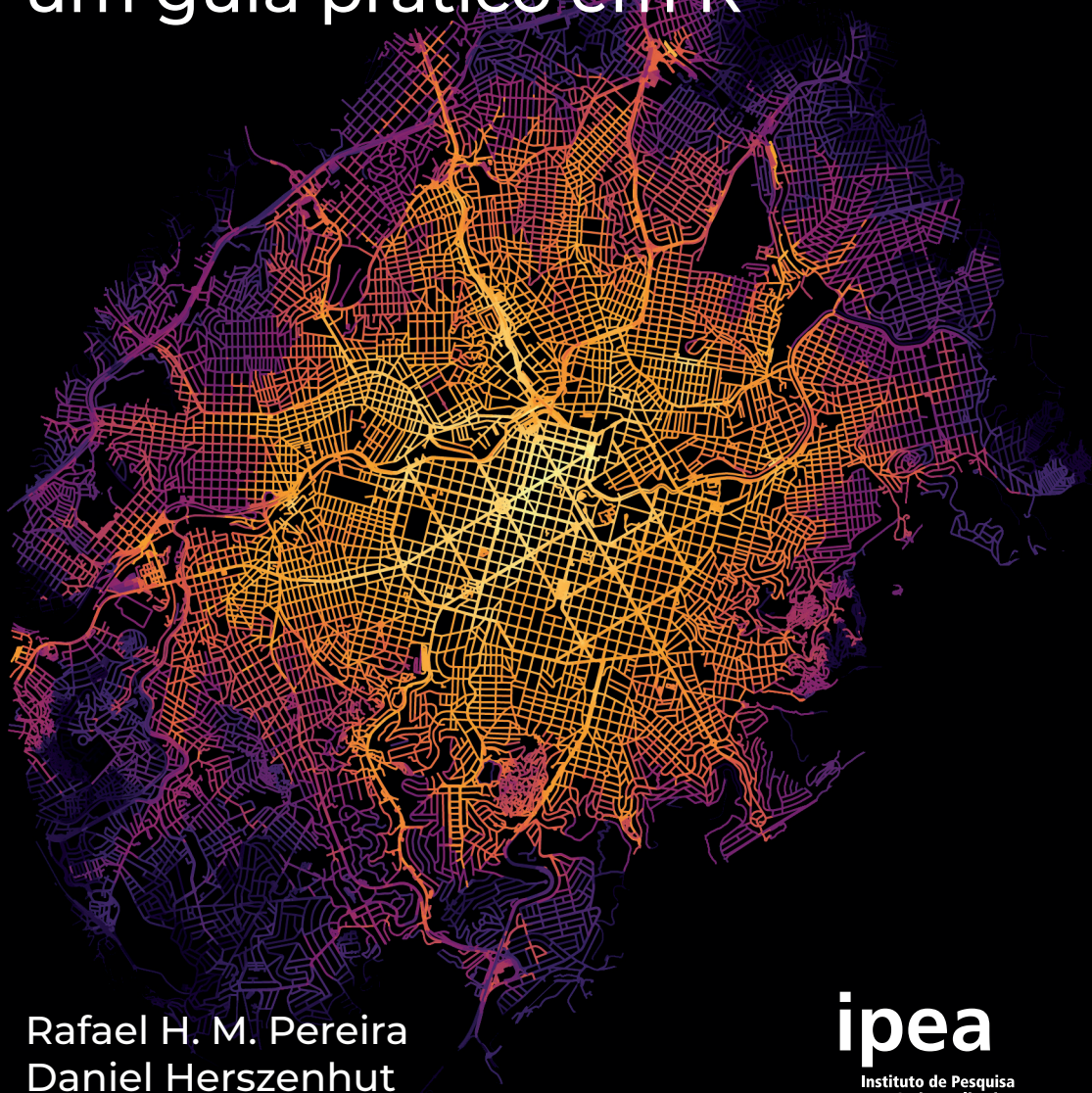


# Introdução à Acessibilidade Urbana

um guia prático em R



Rafael H. M. Pereira  
Daniel Herszenhut

**ipea**

Instituto de Pesquisa  
Econômica Aplicada

# Introdução à Acessibilidade Urbana

um guia prático em R



Rafael H. M. Pereira  
Daniel Herszenhut

**ipea**

## **Governo Federal**

### **Ministério do Planejamento e Orçamento**

**Ministra** Simone Nassar Tebet

# **ipea**

**Instituto de Pesquisa  
Econômica Aplicada**

Fundação pública vinculada ao Ministério do Planejamento e Orçamento, o Ipea fornece suporte técnico e institucional às ações governamentais – possibilitando a formulação de inúmeras políticas públicas e programas de desenvolvimento brasileiros – e disponibiliza, para a sociedade, pesquisas e estudos realizados por seus técnicos.

#### **Presidenta**

Luciana Mendes Santos Servo

#### **Diretor de Desenvolvimento Institucional**

Fernando Gaiger Silveira

#### **Diretora de Estudos e Políticas do Estado, das Instituições e da Democracia**

Luseni Maria Cordeiro de Aquino

#### **Diretor de Estudos e Políticas Macroeconômicas**

Cláudio Roberto Amitrano

#### **Diretor de Estudos e Políticas Regionais, Urbanas e Ambientais**

Aristides Monteiro Neto

#### **Diretora de Estudos e Políticas Setoriais, de Inovação, Regulação e Infraestrutura**

Fernanda De Negri

#### **Diretor de Estudos e Políticas Sociais**

Carlos Henrique Leite Corseuil

#### **Diretor de Estudos Internacionais**

Fábio Vêras Soares

#### **Chefe de Gabinete**

Alexandre dos Santos Cunha

#### **Coordenador-Geral de Imprensa e Comunicação Social**

Antonio Lassance

Ouvidoria: <http://www.ipea.gov.br/ouvidoria>

URL: <http://www.ipea.gov.br>



# Introdução à Acessibilidade Urbana

um guia prático em R



Rafael H. M. Pereira  
Daniel Herszenhut

**ipea**

Rio de Janeiro, 2023



P436i Pereira, Rafael H. M.

Introdução à acessibilidade urbana: um guia prático em R / Rafael H. M. Pereira, Daniel Herszenhut. – Brasília : IPEA, 2023.

155 p. : il.

Inclui referências bibliográficas.

ISBN: 978-65-5635-054-7

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

CDD 362.4

Ficha catalográfica elaborada por Andréa de Mello Sampaio CRB-1/1650

### Como citar:

PEREIRA, Rafael H. M.; HERSZENHUT, Daniel. **Introdução à acessibilidade urbana**: um guia prático em R. Brasília, DF: Ipea, 2023. 155 p., il. color. ISBN: 978-65-5635-054-7. DOI: <http://dx.doi.org/10.38116/9786556350547>.

