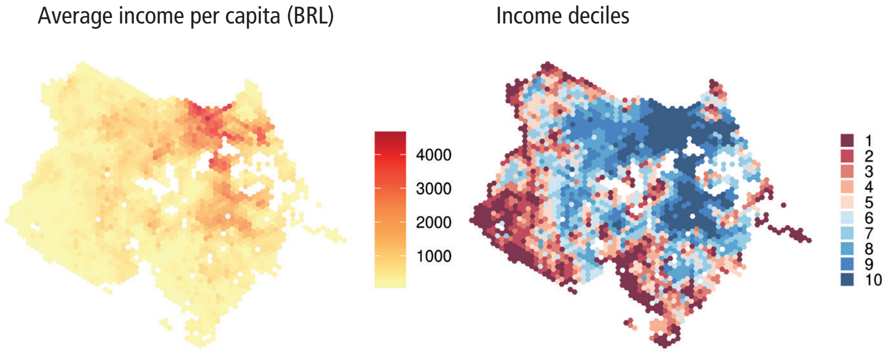
Source: Figure generated by the code snippet above.

7.3 Income spatial distribution

Finally, the dataset also includes information on the average income per capita of each hexagon (R001) and their classification in terms of income quintile (R002) and decile (R003). Using this data, we can visualize the income spatial distribution in the city.

```
income <- ggplot(subset(data_fortaleza, P001 > 0)) +
  geom_sf(aes(fill = R001), color = NA, alpha = 0.8) +
  scale_fill_distiller(name = NULL, palette = "YlOrRd", direction = 1) +
  labs(title = "Average income per capita (BRL)") +
  theme_void()

deciles <- ggplot(subset(data_fortaleza, !is.na(R002))) +
  geom_sf(aes(fill = factor(R003)), color = NA, alpha = 0.8) +
  scale_fill_brewer(name = NULL, palette = "RdBu") +
  labs(title = "Income deciles") +
  theme_void() +
  theme(legend.key.size = unit(0.3, "cm"))

income + deciles
```

FIGURE 25
Income spatial distribution in Fortaleza



Average income per capita (BRL)

Income deciles

Source: Figure generated by the code snippet above.

8 DATA ON THE SPATIAL DISTRIBUTION OF OPPORTUNITIES

The {aopdata} package allows one to download data from 2017, 2018 and 2019 on the spatial distribution of jobs (low, middle and high education), public health facilities (low, medium and high complexity), public schools (early childhood, primary and secondary school levels) and CRAS. This data is available for all cities included in the project.

These datasets can be downloaded with the read_landuse() function, which works similarly to read_population(). To use it, indicate the city whose data should be downloaded using the city parameter, along with the reference year (year) and whether to include the spatial information of grid cells or not (geometry).

In the example below, we show how to download land use data from 2019 for the city of Belo Horizonte. Please note that this function outputs a table that also includes sociodemographic data.

```r
data_bh <- aopdata::read_landuse(
  city = "Belo Horizonte",
  year = 2019,
  geometry = TRUE,
  showProgress = FALSE
)

names(data_bh)
```

```
 [1] "id_hex"      "abbrev_muni" "name_muni"   "code_muni"   "P001"
 [6] "P002"        "P003"        "P004"        "P005"        "P006"
[11] "P007"        "P010"        "P011"        "P012"        "P013"
[16] "P014"        "P015"        "P016"        "R001"        "R002"
[21] "R003"        "year"        "T001"        "T002"        "T003"
[26] "T004"        "E001"        "E002"        "E003"        "E004"
[31] "M001"        "M002"        "M003"        "M004"        "S001"
[36] "S002"        "S003"        "S004"        "C001"        "geometry"
```

Table 13 presents the data dictionary with the description of the table columns (excluding those previously included in the sociodemographic dataset). This description can also be found in the documentation of the function, running the command ? read_landuse in an R session.

TABLE 13
Description of the columns in the land use dataset

Column	Description
`year`	Reference year
`id_hex`	Unique hexagon identifier
`abbrev_muni`	3-letter abbreviation of municipality name
`name_muni`	Municipality name
`code_muni`	7-digit municipality IBGE code
`T001`	Total number of jobs
`T002`	Number of low-education jobs
`T003`	Number of middle-education jobs
`T004`	Number of high-education jobs
`E001`	Total number of public schools
`E002`	Number of public early childhood schools
`E003`	Number of public primary schools
`E004`	Number of public secondary schools
`M001`	Total number of students enrolled in public schools
`M002`	Number of students enrolled in public early childhood schools
`M003`	Number of students enrolled in public primary schools
`M004`	Number of students enrolled in public secondary schools
`S001`	Total number of public health facilities
`S002`	Number of low complexity public health facilities
`S003`	Number of mid complexity public health facilities
`S004`	Number of high complexity public health facilities
`C001`	Total number of CRAS
`geometry`	Spatial geometry

Authors' elaboration.

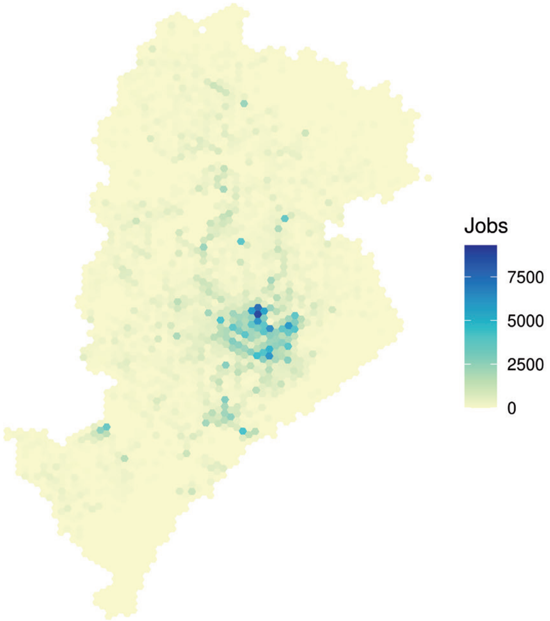
The following sections show a few examples illustrating how to create spatial visualizations out of this dataset.