As publicações do Ipea estão disponíveis para *download* gratuito nos formatos PDF (todas) e EPUB (livros e periódicos). Acesse: <http://www.ipea.gov.br/portal/publicacoes>

As opiniões emitidas nesta publicação são de exclusiva e inteira responsabilidade dos autores, não exprimindo, necessariamente, o ponto de vista do Instituto de Pesquisa Econômica Aplicada ou do Ministério do Planejamento e Orçamento.

É permitida a reprodução deste texto e dos dados nele contidos, desde que citada a fonte. Reproduções para fins comerciais são proibidas.

# SUMÁRIO

<b>APRESENTAÇÃO .....</b>	<b>7</b>
---------------------------	----------

## **SEÇÃO 1 – INTRODUÇÃO À ACESSIBILIDADE URBANA**

CAPÍTULO 1	
<b>O QUE É ACESSIBILIDADE? .....</b>	<b>13</b>

CAPÍTULO 2	
<b>MEDIDAS DE ACESSIBILIDADE .....</b>	<b>19</b>

## **SEÇÃO 2 – COMO CALCULAR ACESSIBILIDADE URBANA**

CAPÍTULO 3	
<b>CALCULANDO ESTIMATIVAS DE ACESSIBILIDADE EM R .....</b>	<b>27</b>

## **SEÇÃO 3 – DADOS DE TRANSPORTE PÚBLICO**

CAPÍTULO 4	
<b>DADOS GTFS .....</b>	<b>47</b>

CAPÍTULO 5	
<b>MANIPULAÇÃO E VISUALIZAÇÃO DE DADOS GTFS.....</b>	<b>59</b>

## **SEÇÃO 4 – AVALIAÇÃO DE IMPACTO DE PROJETOS DE TRANSPORTE**

CAPÍTULO 6	
<b>COMPARANDO A ACESSIBILIDADE ENTRE DOIS CENÁRIOS DE TRANSPORTE.....</b>	<b>85</b>

## **SEÇÃO 5 – DADOS DO PROJETO ACESSO A OPORTUNIDADES**

CAPÍTULO 7	
<b>DADOS SOCIOECONÔMICOS E DE POPULAÇÃO.....</b>	<b>127</b>

CAPÍTULO 8	
<b>DADOS DE DISTRIBUIÇÃO ESPACIAL DE OPORTUNIDADES .....</b>	<b>133</b>

CAPÍTULO 9

<b>ESTIMATIVAS DE ACESSIBILIDADE .....</b>	<b>139</b>
--	------------

<b>REFERÊNCIAS .....</b>	<b>151</b>
--------------------------	------------



## APRESENTAÇÃO<sup>1</sup>

Acessibilidade é a facilidade com que as pessoas conseguem alcançar lugares e oportunidades como empregos, serviços de saúde e educação, atividades culturais, áreas verdes etc. As condições de acessibilidade em uma cidade ou bairro dependem da eficiência e conectividade da rede de transportes e da proximidade entre pessoas e atividades. O tema da acessibilidade tem recebido cada vez mais atenção de agências de transporte, instituições de financiamento, tomadores de decisão e pesquisadores da área de planejamento urbano e de transportes (Papa *et al.*, 2015; Boisjoly e El-Geneidy, 2017). Com isso, existe um crescente número de artigos científicos (Miller, 2018; Van Wee, 2021) e livros (Levine, Grengs e Merlin, 2019; Levinson e King, 2020) que trazem rico material para discussão e aprofundamento sobre questões relacionadas à acessibilidade urbana. No entanto, atualmente não existem livros ou artigos que sirvam simultaneamente de material didático introdutório sobre o tema e de manual prático de metodologia para calcular e analisar dados de acessibilidade. A ausência desse tipo de material explica, ao menos em parte, por que diversas agências de transporte e analistas ainda enfrentam desafios para incorporar questões e indicadores de acessibilidade no dia a dia do planejamento e da pesquisa (Silva *et al.*, 2017; Büttner, 2021).

O objetivo deste livro é equipar seus leitores com os conceitos fundamentais e as ferramentas de análise e processamento de dados necessários para realizar análises de acessibilidade urbana e conduzir avaliações de impacto de projetos e políticas de transporte. O livro foi escrito pensando nas necessidades de trabalho de gestores públicos, analistas, alunos e pesquisadores de planejamento e transporte urbano, e, por isso, possui caráter prático. Todo o material do livro é apresentado com exemplos reproduzíveis e dados abertos, utilizando para isso a linguagem de programação R.<sup>2</sup>

---

1. Este livro foi elaborado pelo Instituto de Pesquisa Econômica Aplicada (Ipea) no âmbito da parceria com a Secretaria de Mobilidade e Desenvolvimento Regional e Urbano do Ministério do Desenvolvimento Regional (SMDRU/MDR). Os autores agradecem os comentários e sugestões de Lucas Mation.

2. Este livro pressupõe que o leitor tenha um conhecimento básico de R. Caso queira se familiarizar com essa linguagem de programação, recomendamos uma leitura de Damiani *et al.* (2022), Wickham e Golemund (2017) e Lovelace, Nowosad e Muenchow (2019).

## 1 ORGANIZAÇÃO DO LIVRO

Este livro está dividido em 5 seções, descritas a seguir.

- 1) *Introdução à acessibilidade urbana*: apresenta o conceito de acessibilidade urbana, esclarece as diferenças entre acessibilidade e mobilidade e aponta os principais indicadores usados para medir a acessibilidade urbana.
- 2) *Como calcular acessibilidade urbana*: ensina como calcular estimativas de acessibilidade urbana em R usando os pacotes `{r5r}` e `{accessibility}` a partir de dados públicos abertos e como visualizar os resultados com mapas e gráficos.
- 3) *Dados de transporte público*: apresenta a Especificação Geral de Redes de Transporte Público (General Transit Feed Specification – GTFS) e mostra como manipular e analisar dados GTFS usando o pacote `{gtfstools}`.
- 4) *Avaliação de impacto de projetos de transporte*: traz um estudo de caso para mostrar como o conhecimento dos capítulos anteriores pode ser utilizado para avaliar o impacto de políticas públicas sobre as condições de acessibilidade urbana.
- 5) *Dados do projeto Acesso a Oportunidades*: mostra como baixar, analisar e visualizar os dados produzidos pelo projeto Acesso a Oportunidades (AOP), que incluem dados detalhados sobre padrões de uso do solo e acessibilidade nas vinte maiores cidades brasileiras.

### *Reproduzindo o livro localmente*

Este livro foi escrito e publicado com o sistema de publicação Quarto. Todo o código utilizado em seu preparo e na sua publicação pode ser encontrado em repositório. Para reproduzir o livro localmente, você deve primeiro baixar o seu código-fonte. Isso pode ser feito com o *software* Git ou manualmente.<sup>3</sup> Caso o segundo método seja utilizado, é necessário também extrair o conteúdo do arquivo `.zip` baixado para uma nova pasta.

A renderização do livro requer que o programa Quarto esteja instalado. A reprodução dos seus capítulos, por sua vez, requer o uso do pacote de R `{renv}`, que faz a gestão de dependências do livro.

---

3. Para baixar o código-fonte manualmente, use o *link* disponível em: [https://github.com/ipeaGIT/intro\\_access\\_book/archive/refs/heads/main.zip](https://github.com/ipeaGIT/intro_access_book/archive/refs/heads/main.zip).

Depois de instaladas as dependências do livro, seu código-fonte poderá ser executado normalmente. Os capítulos estão ordenados numericamente em arquivos no formato .qmd. Para rodar todos os capítulos de uma vez, utilize um dos seguintes comandos:

```
quarto::quarto_preview()  
  
quarto::quarto_render()
```

Para mais detalhes de como instalar as dependências do livro e rodar seu conteúdo localmente, veja as instruções completas de instalação no repositório do livro.

#### *Reproduzindo o livro na nuvem com um binder*

Um *binder* é uma ferramenta que permite rodar códigos na nuvem em um navegador, como o Chrome e o Firefox. O código do livro pode ser rodado usando um servidor publicado pelo MyBinder.<sup>4</sup> Note que as sessões do MyBinder estão limitadas a 2 GB de memória RAM. Essa restrição pode impedir que capítulos de processamento um pouco mais pesado, em particular o capítulo 6, sejam executados adequadamente. Em caso de uso do *binder*, sugerimos que não tente renderizar o livro todo com o Quarto, como mostrado anteriormente.

---

4. Disponível em: <[https://mybinder.org/v2/gh/ipeaGIT/intro\\_access\\_book/HEAD?urlpath=rstudio](https://mybinder.org/v2/gh/ipeaGIT/intro_access_book/HEAD?urlpath=rstudio)>. Após acessar o *link*, uma sessão do RStudio Cloud, que inclui todos os arquivos e dados necessários para rodar o código do livro, começará a rodar no seu navegador.





## INTRODUÇÃO À ACESSIBILIDADE URBANA

Os objetivos desta seção são: i) apresentar o conceito de acessibilidade urbana, esclarecendo a diferença entre acessibilidade e mobilidade; e ii) apresentar uma visão geral dos principais indicadores usados para medir níveis de acessibilidade.

Quantos postos de trabalho uma pessoa em um determinado local consegue alcançar em até uma hora de viagem usando transporte público? Quanto tempo essa pessoa leva para chegar até o posto de saúde ou a escola mais próxima de sua casa? As respostas a essas perguntas dependem diretamente das políticas locais de transporte e de desenvolvimento urbano. Essas políticas determinam os níveis de acessibilidade urbana em cada cidade, isto é, a facilidade com que cada indivíduo consegue acessar oportunidades de emprego, serviços de saúde e educação, atividades culturais e de lazer, entre outros tipos de atividade. A acessibilidade, portanto, é resultado da conectividade e da *performance* dos sistemas de transporte juntamente com a organização espacial das cidades em termos de distribuição populacional, de atividades econômicas e de serviços públicos em seus territórios. Promover um planejamento urbano e de transportes orientado para a promoção de melhores condições de acessibilidade pode contribuir para um desenvolvimento urbano mais sustentável e inclusivo. Ao longo desta seção, vamos entender com mais detalhes o conceito de acessibilidade urbana, por que ele é importante para analisarmos o funcionamento das cidades e quais os indicadores mais comumente utilizados para medir os níveis de acessibilidade.





## 1 O QUE É ACESSIBILIDADE?

### 1.1 Definição de acessibilidade urbana

Acessibilidade é a facilidade com que as pessoas conseguem alcançar lugares e oportunidades – ou, inversamente, uma característica de lugares e oportunidades em termos do quão facilmente eles podem ser alcançados pela população (Geurs e Van Wee, 2004; Neutens *et al.*, 2010).

As condições de acessibilidade são influenciadas tanto pela codistribuição espacial da população e de atividades econômicas e serviços públicos quanto pela configuração e desempenho da rede de transportes. Nesse sentido, a acessibilidade urbana tem papel fundamental na capacidade das pessoas de se deslocarem para acessar oportunidades, como empregos, escolas etc.

Os níveis de acessibilidade urbana são estabelecidos, portanto, por três componentes distintos, conforme a seguir descrito.

- 1) **Infraestrutura:** a facilidade de acessar atividades depende da infraestrutura e dos serviços de transporte existentes. Isso inclui, por exemplo, a capilaridade da rede de transporte público, a conectividade da rede viária, a existência de corredores de transporte de alta capacidade, como trens e metrô etc. Aqui, tanto a eficiência quanto a conectividade espacial e temporal da rede de transportes são de extrema importância.
- 2) **Uso do solo:** o quão facilmente atividades podem ser acessadas também depende da codistribuição espacial de pessoas e atividades, como escolas, serviços de saúde, áreas de lazer etc. Esse componente diz respeito à proximidade geográfica entre pessoas e oportunidades: quanto mais longe, mais difícil é o acesso às atividades.
- 3) **Pessoas:** por fim, é importante ressaltar que a facilidade de acesso a atividades também é afetada pelas características individuais de cada pessoa. Fatores como dificuldades motoras e cognitivas, idade, gênero, cor e renda, por exemplo, podem influenciar de maneira importante a capacidade das pessoas de se locomoverem, de utilizarem determinados modos de transporte e de circularem pela cidade sem medo de algum tipo de violência ou discriminação.

Este último componente traz importantes informações para análises de equidade e inclusão social. No entanto, a sua influência sobre as condições de acessibilidade das pessoas costuma ser melhor captada em análises qualitativas.

Por conta de dificuldades operacionais e computacionais, essa dimensão da acessibilidade costuma receber pouca atenção de avaliações de impacto de projetos de transporte de larga escala. No capítulo 2, discutimos mais a fundo as vantagens e desvantagens operacionais, teóricas e de comunicação de diferentes indicadores de acessibilidade.

## 1.2 Diferença entre microacessibilidade e acessibilidade urbana

Para fins de esclarecimento de conceitos, é importante fazer uma distinção entre o que chamamos de acessibilidade urbana e o que em geral se entende por acessibilidade, em português.