8.1 Spatial distribution of jobs

In the code below, we load a couple data visualization libraries and configure the map. Columns starting with the letter `T` describe the spatial distribution of jobs in each city. The example shows the spatial distribution of the total number of jobs in each grid cell (variable `T001`) in Belo Horizonte:

```
library(patchwork)
library(ggplot2)

ggplot(data_bh) +
  geom_sf(aes(fill = T001), color = NA, alpha = 0.9) +
  scale_fill_distiller(palette = "YlGnBu", direction = 1) +
  labs(fill = "Jobs") +
  theme_void()
```

FIGURE 26
Spatial distribution of jobs in Belo Horizonte



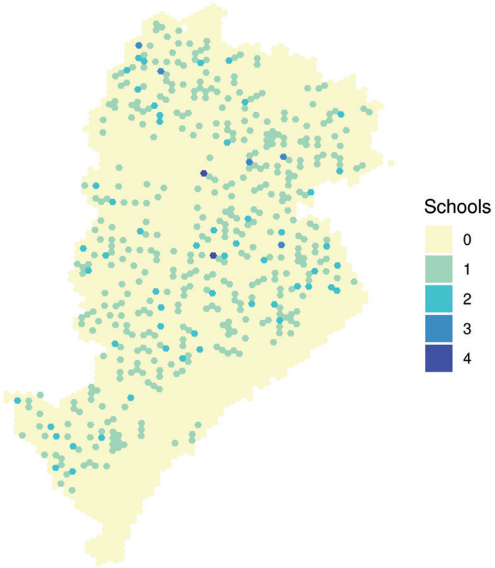
Source: Figure generated by the code snippet above.

8.2 Spatial distribution of public schools

The columns with information on the number of public schools in each cell begin with the letter `E`. In the example below, we present the spatial distribution of all public schools in Belo Horizonte (variable `E001`).

```
ggplot(data_bh) +
  geom_sf(aes(fill = as.factor(E001)), color = NA, alpha = 0.9) +
  scale_fill_brewer(palette = "YlGnBu", direction = 1) +
  labs(fill = "Schools") +
  theme_void()
```

FIGURE 27
Spatial distribution of public schools in Belo Horizonte



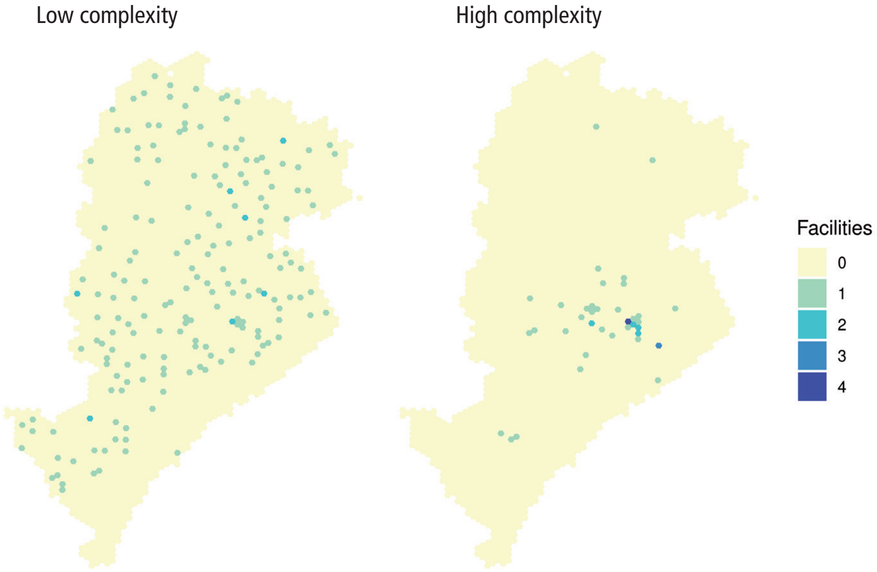
Source: Figure generated by the code snippet above.

8.3 Spatial distribution of public health facilities

The columns with information on the number of public health facilities in each cell begin with the letter `S`. The visualization below compares the spatial distribution of low complexity (`S002`) and high complexity (`S004`) public health facilities.

```
low_complexity <- ggplot(data_bh) +
  geom_sf(aes(fill = as.factor(S002)), color = NA, alpha = 0.9) +
  scale_fill_brewer(palette = "YlGnBu", direction = 1, limits =
factor(0:4)) +
  labs(title = "Low complexity", fill = "Facilities") +
  theme_void()

high_complexity <- ggplot(data_bh) +
  geom_sf(aes(fill = as.factor(S004)), color = NA, alpha = 0.9) +
  scale_fill_brewer(palette = "YlGnBu", direction = 1, limits =
factor(0:4)) +
  labs(title = "High complexity", fill = "Facilities") +
  theme_void()

low_complexity + high_complexity + plot_layout(guides = "collect")
```

FIGURE 28
Spatial distribution of low complexity and high complexity public health facilities in Belo Horizonte



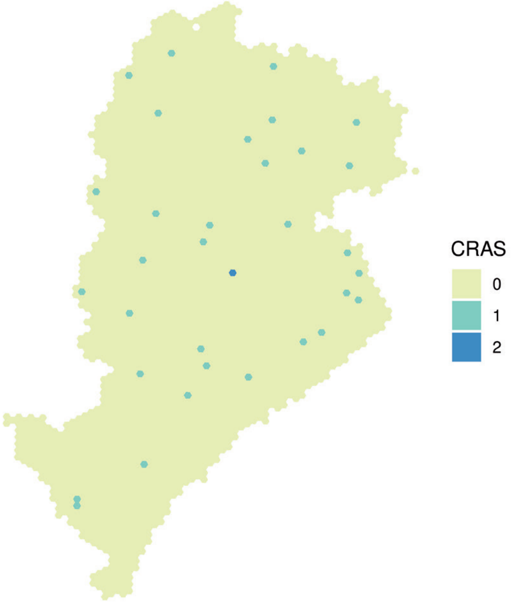
Source: Figure generated by the code snippet above.

8.4 Spatial distribution of CRAS

Finally, the column `C001` has information on the number of CRAS in each grid cell. The map below shows the spatial distribution of these services in Belo Horizonte.

```
ggplot(data_bh) +
  geom_sf(aes(fill = as.factor(C001)), color = NA, alpha = 0.9) +
  scale_fill_brewer(palette = "YlGnBu", direction = 1) +
  labs(fill = "CRAS") +
  theme_void()
```

FIGURE 29
Spatial distribution of CRAS in Belo Horizonte



Source: Figure generated by the code snippet above.

9 ACCESSIBILITY ESTIMATES

Finally, the {aopdata} package also allows one to download estimates of accessibility to jobs, public health facilities, public schools and social assistance services. These estimates were calculated using 2017, 2018 and 2019 as reference years.