O termo *acessibilidade* é comumente utilizado para se referir a questões de padrões e normas de *design* universal e de construção e planejamento para inclusão de pessoas com diferentes graus de dificuldades motoras e cognitivas. Isso é o que normalmente entende-se por *microacessibilidade*, pois abrange questões de acesso a serviços e atividades na escala micro. Por exemplo, trata de como características do planejamento de espaços públicos e privados e o projeto de construção de edifícios e veículos afetam a capacidade de cada indivíduo de conseguir acessar lugares, serviços, produtos etc.

A *acessibilidade urbana*, por sua vez, pode ser entendida como *macroacessibilidade*, pois trata de questões de acesso de uma maneira mais ampla. Quando falamos de acessibilidade urbana, nos concentramos em questões estruturais de planejamento e desenvolvimento urbano. Ou seja, em como a disposição de corredores de transportes e a distribuição espacial de pessoas e atividades, por exemplo, impactam a capacidade das pessoas de acessar oportunidades. Nesse sentido, ela trata de como a capacidade de acessar atividades é influenciada tanto pela capacidade das pessoas de utilizar tecnologias de transporte quanto pela codistribuição espacial de pessoas e atividades e pela cobertura e conectividade da rede de transporte.

Notadamente, a microacessibilidade e a macroacessibilidade são elementos complementares de uma noção mais ampla de acessibilidade. Condições de microacessibilidade, por exemplo, afetam diretamente a capacidade de pessoas de embarcar e utilizar diferentes modos de transporte, de se locomover com segurança sobre calçadas, de atravessar ruas etc. De pouco adianta um indivíduo morar em uma região atendida por uma grande oferta de modos de transporte se ele tem dificuldades de locomoção, por exemplo, e a rede de transporte não é adaptada para essas dificuldades.

Neste livro, focaremos apenas em análises de acessibilidade urbana e muitas vezes utilizaremos o termo acessibilidade como um sinônimo de macroacessibilidade por uma questão de fluência do texto. É importante reconhecer, no entanto, que a

macroacessibilidade de forma isolada não é suficiente para descrever as condições de acessibilidade de um indivíduo em sua totalidade, e que para tal precisamos olhar também para a microacessibilidade.

### 1.3 Por que a acessibilidade urbana importa?

O conceito de acessibilidade é central em estudos de transporte por diferentes razões. Uma delas se refere ao fato de que este conceito articula de maneira explícita como a interação entre políticas de transporte e de desenvolvimento e uso do solo urbano impacta a capacidade de acesso de pessoas a oportunidades distribuídas nas cidades.

Por sua vez, o acesso a oportunidades e atividades, como postos de trabalho e serviços de educação e de saúde, por exemplo, tem papel fundamental para a satisfação das necessidades individuais e sociais da população e para a promoção da inclusão social (Pereira e Karner, 2021; Luz e Portugal, 2022). Boas condições de acessibilidade também são requisito necessário, embora não suficiente, para a expansão da liberdade de escolha das pessoas (Church, Frost e Sullivan, 2000; Lucas *et al.*, 2016; Van Wee, 2022).

O conceito de acessibilidade, portanto, serve como fio condutor para relacionar investimentos e políticas de transportes e uso do solo a questões referentes à exclusão social e à qualidade de vida de indivíduos, como o potencial de satisfação de necessidades básicas e a liberdade.

Adicionalmente, a ideia de acessibilidade traz à tona a dimensão espacial de um problema central de justiça social: a desigualdade de oportunidades. Ela ajuda a incorporar de maneira explícita a noção de espaço no desenho de políticas destinadas a enfrentar essas injustiças (Farrington e Farrington, 2005; Pereira, Schwanen e Banister, 2017). Assim, a noção de acessibilidade é peça fundamental para pensar a equidade de políticas públicas e para avaliar quais grupos sociais e localidades se beneficiam dessas políticas.

Como vimos anteriormente, os níveis de acesso a oportunidades em uma cidade são resultado conjunto da capacidade de cada pessoa utilizar tecnologias de transporte, da integração entre a distribuição geográfica das atividades e a distribuição espacial da população na cidade e da conectividade espacial e temporal da rede de transportes (Miller, 2018; Páez, Scott e Morency, 2012). Desse modo, o planejamento com foco na acessibilidade passa pelo planejamento integrado entre os sistemas de uso do solo e de transportes, buscando aproximar pessoas e atividades e reduzir a dependência de modos de transporte motorizados (Banister, 2011). Planejar as cidades e sistemas de transporte visando melhorar as condições de acessibilidade, portanto, é essencial para o desenvolvimento de cidades inclusivas e sustentáveis.



#### 1.4 Diferença entre acessibilidade e mobilidade

É importante esclarecer a diferença entre a acessibilidade e outro conceito muito presente em nosso dia a dia: a mobilidade. Infelizmente, a diferença entre esses conceitos é frequentemente ignorada, mesmo por pesquisadores e planejadores que lidam com esses temas diariamente. Afinal, existe uma grande intersecção temática entre a acessibilidade e o que se entende por mobilidade urbana enquanto área de pesquisa e política pública: uma área que lida com os deslocamentos das pessoas e está relacionada ao planejamento de sistemas de transporte coletivo e individual, ao planejamento de redes ciclovárias e de pedestres etc. Nesse contexto, não é raro, por exemplo, ouvir que um determinado grupo “tem menos mobilidade” do que outro, quando, na verdade, o correto seria dizer que esse grupo apresenta piores condições de acessibilidade. Qual é, então, a diferença entre acessibilidade e mobilidade?

Na literatura científica e de planejamento urbano e de transportes, o conceito de mobilidade diz respeito aos padrões de viagens que as pessoas efetivamente fazem no seu dia a dia. Por exemplo, quantas viagens foram feitas, quais modos de transporte foram usados, qual a distância média das viagens, quanto tempo se gasta no deslocamento casa-trabalho etc.

Informações de mobilidade são tradicionalmente captadas por meio de pesquisas domiciliares origem-destino. Recentemente, com o surgimento de novas tecnologias digitais, dados de GPS de telefones celulares, de cartões de bilhetagem eletrônica, de radares e semáforos urbanos, entre outros, também vêm sendo usados com a finalidade de descrever os deslocamentos diários da população (Anda, Erath e Fourie, 2017; Kandt e Batty, 2021). Dados e análises de mobilidade trazem informações sobre o uso do sistema de transportes e sobre os padrões de viagens de indivíduos de diferentes grupos socioeconômicos, o que nos permite captar importantes aspectos do desempenho econômico e ambiental das cidades e do bem-estar da população.