This data can be downloaded with the read_access() function, which works similarly to read_population() and read_landuse(). Besides indicating the city (city parameter) and the reference year (year), though, it is also necessary to inform the transport mode (mode) and the interval of the day (peak, between 6 am and 8 am, or off-peak, between 2 pm and 4 pm, controlled by peak) which identify the accessibility data that should be downloaded.

With the code below, we show how to download accessibility estimates that refer to the peak period in São Paulo in 2019. In this example, we downloaded accessibility estimates both by car and by public transport and merged them into a single data.frame. Please note that this function results in a table that also includes sociodemographic and land use data.

```
access_pt <- aopdata::read_access(
  city = "São Paulo",
  mode = "public_transport",
  year = 2019,
  peak = TRUE,
  geometry = TRUE,
  showProgress = FALSE
)

access_car <- aopdata::read_access(
  city = "São Paulo",
  mode = "car",
  year = 2019,
  peak = TRUE,
  geometry = TRUE,
  showProgress = FALSE
)

data_sp <-rbind(access_pt, access_car)
```

```
names(data_sp)
```

```
  [1] "id_hex"       "abbrev_muni"  "name_muni"    "code_muni"    "year"
  [6] "P001"         "P002"         "P003"         "P004"         "P005"
 [11] "P006"         "P007"         "P010"         "P011"         "P012"
 [16] "P013"         "P014"         "P015"         "P016"         "R001"
 [21] "R002"         "R003"         "T001"         "T002"         "T003"
 [26] "T004"         "E001"         "E002"         "E003"         "E004"
 [31] "M001"         "M002"         "M003"         "M004"         "S001"
 [36] "S002"         "S003"         "S004"         "C001"         "mode"
 [41] "peak"         "CMATT15"      "CMATB15"      "CMATM15"      "CMATA15"
 [46] "CMAST15"      "CMASB15"      "CMASM15"      "CMASA15"      "CMAET15"
 [51] "CMAEI15"      "CMAEF15"      "CMAEM15"      "CMAMT15"      "CMAMI15"
 [56] "CMAMF15"      "CMAMM15"      "CMACT15"      "CMPPT15"      "CMPPH15"
 [61] "CMPPM15"      "CMPPB15"      "CMPPA15"      "CMPPI15"      "CMPPN15"
 [66] "CMPP0005I15"  "CMPP0614I15"  "CMPP1518I15"  "CMPP1924I15"  "CMPP2539I15"
 [71] "CMPP4069I15"  "CMPP70I15"    "CMATT30"      "CMATB30"      "CMATM30"
 [76] "CMATA30"      "CMAST30"      "CMASB30"      "CMASM30"      "CMASA30"
 [81] "CMAET30"      "CMAEI30"      "CMAEF30"      "CMAEM30"      "CMAMT30"
 [86] "CMAMI30"      "CMAMF30"      "CMAMM30"      "CMACT30"      "CMPPT30"
 [91] "CMPPH30"      "CMPPM30"      "CMPPB30"      "CMPPA30"      "CMPPI30"
 [96] "CMPPN30"      "CMPP0005I30"  "CMPP0614I30"  "CMPP1518I30"  "CMPP1924I30"
[101] "CMPP2539I30"  "CMPP4069I30"  "CMPP70I30"    "CMATT60"      "CMATB60"
[106] "CMATM60"      "CMATA60"      "CMAST60"      "CMASB60"      "CMASM60"
[111] "CMASA60"      "CMAET60"      "CMAEI60"      "CMAEF60"      "CMAEM60"
[116] "CMAMT60"      "CMAMI60"      "CMAMF60"      "CMAMM60"      "CMACT60"
[121] "CMPPT60"      "CMPPH60"      "CMPPM60"      "CMPPB60"      "CMPPA60"
[126] "CMPPI60"      "CMPPN60"      "CMPP0005I60"  "CMPP0614I60"  "CMPP1518I60"
[131] "CMPP1924I60"  "CMPP2539I60"  "CMPP4069I60"  "CMPP70I60"    "CMATT90"
[136] "CMATB90"      "CMATM90"      "CMATA90"      "CMAST90"      "CMASB90"
[141] "CMASM90"      "CMASA90"      "CMAET90"      "CMAEI90"      "CMAEF90"
[146] "CMAEM90"      "CMAMT90"      "CMAMI90"      "CMAMF90"      "CMAMM90"
[151] "CMACT90"      "CMPPT90"      "CMPPH90"      "CMPPM90"      "CMPPB90"
[156] "CMPPA90"      "CMPPI90"      "CMPPN90"      "CMPP0005I90"  "CMPP0614I90"
[161] "CMPP1518I90"  "CMPP1924I90"  "CMPP2539I90"  "CMPP4069I90"  "CMPP70I90"
[166] "CMATT120"     "CMATB120"     "CMATM120"     "CMATA120"     "CMAST120"
[171] "CMASB120"     "CMASM120"     "CMASA120"     "CMAET120"     "CMAEI120"
[176] "CMAEF120"     "CMAEM120"     "CMAMT120"     "CMAMI120"     "CMAMF120"
[181] "CMAMM120"     "CMACT120"     "CMPPT120"     "CMPPH120"     "CMPPM120"
[186] "CMPPB120"     "CMPPA120"     "CMPPI120"     "CMPPN120"     "CMPP0005I120"
[191] "CMPP0614I120" "CMPP1518I120" "CMPP1924I120" "CMPP2539I120" "CMPP4069I120"
[196] "CMPP70I120"   "TMIST"        "TMISB"        "TMISM"        "TMISA"
[201] "TMIET"        "TMIEI"        "TMIEF"        "TMIEM"        "TMICT"
[206] "geometry"
```

The names of the accessibility estimates columns, such as `CMAEF30`, `TMISB` and `CMPPM60`, result from a combination of three components, as follows.

1) The type of accessibility measure, which is indicated by the first 3 letters of the code. The data includes three types of measures:

   a) `CMA` – active cumulative accessibility;

   b) `CMP` – passive cumulative accessibility; and

   c) `TMI` – minimum travel time to the nearest opportunity.

2) The type of activity to which the accessibility levels were calculated, indicated by the following two letters, in the middle of the column name. The data includes accessibility estimates to various types of activities:

   a) `TT` – all jobs;

   b) `TB` – low education jobs;

   c) `TM` – middle education jobs;

   d) `TA` – high education jobs;

   e) `ST` – all public health facilities;

   f) `SB` – low complexity public health facilities;

   g) `SM` – medium complexity public health facilities;

   h) `SA` – high complexity public health facilities;

   i) `ET` – all public schools;

   j) `EI` – early childhood public schools;

   k) `EF` – primary public schools;

   l) `MS` – secondary public schools;

   m) `MT` – total number of enrollments in public schools;

   n) `MI` – number of enrollments in early childhood public schools;

   o) `MF` – number of enrollments in primary public schools;

   p) `MM` – number of enrollments in secondary public schools; and

   q) `CT` – all CRAS.

In the case of the passive cumulative measure, the letters in the middle of the column name indicate the population group which the accessibility estimates refer to:

a) `PT` – the entire population;

b) `PH` – male population;

c) `PM` – female population;

d) `PB` – white population;

e) `PN` – black population;

f) `PA` – yellow population;

g) `PI` – indigenous population;

h) `P0005I` – population from 0 to 5 years old;

i) `P0614I` – population from 6 to 14 years old;

j) `P1518I` – population from 15 to 18 years old;

k) `P1924I` – population from 19 to 24 years old;

l) `P2539I` – population from 25 to 39 years old;

m) `P4069I` – population from 40 to 69 years old; and

n) `P70I` – population aged 70 years old and over.

3) The travel time threshold used to estimate the accessibility levels, which is indicated by the two numbers at the end of the column name. This component only applies to the active and passive cumulative measure. The data includes accessibility estimates calculated with cutoffs of 15, 30, 45, 60, 90 and 120 minutes, depending on the transport mode.

Examples:

- `CMAEF30`: number of accessible primary public schools within 30 minutes of travel;
- `TMISB`: minimum travel time to the closest low complexity public health facility; and
- `CMPPM60`: number of women that can access a certain grid cell within 60 minutes of travel.

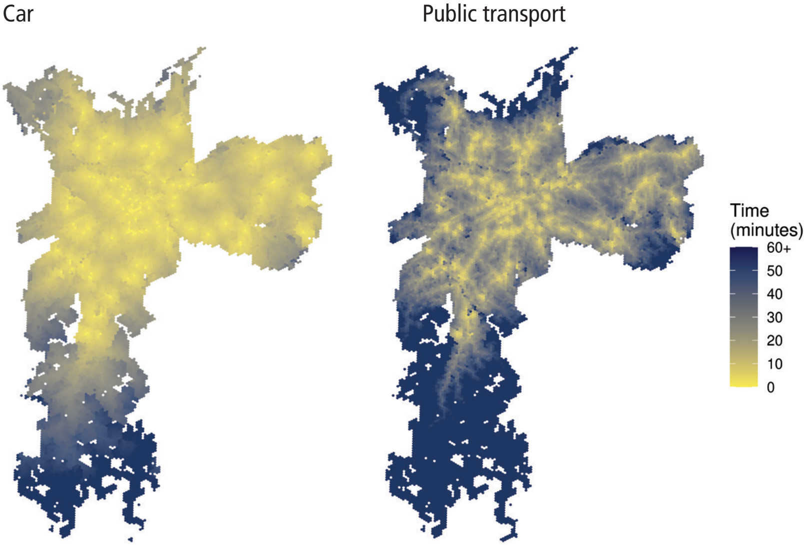
The full description of the columns can also be found in the function documentation, running the `?read_access` command in R. The following sections show examples illustrating how to create spatial visualizations and charts out of the accessibility dataset.

9.1 Map of travel time to access the nearest hospital

In this example, we compare the access time from each grid cell to the nearest public hospital by car and by public transport. To analyze the minimum travel time (TMI) to high complexity public hospitals (SA), we use the TMISA column. With the code below, we load the data visualization libraries and configure the maps showing the spatial distribution of access time by both transport modes. Because public transport trips are usually much longer than car trips, we truncate the travel time distribution to 60 minutes.

```r
library(ggplot2)
library(patchwork)

# truncates travel times to 60 minutes
data_sp$TMISA <- ifelse(data_sp$TMISA > 60, 60, data_sp$TMISA)

ggplot(subset(data_sp, !is.na(mode))) +
  geom_sf(aes(fill = TMISA), color = NA, alpha = 0.9) +
  scale_fill_viridis_c(
    option = "cividis",
    direction = -1,
    breaks = seq(0, 60, 10),
    labels = c(seq(0, 50, 10), "60+")
  ) +
  labs(fill = "Time\n(minutes)") +
  facet_wrap(
    ~ mode,
    labeller = as_labeller(
      c(car = "Car", public_transport = "Public transport")
    )
  ) +
  theme_void()
```

FIGURE 30
Travel time to the closest high complexity public hospital in São Paulo


Car

Public transport

Source: Figure generated by the code snippet above.