O conceito de acessibilidade, por sua vez, está intrinsecamente relacionado ao *potencial* que as pessoas têm de alcançar atividades e oportunidades. Enquanto uma análise de mobilidade foca, por exemplo, no tempo que as pessoas levam diariamente de sua casa ao seu trabalho, uma análise de acessibilidade tenta identificar questões como a quantidade de empregos que podem ser alcançados dentro de um determinado custo de viagem, ou se as pessoas conseguiriam alcançar serviços públicos em um tempo de viagem tido como razoável.

A acessibilidade trata do quão fácil/factível é alcançar um local, enquanto a mobilidade trata dos meios de deslocamento efetivamente utilizados para chegar até ele. Níveis de acessibilidade são, portanto, medidas potenciais, ao passo que os dados de mobilidade descrevem padrões reais, realizados.

Tradicionalmente, o planejamento urbano e de transportes tem como foco a mobilidade (Banister, 2011; Vasconcellos, 2018; Levine e King, 2020). Ainda hoje, o foco na mobilidade motiva políticas que priorizam a circulação de automóveis e visam aumentar a velocidade e a fluidez de trânsito para reduzir congestionamentos e, conseqüentemente, tempos de deslocamento (Levine, Grengs e Merlin, 2019). Essas políticas, no entanto, tendem a dar um enfoque quantitativo na mobilidade: aumentar o número de viagens, aumentar a velocidade média, diminuir o tempo de congestionamento etc.

Nesse contexto, portanto, a mobilidade é vista como um fim em si mesma, e “melhorá-la” depende de políticas com soluções puramente técnicas que devem “otimizar” os aspectos quantitativos mencionados anteriormente. A mobilidade, no entanto, não pode ser encarada como uma finalidade em si. As pessoas raramente se deslocam pelo simples prazer de se deslocar. Ao contrário, na grande maioria das vezes as pessoas se locomovem para acessar as atividades localizadas no destino da viagem.

Nesse sentido, tem-se observado um crescente consenso entre pesquisadores e agências de transporte de que o objetivo de uma política de transportes é melhorar o acesso da população a bens e oportunidades (Pereira, Schwanen e Banister, 2017; Martens, 2012; Bertolini, Clercq e Kapoen, 2005). Se o que as pessoas querem é acessar atividades, precisamos pensar em formas de planejamento que resultem em políticas que facilitem seu acesso a tais atividades – sem necessariamente promover a motorização e o aumento da velocidade no trânsito, que implicam o crescimento de externalidades econômicas, ambientais, de saúde pública, entre outras.

O que observamos, portanto, é uma mudança de paradigma no planejamento urbano e de transportes: a busca por padrões de viagem mais sustentáveis implica a mudança do foco da mobilidade para a acessibilidade (Banister, 2008; Cervero, 2005; Levine, Grengs e Merlin, 2019).

Desse modo, políticas de aumento de velocidade nas vias e de expansão de faixas, por exemplo, podem ser substituídas por políticas de promoção de maior *mix* de uso do solo e de aproximação de pessoas e atividades, promovendo maior integração entre planejamento de transportes e de uso do solo. Assim, a mudança de foco da mobilidade para a acessibilidade urbana abre um leque maior de possíveis instrumentos e ações de políticas públicas que buscam promover um desenvolvimento urbano mais integrado e calcado na promoção da sustentabilidade e da inclusão social (Banister, 2011; Levine, Grengs e Merlin, 2019).



## 2 MEDIDAS DE ACESSIBILIDADE

A mudança de paradigma no planejamento urbano e de transportes responsável pelo maior foco na melhoria das condições de acessibilidade é acompanhada por diversos desafios. Entre eles está a necessidade de criar métodos que meçam as condições de acessibilidade nas cidades. A busca por estimativas de acessibilidade facilmente comunicáveis, metodologicamente robustas e pouco computacionalmente intensivas levou ao desenvolvimento de um grande número de diferentes medidas (Páez, Scott e Morency, 2012). Essas medidas podem ser divididas em dois grandes grupos: medidas baseadas em lugares e medidas baseadas em pessoas (Dijst, Jong e Van Eck, 2002).

### 2.1 Medidas baseadas em lugares

Medidas baseadas em lugares medem a acessibilidade como uma característica de determinado local. Por simplificação, esses indicadores assumem que todas as pessoas que se encontram em um mesmo local têm as mesmas condições de acesso às atividades distribuídas pela cidade. Ou seja, se uma análise de acessibilidade utiliza uma medida baseada em lugares e divide a área de estudo em uma grade quadriculada, cada célula dessa grade (um quadrado) terá a ela associado um nível de acessibilidade, posteriormente atribuído às pessoas que residem dentro de cada célula. Essas medidas são sensíveis a fatores relacionados à distribuição espacial de atividades e à configuração e ao desempenho da rede de transportes, mas não levam em consideração as características individuais das pessoas.

As medidas desse tipo são as mais amplamente utilizadas por agências de transporte e pesquisadores (Boisjoly e El-Geneidy, 2017; Papa *et al.*, 2015), porque exigem menor quantidade de dados e tendem a ser consideravelmente mais fáceis de serem calculadas e interpretadas do que as baseadas em pessoas. Por isso, os exemplos e estudos de caso apresentados neste e nos próximos capítulos do livro focam somente nesse grupo de medidas.

Medidas de acessibilidade baseadas em lugares associam a cada deslocamento um custo, usualmente expresso em tempo de viagem (El-Geneidy *et al.*, 2016; Venter, 2016). Ou seja, se um local pode ser alcançado a partir de outro em meia hora, o custo para realizar essa viagem é de trinta minutos. Nada impede, no entanto, que outros tipos de custo, como a distância da viagem, seu custo monetário e a percepção de conforto dos usuários, por exemplo, sejam considerados (Arbex e Cunha, 2020; Herszenhut *et al.*, 2022). Apresentamos, a seguir, a descrição de algumas das medidas baseadas em lugares mais frequentemente utilizadas na literatura científica e na prática de agências de transporte. Aqui, o termo custo é

utilizado de maneira ampla e pode se referir a qualquer tipo de unidade utilizada para quantificar a impedância de uma viagem, seja ela tempo de viagem, custo monetário ou demais alternativas.

### 2.1.1 Custo mínimo de viagem

É uma das medidas de acessibilidade mais simples, indicando o menor custo necessário para alcançar a oportunidade mais próxima a partir de uma determinada origem. Ela permite estimar, por exemplo, o tempo de viagem até o posto de saúde mais próximo de cada quarteirão de uma cidade. O indicador é calculado com a seguinte fórmula:

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

Em que  $A_i$  é a acessibilidade na origem  $i$ ,  $c_{ij}$  é o custo de deslocamento entre a origem  $i$  e o destino  $j$ ,  $n$  é o número total de destinos na área de estudo e  $O_j$  é o número de oportunidades no destino  $j$ .