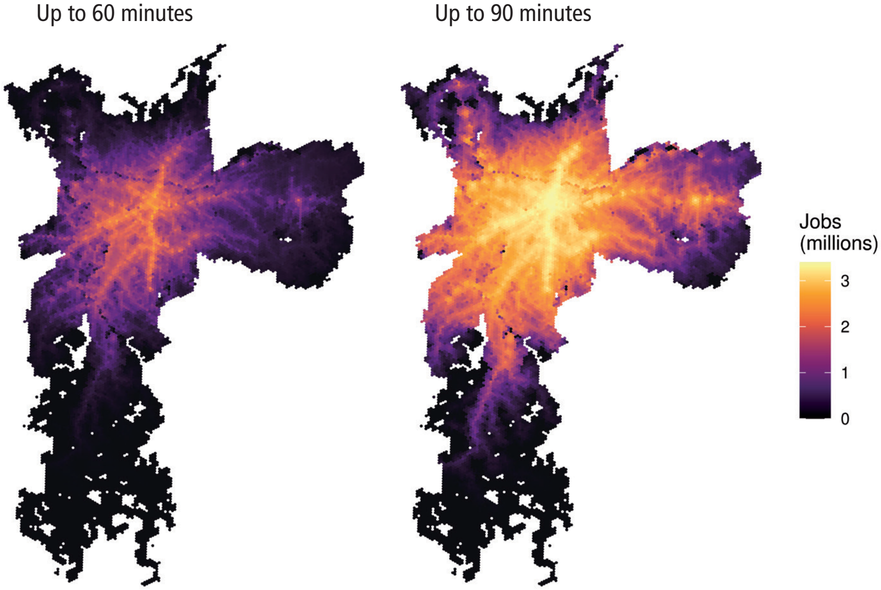
9.2 Map of employment accessibility

The accessibility dataset also makes it very easy to compare the number of accessible opportunities when considering different travel time thresholds. Using the code below, for example, we illustrate how to visualize, side-by-side, the spatial distribution of employment accessibility by public transport trips of up to 60 and 90 minutes.

```r
# determine min and max values for the legend
limit_values  <-c(0, max(access_pt $CMATT90, na.rm = TRUE) / 1000000)

fig60 <- ggplot(subset(access_pt, ! is.na(mode))) +
  geom_sf(aes(fill = CMATT60 / 1000000), color = NA, alpha = 0.9) +
  scale_fill_viridis_c(option = "inferno", limits = limit_values) +
  labs(subtitle = "Up to 60 minutes" , fill = "Jobs\n(millions)") +
  theme_void()

fig90 <- ggplot(subset(access_pt, ! is.na(mode))) +
  geom_sf(aes(fill = CMATT90 / 1000000), color = NA, alpha = 0.9) +
  scale_fill_viridis_c(option = "inferno", limits = limit_values) +
  labs(subtitle = "Up to 90 minutes", fill = "Jobs\n(millions)") +
  theme_void()

fig60 + fig90 + plot_layout(guides = "collect")
```

FIGURE 31

Job accessibility by public transport in São Paulo



Source: Figure generated by the code snippet above.

9.3 Accessibility inequalities

Finally, `{aopdata}` accessibility dataset can be used to analyze accessibility inequalities across different Brazilian cities in several different ways. In this subsection, we present three examples of this type of analysis.

9.3.1 Inequality in travel time to access opportunities

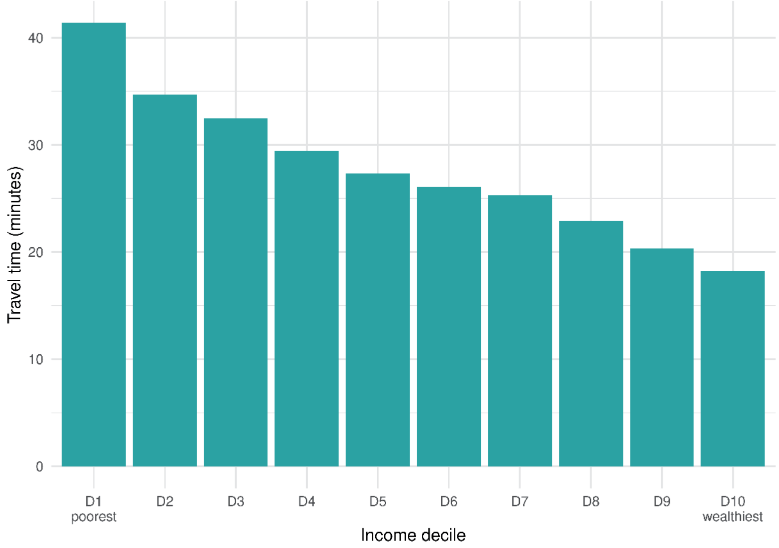
In this first example, we compare the average travel time to the nearest high complexity public hospital for people of different income levels. To do this, we calculate, for each income group, the average travel time to reach the nearest high complexity health facility, weighted by the population of each grid cell. Weighting the travel time by population is necessary because each cell has a different population size, thus contributing differently to the average accessibility of the population as a whole.

Before performing the calculation, we should note that some grid cells cannot reach any high complexity hospital within two hours of travel. In these cases, the minimum travel time columns assume an infinite value (`Inf`). To deal with this situation in our example, we replace all `Inf` values by a travel time of 120 minutes.

```
# copies access data into a new data.frame
ineq_pt <-data.table::as.data.table(access_pt)

# replaces Inf values with 120
ineq_pt [, TMISA := ifelse(is.infinite(TMISA), 120, TMISA)]

# calculates the average travel time by income decile
ineq_pt <- ineq_pt[
  ,
  .(avrg = weighted.mean(x = TMISA, w = P001, na.rm = TRUE)),
  by = R003
]
ineq_pt <- subset(ineq_pt, ! is.na(avrg))

ggplot(ineq_pt) +
  geom_col(aes(y = avrg, x = factor(R003)), fill =
  "#2c9e9e", color = NA) +
  scale_x_discrete(
      labels = c("D1\npoorest", paste0("D", 2:9), "D10\nwealthiest")
  ) +
  labs(x = "Income decile", y = "Travel time (minutes)") +
  theme_minimal()
```

FIGURE 32

Average travel time by public transport to the nearest high complexity hospital in São Paulo



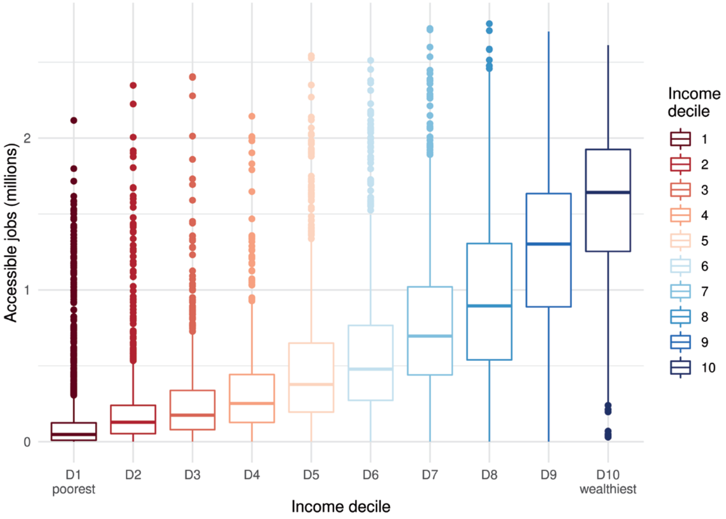
Source: Figure generated by the code snippet above.

9.3.2 Inequality in the number of accessible opportunities

Another way of examining accessibility inequalities is by comparing the number of opportunities that can be reached by different population groups considering the same transport modes and travel time limits. In this case, we analyze the active cumulative accessibility measure, represented by columns whose names start with CMA in the {aopdata} dataset. Using the code below, we compare the number of jobs accessible by people of different income deciles by public transport in up to 60 minutes.

```
ggplot(subset(access_pt, !is.na(R003))) +
  geom_boxplot(
    aes(x = factor(R003), y = CMATT60 / 1000000, color = factor(R003))
  ) +
  scale_color_brewer(palette = "RdBu") +
  labs(
    color = "Income\ndecile",
    x = "Income decile",
    y = "Accessible jobs (millions)"
  ) +
  scale_x_discrete(
    labels = c("D1\npoorest", paste0("D", 2:9), "D10\nwealthiest")
  ) +
  theme_minimal()
```

FIGURE 33
Distribution of job accessibility by public transport in up to 60 minutes of travel in São Paulo



Source: Figure generated by the code snippet above.