Vantagens e desvantagens: essa medida tem as vantagens de ser fácil de calcular e exigir poucos dados, além de ser fácil de comunicar. Duas desvantagens, no entanto, são que ela não capta a quantidade de oportunidades acessíveis nos destinos para os quais o custo de acesso é o mínimo nem considera aspectos de competição na demanda pelas oportunidades. Por exemplo, mesmo que uma pessoa more muito perto de um hospital, essa proximidade não necessariamente garante um bom acesso aos serviços de saúde se esse for o único hospital da região e estiver sujeito a picos de demanda que sobrecarregam os serviços além de suas capacidades.

### 2.1.2 Medida de oportunidades cumulativas

Mede a quantidade de oportunidades que pode ser alcançada dentro de um determinado limite de custo de viagem. Por exemplo, este indicador pode ser utilizado para medir a quantidade de empregos acessíveis por transporte público em até sessenta minutos ou a quantidade de escolas acessíveis em até trinta minutos de viagem a pé. A medida é calculada com a seguinte fórmula:

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

$$f(c_{ij}) = \begin{cases} 1 & \text{se } c_{ij} \leq C \\ 0 & \text{caso contrário} \end{cases} \quad (3)$$

Em que  $A_i$  é a acessibilidade na origem  $i$ ,  $O_j$  é o número de oportunidades no destino  $j$ ,  $n$  é o número total de destinos na área de estudo,  $f(c_{ij})$  é uma função binária que assume os valores 0 ou 1, a depender do custo de deslocamento  $c_{ij}$  entre a origem  $i$  e o destino  $j$ , e  $C$  é o limite de custo de deslocamento estabelecido.

Vantagens e desvantagens: a medida de oportunidades cumulativas também é calculada com facilidade, exige poucos dados e é fácil de comunicar. Isso contribui para que este indicador seja um dos mais utilizados por agências de transporte e de financiamento em análises de acessibilidade (Papa *et al.*, 2015; Boisjoly e El-Geneidy, 2017). Entre as suas desvantagens estão o fato de que este indicador não considera a influência da competição sobre oportunidades e exige a escolha de um único ponto de corte como limite de custo de viagem. Além disso, esta medida assume que todas as oportunidades que possam ser alcançadas dentro do limite de custo preestabelecido são igualmente desejáveis e alcançáveis pelas pessoas. Por exemplo, se considerarmos um limite de sessenta minutos de tempo de viagem, uma oportunidade a quarenta minutos de uma origem é considerada tão acessível quanto outra que esteja a dez minutos dessa mesma origem.

### 2.1.3 Medidas gravitacionais

Mais do que um tipo de medida específica, podemos entender as medidas gravitacionais como uma família de medidas. Assim como no caso da medida de oportunidades cumulativas, esta família de métricas considera a soma das oportunidades que podem ser alcançadas a partir de um determinado local. A contagem de cada oportunidade, no entanto, é gradualmente descontada à medida que o custo de viagem aumenta. Assim, oportunidades mais fáceis de acessar têm uma importância maior, e o peso de cada oportunidade na soma total diminui com a sua dificuldade de acesso em relação à origem.

O ritmo de decaimento desse peso é afetado pelo custo da viagem e é ditado por uma *função de decaimento*, que pode ser definida de diferentes formas. Por exemplo, a função de decaimento linear considera que o peso de cada oportunidade diminui de maneira constante até um determinado custo limite, em que ele passa a ser zero. Já a função de decaimento exponencial negativa considera que o peso de cada oportunidade é dividido por um fator que cresce exponencialmente, fazendo com que ele diminua rapidamente a custos de viagem baixos e se aproxime de 0 a custos altos. As fórmulas a seguir apresentam a formulação genérica de uma medida gravitacional e as funções de decaimento linear e exponencial negativa, mencionadas anteriormente.

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

$$f_{lin}(c_{ij}) = \begin{cases} 1 - c_{ij}/C & \text{se } c_{ij} \leq C \\ 0 & \text{caso contrário} \end{cases} \quad (5)$$

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

Em que  $A_i$  é a acessibilidade na origem  $i$ ,  $O_j$  é o número de oportunidades no destino  $j$ ,  $n$  é o número total de destinos na área de estudo,  $f(c_{ij})$  é uma função de decaimento cujo resultado varia com o custo de deslocamento  $c_{ij}$  entre a origem  $i$  e o destino  $j$ ,  $f_{lin}(c_{ij})$  é a função de decaimento linear,  $C$  é o limite de custo de deslocamento estabelecido,  $f_{exp}(c_{ij})$  é a função de decaimento exponencial negativa e  $\beta$  é um parâmetro que dita a velocidade de decaimento.

Inúmeras funções de decaimento podem ser utilizadas no cálculo de medidas gravitacionais. A medida de oportunidades cumulativas, por exemplo, pode ser pensada simplesmente como um caso especial de medida gravitacional em que o peso de cada oportunidade é ditado por uma função binária, em vez de decair gradualmente. Levinson e King (2020, p. 49) apresentam uma lista de funções de decaimento frequentemente utilizadas por agências de transportes e pesquisadores em análises que envolvem medidas gravitacionais.

Vantagens e desvantagens: a principal vantagem de medidas gravitacionais é que o desconto do peso das oportunidades pelo custo da viagem reflete em alguma medida a forma como as pessoas percebem o acesso a oportunidades, pois, de forma geral, serviços e atividades costumam ser mais atrativos quanto mais próximos eles estiverem. Este indicador, no entanto, tem ao menos duas desvantagens. A primeira delas é que os níveis de acessibilidade estimados são de difícil interpretação, pela forma como a contagem de oportunidades é descontada pelo custo de viagem. Além disso, para que esses níveis sejam mais representativos do comportamento de viagem das pessoas, o ritmo de decaimento da função de impedância (o parâmetro  $\beta$  da função exponencial negativa, por exemplo) precisa ser calibrado. Por isso, esta medida requer a disponibilidade de dados de comportamento de viagens ou de outros dados que possam ser usados no processo de calibração, disponíveis, por exemplo, a partir de pesquisas origem-destino, de coletas feitas por empresas de telefonia celular etc.

#### 2.1.4 Medidas de acessibilidade com competição: *floating catchment area*

Em muitos casos, o acesso a oportunidades é afetado não apenas por questões de proximidade geográfica e de custos de transporte, mas também pela competição de diferentes pessoas por uma mesma oportunidade. Isso é muito comum, por exemplo, em casos de acesso a serviços de saúde, escolas e empregos. Uma vaga de emprego só pode ser ocupada por uma pessoa de cada vez, e o mesmo vale para um leito de Unidade de Terapia Intensiva (UTI) ou uma matrícula escolar.

Existe um grande número de medidas que buscam incorporar essa competição na estimativa dos níveis de acessibilidade. Algumas das medidas de competição mais amplamente utilizadas são as do tipo *floating catchment area* (FCA), ou área de influência flutuante, em português. A título de exemplo, esses indicadores tentam levar em consideração como uma mesma pessoa pode potencialmente

acessar vários leitos de UTI e, simultaneamente, como cada leito de UTI pode potencialmente ser acessado por diversas pessoas. Assim, o acesso de uma pessoa a leitos de UTI é influenciado simultaneamente por questões de custos de transporte e pela disponibilidade de leitos, considerando a potencial concorrência na demanda por aqueles leitos.

Dentro da família de medidas de FCA, a mais comumente utilizada é a *2-step floating catchment area* (2SFCA), proposta originalmente por Luo e Wang (2003). Uma limitação da 2SFCA é que ela considera que uma mesma pessoa pode demandar várias oportunidades ao mesmo tempo e que, analogamente, um mesmo serviço pode ser utilizado por várias pessoas ao mesmo tempo. Esses fenômenos são conhecidos como problemas de inflação de demanda e de oferta, respectivamente, e podem gerar estimativas de acessibilidade enviesadas ou pouco precisas (Páez, Higgins e Vivona, 2019). Para lidar com esses problemas, Páez, Higgins e Vivona (2019) propuseram a *balanced floating catchment area* (BFCA), uma das medidas mais novas da família FCA.

Vantagens e desvantagens: diferentes medidas de FCA têm diferentes vantagens e desvantagens, em maior ou menor grau. No entanto, de maneira geral, a principal vantagem de medidas desta família é a sua capacidade de incorporar aspectos de competição em estimativas de acessibilidade. A principal desvantagem, em contrapartida, é a difícil interpretação e comunicação dos seus resultados.

## 2.2 Medidas baseadas em pessoas

Medidas de acessibilidade baseadas em pessoas são sensíveis não apenas à distribuição espacial de atividades e à configuração e ao desempenho da rede de transporte, mas também levam em consideração como as características pessoais de cada indivíduo (como sexo, idade, deficiência física etc.), e até questões como participação em atividades e compromissos pessoais, podem afetar sua facilidade de acesso a determinadas atividades. Esta categoria inclui, por exemplo, indicadores baseados em atividades (Dong *et al.*, 2006) e medidas de espaço-tempo (Neutens *et al.*, 2012).

Vantagens e desvantagens: embora indicadores desta categoria sejam mais sofisticados, eles costumam demandar grandes quantidades de dados, como registros de diários de viagem, pesquisas domiciliares tipo origem-destino etc. Por isso, o cálculo dessas medidas é computacionalmente mais intensivo, o que faz com que elas sejam menos utilizadas do que as medidas baseadas em lugares (Neutens *et al.*, 2010; Miller, 2018). Como medidas baseadas em pessoas tendem a não ser agregadas em um indicador sintético (exatamente por levarem em conta particularidades de cada indivíduo em seu cálculo, que seriam desconsideradas no cálculo de um valor médio), a comunicação de seus resultados também costuma ser mais complexa.





## SEÇÃO 2

### COMO CALCULAR ACESSIBILIDADE URBANA

O objetivo desta seção é mostrar como calcular estimativas de acessibilidade urbana em R usando os pacotes `{r5r}` e `{accessibility}`.

O cálculo dos níveis de acessibilidade em uma determinada área de estudo compreende duas etapas principais: primeiro, precisamos calcular uma matriz de custo de transporte, geralmente o tempo de viagem, entre as origens e os destinos que compõem a área de estudo. Feito isso, calculamos a acessibilidade nos pontos de origem, considerando os custos de transporte entre cada par de origem e destino e o número de oportunidades em cada destino. Nesta seção, aprenderemos como executar essas duas etapas usando a linguagem de programação R. Também aprenderemos sobre os dados que são necessários para executá-las e sobre as vantagens e desvantagens dos diferentes métodos que podem ser usados para isso.



## 3 CALCULANDO ESTIMATIVAS DE ACESSIBILIDADE EM R

### 3.1 Cálculo da matriz de tempo de viagem

A primeira etapa necessária para estimar os níveis de acessibilidade de uma área de estudo é calcular a matriz de custo de viagem entre as diversas origens e destinos que a compõem. Como comentado anteriormente, na literatura científica e na prática do planejamento de sistemas de transporte esse custo é mais frequentemente representado pelo tempo de viagem que separa dois pontos (El-Geneidy *et al.*, 2016; Venter, 2016), embora trabalhos recentes tenham considerado também outros fatores, como o dinheiro necessário para realizar uma viagem e o nível de conforto da viagem entre um ponto e outro (Arbex e Cunha, 2020; Herszenhut *et al.*, 2022). Neste livro, iremos focar em matrizes de tempo de viagem por serem as mais utilizadas na literatura e na prática, mas iremos cobrir outros tipos de custos num futuro livro sobre análise avançada de acessibilidade em R.

Atualmente, a forma mais fácil e rápida de gerar uma matriz de tempo de viagem em R é utilizando o pacote {r5r} (Pereira *et al.*, 2021), desenvolvido pela equipe do projeto AOP, do Ipea. O pacote utiliza, por trás dos panos, o *software* de roteamento de transporte multimodal R5, desenvolvido pela Conveyal.<sup>5</sup>

#### 3.1.1 Instalação do {r5r}

A instalação do {r5r} funciona como a instalação de qualquer pacote de R (todos os exemplos de código a seguir devem ser rodados em uma sessão de R).