Finally, we can also compare how the usage of different transport modes can lead to different accessibility levels and how the discrepancy between modes varies across cities. In the example below, we compare the number of jobs that one can access in up to 30 minutes of walking and driving. To do this, we first download accessibility estimates by both transport modes for all cities covered by AOP.

```r
data_car <- aopdata::read_access(
  city = "all",
  mode = "car",
  year = 2019,
  showProgress = FALSE
)

data_walk <- aopdata::read_access(
  city = "all",
  mode = "walk",
  year = 2019,
  showProgress = FALSE
)
```

Next, we calculate, for each city and transport mode, the weighted average number of jobs accessible by trips of up to 30 minutes (`CMATT30`). We then join these estimates together into a single table and calculate the ratio between car and walk accessibility levels.

```r
avg_car <- data_car[
  ,
  .(access_car = weighted.mean(CMATT30, w = P001, na.rm = TRUE)),
  by = name_muni
]

avg_walk <- data_walk[
  ,
  .(access_walk = weighted.mean(CMATT30, w = P001, na.rm = TRUE)),
  by = name_muni
]

# merges the data and calculates the ratio between access by
# car and on foot
avg_access <- merge(avg_car, avg_walk)
avg_access[, ratio := access_car / access_walk]

head(avg_access)
```

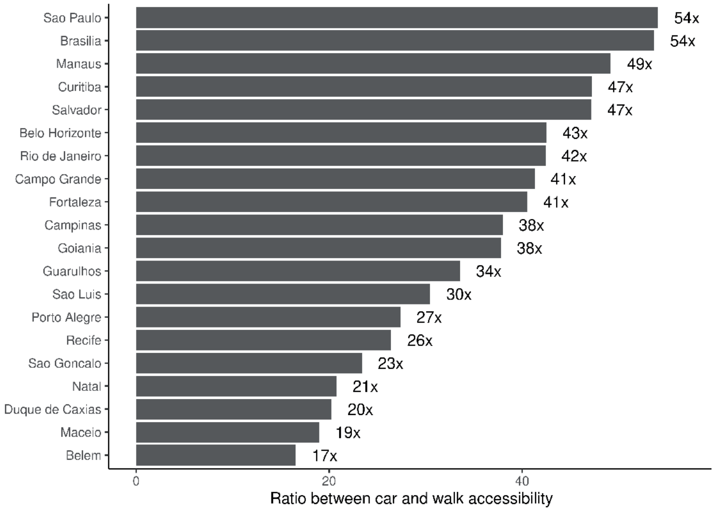
```
          name_muni access_car access_walk    ratio
1:            Belem   155270.4    9392.235 16.53179
2:   Belo Horizonte   529890.0   12464.233 42.51284
3:         Brasilia   220575.9    4110.703 53.65892
4:         Campinas   256333.1    6748.923 37.98133
5:     Campo Grande   172680.5    4181.209 41.29919
6:         Curitiba   494376.9   10471.135 47.21331
```

Finally, we can analyze the results using a chart:

```
ggplot(avg_access, aes(x = ratio, y = reorder(name_muni, ratio))) +
  geom_bar(stat = "identity") +
  geom_text(aes(x = ratio + 3 , label = paste0(round(ratio), "x"))) +
  labs(y = NULL, x = "Ratio between car and walk accessibility") +
  theme_classic()
```

FIGURE 34

Ratio between job accessibility levels by car and by foot considering trips of up to 30 minutes in the 20 biggest Brazilian cities



Source: Figure generated by the code snippet above.

As expected, figure 34 shows that car trips lead to much higher accessibility levels than equally long walking trips. This difference, however, greatly varies across cities. In São Paulo and Brasília, a 30-minute car trip allows one to access, on average, 54 times more jobs than what it would be possible to access with walking trips. In Belém, the city from our sample with the smallest difference, one can access 17 times more jobs by car than by foot – still a substantial difference, but much smaller than in other cities.

REFERENCES

ANDA, C.; ERATH, A.; FOURIE, P. J. Transport modelling in the age of big data. **International Journal of Urban Sciences**, v. 21, n. 1, p. 19-42, 2017. Retrieved from: https://doi.org/10.1080/12265934.2017.1281150.

ARBEX, R.; CUNHA, C. B. Estimating the influence of crowding and travel time variability on accessibility to jobs in a large public transport network using smart card big data. **Journal of Transport Geography**, v. 85, 2020. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2020.102671.

BANISTER, D. The sustainable mobility paradigm. **Transport Policy**, v. 15, n. 2, p. 73-80, 2008. Retrieved from: https://doi.org/10.1016/j.tranpol.2007.10.005.

______. The trilogy of distance, speed and time. **Journal of Transport Geography**, v. 19, n. 4, p. 950-959, 2011. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2010.12.004.

BARRINGTON-LEIGH, C.; MILLARD-BALL, A. The world's user-generated road map is more than 80% complete. **PLOS ONE**, v. 12, n. 8, 2017. Retrieved from: https://doi.org/10.1371/journal.pone.0180698.

BERTOLINI, L.; CLERCQ, F. le; KAPOEN, L. Sustainable accessibility: a conceptual framework to integrate transport and land use plan-making – two test-applications in the Netherlands and a reflection on the way forward. **Transport Policy**, v. 12, n. 3, p. 207-220, 2005. Retrieved from: https://doi.org/10.1016/j.tranpol.2005.01.006.

BOISJOLY, G.; EL-GENEIDY, A. M. How to get there? A critical assessment of accessibility objectives and indicators in metropolitan transportation plans. **Transport Policy**, v. 55, p. 38-50, Apr. 2017. Retrieved from: https://doi.org/10.1016/j.tranpol.2016.12.011.

BRAGA, C. K. V. et al. **Impactos da expansão do metrô de Fortaleza sobre o acesso a oportunidades de emprego, saúde e educação**. Brasília: Ipea, 2022. (Texto para Discussão, n. 2767).

BRODSKY, I. H3: Uber's hexagonal hierarchical spatial index. **Uber Engineering Blog**, 27 June 2018. Retrieved from: https://eng.uber.com/h3/.

BULIUNG, R. et al. More than just a bus trip: school busing, disability and access to education in Toronto, Canada. **Transportation Research Part A**: Policy and Practice, v. 148, p. 496-505, 2021. Retrieved from: https://doi.org/10.1016/j.tra.2021.04.005.

BÜTTNER, B. Accessibility tools for transport policy and planning. In: VICKERMAN, R. (Ed.). **International Encyclopedia of Transportation**. Oxford: Elsevier, 2021. p. 83-86. Retrieved from: https://doi.org/10.1016/B978-0-08-102671-7.10618-9.

CAMBOIM, S. P.; BRAVO, J. V. M.; SLUTER, C. R. An investigation into the completeness of, and the updates to, OpenStreetMap data in a heterogeneous area in Brazil. **ISPRS International Journal of Geo-Information**, v. 4, n. 3, p. 1366-1388, 2015. Retrieved from: https://doi.org/10.3390/ijgi4031366.

CERVERO, R. **Accessible cities and regions**: a framework for sustainable transport and urbanism in the 21st century. Berkeley: Center for Future Urban Transport, 2005. (Working Paper).

CHURCH, A.; FROST, M.; SULLIVAN, K. Transport and social exclusion in London. **Transport Policy**, v. 7, n. 3, p. 195-205, 2000. Retrieved from: https://doi.org/10.1016/S0967-070X(00)00024-X.

CONWAY, M. W.; BYRD, A.; VAN DER LINDEN, M. Evidence-based transit and land use sketch planning using interactive accessibility methods on combined schedule and headway-based networks. **Transportation Research Record**: Journal of the Transportation Research Board, v. 2653, n. 1, p. 45-53, 2017. Retrieved from: https://doi.org/10.3141/2653-06.

DAMIANI, A. et al. Ciência de dados em R. **Curso-R**, 2022. Retrieved from: https://livro.curso-r.com/index.html.

DIJST, M.; JONG, T. de; VAN ECK, J. R. Opportunities for transport mode change: an exploration of a disaggregated approach. **Environment and Planning B**: Planning and Design, v. 29, n. 3, p. 413-430, 2002. Retrieved from: https://doi.org/10.1068/b12811.

DONG, X. et al. Moving from trip-based to activity-based measures of accessibility. **Transportation Research Part A**: Policy and Practice, v. 40, n. 2, p. 163-180, 2006. Retrieved from: https://doi.org/10.1016/j.tra.2005.05.002.

EL-GENEIDY, A. et al. The cost of equity: assessing transit accessibility and social disparity using total travel cost. **Transportation Research Part A**: Policy and Practice, v. 91, p. 302-316, 2016. Retrieved from: https://doi.org/10.1016/j.tra.2016.07.003.

FARRINGTON, J.; FARRINGTON, C. Rural accessibility, social inclusion and social justice: towards conceptualization. **Journal of Transport Geography**, v. 13, n. 1, p. 1-12, 2005. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2004.10.002.

GEURS, K.; VAN WEE, B. Accessibility evaluation of land-use and transport strategies: review and research directions. **Journal of Transport Geography**, v. 12, n. 2, p. 127-140, 2004. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2003.10.005.

GRISÉ, E. et al. Elevating access: comparing accessibility to jobs by public transport for individuals with and without a physical disability. **Transportation Research Part A**: Policy and Practice, v. 125, p. 280-293, 2019. Retrieved from: https://doi.org/10.1016/j.tra.2018.02.017.

GUZMAN, L. A.; OVIEDO, D. Accessibility, affordability and equity: assessing "pro-poor" public transport subsidies in Bogotá. **Transport Policy**, v. 68, p. 37-51, 2018. Retrieved from: https://doi.org/10.1016/j.tranpol.2018.04.012.

HERSZENHUT, D. et al. The impact of transit monetary costs on transport inequality. **Journal of Transport Geography**, v. 99, Feb. 2022. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2022.103309.

HIGGINS, C. et al. Calculating place-based transit accessibility: methods, tools and algorithmic dependence. **Journal of Transport and Land Use**, v. 15, n. 1, 2022. Retrieved from: https://doi.org/10.5198/jtlu.2022.2012.

KANDT, J.; BATTY, M. Smart cities, big data and urban policy: towards urban analytics for the long run. **Cities**, v. 109, Feb. 2021. Retrieved from: https://doi.org/10.1016/j.cities.2020.102992.

KIM, H.-M.; KWAN, M.-P. Space-time accessibility measures: a geocomputational algorithm with a focus on the feasible opportunity set and possible activity duration. **Journal of Geographical Systems**, v. 5, n. 1, p. 71-91, 2003. Retrieved from: https://doi.org/10.1007/s101090300104.

LEVINE, J.; GRENGS, J.; MERLIN, L. A. **From mobility to accessibility**: transforming urban transportation and land-use planning. Ithaca: Cornell University Press, 2019.

LEVINSON, D.; KING, D. **Transport access manual**: a guide for measuring connection between people and places. Sydney: Committee of the Transport Access Manual/University of Sydney, 2020.

LOVELACE, R.; NOWOSAD, J.; MUENCHOW, J. **Geocomputation with R**. London: Chapman and Hall, 2019. Retrieved from: https://geocompr.robinlovelace.net.

LUCAS, K. Transport poverty and its adverse social consequences. **Proceedings of the Institution of Civil Engineers**: Transport, v. 169, n. 6, p. 353-365, 2016. Retrieved from: https://doi.org/10.1680/jtran.15.00073.

LUCAS, K.; VAN WEE, B.; MAAT, K. A method to evaluate equitable accessibility: combining ethical theories and accessibility-based approaches. **Transportation**, v. 43, n. 3, p. 473-490, 2016. Retrieved from: https://doi.org/10.1007/s11116-015-9585-2.

LUO, W.; WANG, F. Measures of spatial accessibility to health care in a GIS environment: synthesis and a case study in the Chicago region. **Environment and Planning B**: Planning and Design, v. 30, n. 6, p. 865-884, 2003. Retrieved from: https://doi.org/10.1068/b29120.

LUZ, G.; PORTUGAL, L. Understanding transport-related social exclusion through the lens of capabilities approach. **Transport Reviews**, v. 42, n. 4, p. 503-525, 2022. Retrieved from: https://doi.org/10.1080/01441647.2021.2005183.

MARTENS, K. Justice in transport as justice in accessibility: applying Walzer's "Spheres of Justice" to the transport sector. **Transportation**, v. 39, n. 6, p. 1035-1053, 2012. Retrieved from: https://doi.org/10.1007/s11116-012-9388-7.

MCHUGH, B. Pioneering open data standards: the GTFS story. In: GOLDSTEIN, B.; DYSON, L. (Ed.). **Beyond transparency**: open data and the future of civic innovation. San Francisco: Code for America Press, 2013. p. 125-135.

MILLER, E. J. Accessibility: measurement and application in transportation planning. **Transport Reviews**, v. 38, n. 5, p. 551-555, 2018. Retrieved from: https://doi.org/10.1080/01441647.2018.1492778.

NEUTENS, T. et al. Equity of urban service delivery: a comparison of different accessibility measures. **Environment and Planning A**: Economy and Space, v. 42, n. 7, p. 1613-1635, 2010.

NEUTENS, T. et al. An analysis of day-to-day variations in individual spacetime accessibility. **Journal of Transport Geography**, v. 23, p. 81-91, July 2012. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2012.04.001.

PÁEZ, A.; HIGGINS, C. D.; VIVONA, S. F. Demand and level of service inflation in Floating Catchment Area (FCA) methods. **PLOS ONE**, v. 14, n. 6, 2019. Retrieved from: https://doi.org/10.1371/journal.pone.0218773.

PÁEZ, A.; SCOTT, D. M.; MORENCY, C. Measuring accessibility: positive and normative implementations of various accessibility indicators. **Journal of Transport Geography**, v. 25, p. 141-153, Nov. 2012. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2012.03.016.

PAPA, E. et al. Accessibility instruments for planning practice: a review of European experiences. **Journal of Transport and Land Use**, v. 9, n. 3, p. 57-75 2015. Retrieved from: https://doi.org/10.5198/jtlu.2015.585.

PEREIRA, R. H. M.; ANDRADE, P. R.; VIEIRA, J. P. B. Exploring the time geography of public transport networks with the gtfs2gps package. **Journal of Geographical Systems**, p. 1-14, 2022. Retrieved from: https://doi.org/10.1007/s10109-022-00400-x.

PEREIRA, R. H. M. et al. **Desigualdades socioespaciais de acesso a oportunidades nas cidades brasileiras**: 2019. Brasília: Ipea, 2020. (Texto para Discussão, n. 2535).

PEREIRA, R. H. M. et al. R5r: rapid realistic routing on multimodal transport networks with R5 in R. **Transport Findings**, 5 Mar. 2021. Retrieved from: https://doi.org/10.32866/001c.21262.

PEREIRA, R. H. M. et al. **Distribuição espacial de características sociodemográficas e localização de empregos e serviços públicos das vinte maiores cidades do Brasil**. Brasília: Ipea, 2022a. (Texto para Discussão, n. 2772). Retrieved from: https://doi.org/10.38116/td2772.

PEREIRA, R. H. M. et al. **Estimativas de acessibilidade a empregos e serviços públicos via transporte ativo, público e privado nas vinte maiores cidades do Brasil no período 2017-2019**. Brasília: Ipea, 2022b. (Texto para Discussão, n. 2800).

PEREIRA, R. H. M.; KARNER, A. Transportation equity. In: VICKERMAN, R. (Ed.). **International encyclopedia of transportation**. Oxford: Elsevier, 2021. p. 271-277. Retrieved from: https://doi.org/10.1016/B978-0-08-102671-7.10053-3.

PEREIRA, R. H. M.; SCHWANEN, T.; BANISTER, D. Distributive justice and equity in transportation. **Transport Reviews**, v. 37, n. 2, p. 170-191, 2017. Retrieved from: https://doi.org/10.1080/01441647.2016.1257660.

PRITCHARD, J. P. et al. An international comparison of equity in accessibility to jobs: London, São Paulo and the Randstad. **Transport Findings**, 27 Feb. 2019. Retrieved from: https://doi.org/10.32866/7412.

SARAIVA, M. et al. **Transporte urbano e insuficiência de acesso a escolas no Brasil**. Brasília: Ipea, 2023. (Texto para Discussão, n. 2854).

SILVA, C. et al. Accessibility instruments in planning practice: bridging the implementation gap. **Transport Policy**, v. 53, p. 135-145, Jan. 2017. Retrieved from: https://doi.org/10.1016/j.tranpol.2016.09.006.

VAN WEE, B. Transport modes and accessibility. In: VICKERMAN, R. (Ed.). **International encyclopedia of transportation**. Oxford: Elsevier, 2021. p. 32-37. Retrieved from: https://doi.org/10.1016/B978-0-08-102671-7.10402-6.

______. Accessibility and equity: a conceptual framework and research agenda. **Journal of Transport Geography**, v. 104, Oct. 2022. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2022.103421.

VASCONCELLOS, E. A. Urban transport policies in Brazil: the creation of a discriminatory mobility system. **Journal of Transport Geography**, v. 67, p. 85-91, Feb. 2018. Retrieved from: https://doi.org/10.1016/j.jtrangeo.2017.08.014.

VENTER, C. Assessing the potential of bus rapid transit-led network restructuring for enhancing affordable access to employment: the case of Johannesburg's corridors of freedom. **Research in Transportation Economics**, v. 59, p. 441-449, Nov. 2016. Retrieved from: https://doi.org/10.1016/j.retrec.2016.05.006.

WICKHAM, H.; GROLEMUND, G. **R for data science**. Sebastopol: O'Reilly, 2017. Retrieved from: https://r4ds.had.co.nz/.

Ipea – Institute for Applied Economic Research

PUBLISHING DEPARTMENT

Head of the Publishing Department
Aeromilson Trajano de Mesquita

Assistants to the Head of the Department
Rafael Augusto Ferreira Cardoso
Samuel Elias de Souza

Supervision
Aline Cristine Torres da Silva Martins

Typesetting
Aline Cristine Torres da Silva Martins
Leonardo Simão Lago Alvite
Matheus Manhoni de Paula Alves
Mayara Barros da Mota

Cover design
Rafael H. M. Pereira

The manuscripts in languages other than Portuguese published herein have not been proofread.

Composed in adobe garamond pro 11/13.2 (text)
Frutiger LT pro (headings, graphs and tables)
SF mono regular (codes)
Rio de Janeiro-RJ

Ipea's mission

Enhance public policies that are essential to Brazilian development by producing and disseminating knowledge and by advising the state in its strategic decisions.

9 786556 350653

ipea Institute for Applied Economic Research

MINISTRY OF
PLANNING AND
BUDGET

BRAZILIAN GOVERNMENT
BRASIL
UNITING AND REBUILDING