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

Além do R, o pacote {r5r} requer também a instalação do Java 11.<sup>6</sup> Use o comando a seguir para checar a versão do Java instalada em seu computador.

```
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. Disponível em: <<https://github.com/conveyal/r5>>.

6. O Java 11 está disponível em: <<https://www.oracle.com/java/technologies/downloads/#java11>>; ou <<https://jdk.java.net/java-se-ri/11>>.

Como podemos ver, a versão instalada no livro é compatível com o `{r5r}`. Caso a versão instalada na sua máquina não o seja (resultado do código mencionando a versão 12 ou 1.8.0, por exemplo), será necessário atualizar o Java para a versão 11.

### 3.1.2 Dados necessários

O uso do pacote `{r5r}` requer diferentes tipos de dados. A lista a seguir descreve cada um deles, comenta sobre sua obrigatoriedade e apresenta algumas fontes onde esses dados podem ser obtidos.

- 1) Rede viária (obrigatório): um arquivo com a rede viária e de infraestrutura de pedestres do OpenStreetMap (OSM), em formato `.pbpf`. Pode ser baixado com: `{osmextract}` (pacote de R); [Geofabrik](#); [HOT Export Tool](#); ou [BBBike Extract Service](#).
- 2) Rede de transporte público (opcional): um ou mais arquivos no formato GTFS descrevendo a rede de transporte público da área de estudo. Caso ausente, deslocamentos por transporte público não são considerados no cálculo da matriz de tempo de viagem. Esses dados podem ser baixados com: `{tidytransit}` (pacote de R); ou [Transitland](#). No capítulo 4 deste livro (tabela 9), indicamos também onde baixar dados GTFS de algumas cidades brasileiras que compartilham seus dados publicamente.
- 3) Topografia (opcional): um arquivo de dados *raster* contendo o modelo digital de elevação da área de estudo em formato `.tif/.tiff`. Deve ser utilizado caso se deseje levar em consideração os efeitos da topografia do local sobre os tempos de caminhada e de viagens de bicicleta. Pode ser baixado com: `{elevatr}` (pacote de R); ou [SRTMGL1](#), da National Aeronautics and Space Administration (Nasa).

#### BOX 1

##### Qualidade dos dados do OSM

O OSM é uma base de dados geográfica que traz informações sobre malha viária, prédios, parques etc. Por ser uma base de dados alimentada voluntariamente pela comunidade que a utiliza, a cobertura e a qualidade dos dados do OSM podem variar muito entre regiões (Barrington-Leigh e Millard-Ball, 2017). Via de regra, os dados do OSM no Brasil e no mundo tendem a ter melhor cobertura e qualidade em regiões mais desenvolvidas e áreas urbanas com grandes populações (Barrington-Leigh e Millard-Ball, 2017; Camboim, Bravo e Sluter, 2015).

Elaboração dos autores.

Esses dados devem ser salvos em uma mesma pasta que, preferencialmente, não contenha nenhum outro arquivo. Como veremos adiante, o `{r5r}` combina todos os dados salvos nessa pasta para criar uma rede de transporte multimodal que é utilizada no roteamento de viagens entre pares origem-destino e, conseqüentemente, no cálculo das matrizes de tempo de viagem. Note que é possível ter mais de um

arquivo GTFS na mesma pasta: nesse caso, o `{r5r}` considera as redes de transporte público de todos os *feeds* de forma conjunta. No entanto, a rede viária e a topografia da área de estudo devem ser descritas por um único arquivo cada. Assumindo que os *scripts* de R estejam em uma pasta chamada R, uma possível organização dos arquivos segue o esquema a seguir:

```
/tmp/RtmpL7Wa9v/projeto_acessibilidade
+-- R
|   +-- script1.R
|   \-- script2.R
\-- r5
    +-- rede_transporte_publico.zip
    +-- rede_viaria.osm.pbf
    \-- topografia.tif
```

Para ilustrar as funcionalidades do `{r5r}`, vamos usar uma pequena amostra de dados da cidade de Porto Alegre (Brasil). Esses dados estão disponíveis dentro do próprio pacote `{r5r}`, e seu endereço pode ser acessado com o comando descrito adiante.

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

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

```
fs::dir_tree(pasta)
```

```
/home/runner/work/intro_access_book/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
```

Essa pasta possui cinco arquivos que vamos utilizar ao longo deste capítulo, conforme enumerado a seguir.

- 1) A rede viária do OSM: `poa_osm.pbf`.
- 2) Um *feed* de GTFS descrevendo algumas linhas da rede de ônibus da cidade: `poa_eptc.zip`.

- 3) Um *feed* de GTFS descrevendo algumas linhas da rede de trem da cidade: `poa_trensubr.zip`.
- 4) O modelo digital de elevação da cidade: `poa_elevation.tif`.
- 5) O arquivo `poa_hexgrid.csv`, com as coordenadas geográficas dos centroides de uma grade hexagonal regular que cobre toda a área de estudo e com informações sobre o tamanho da população residente e o número de empregos, hospitais e escolas em cada hexágono. Esses pontos serão utilizados como as origens e os destinos no cálculo da matriz de tempo de viagem.

### 3.1.3 Calculando a matriz de tempo de viagem

Antes de calcular a matriz de tempo de viagem, precisamos aumentar a memória disponível para a execução de processos do Java, linguagem em que o R5 é escrito. Isso é necessário porque, por padrão, o R aloca apenas 512 MB de memória para processos do Java, o que frequentemente não é suficiente para o cálculo de grandes matrizes com o {r5r}. Para aumentar a memória disponível para 2 GB, por exemplo, usamos o seguinte comando no início do *script*, antes mesmo de carregar as bibliotecas de R necessárias para a nossa análise:

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

Feito isso, podemos prosseguir com o cálculo da matriz de tempo de viagem, realizado em dois passos. O primeiro é gerar a rede de transporte multimodal que será utilizada no roteamento. Para isso, carregamos a biblioteca {r5r} e utilizamos a função `setup_r5()`, que baixa o *software* de roteamento R5 e o utiliza para criar a rede. Essa função recebe como *input* o caminho da pasta onde os dados necessários para o roteamento estão armazenados. Como resultado, a função salva na pasta alguns arquivos necessários para o roteamento e retorna uma conexão com o R5, que nesse exemplo foi armazenada na variável `conexao_r5r`. Essa conexão é responsável por garantir que o roteamento seja feito com a rede de transportes descrita pelos arquivos dentro da pasta e é utilizada no cálculo da matriz de tempo de viagem.

```
library(r5r)

conexao_r5r <- setup_r5(pasta, verbose = FALSE)

fs::dir_tree(pasta)
```

```

/home/runner/work/intro_access_book/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

```

O passo final consiste em calcular a matriz de tempo de viagem com a função `travel_time_matrix()`. Como *inputs* básicos, a função recebe a conexão com o R5 criada no passo anterior, pontos de origem e destino em formato `data.frame` com as colunas `id`, `lon` e `lat`, o modo de transporte a ser utilizado, o horário de partida, o tempo máximo de caminhada permitido da origem até o embarque no transporte público e do desembarque até o destino e o tempo máximo de viagem a ser considerado. Diversos outros *inputs* também podem ser usados, como a velocidade de caminhada e o número máximo de pernas de transporte público permitido, entre outros.<sup>7</sup>

```

pontos <- data.table::fread(file.path(pasta,
  "poa_hexgrid.csv"))

matriz <- travel_time_matrix(
  conexao_r5r,
  origins = pontos,
  destinations = pontos,
  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,

```

7. Para mais informações sobre cada um dos parâmetros, consulte a documentação da função em uma sessão de R (com os comandos `?travel_time_matrix()` ou `help("travel_time_matrix")`) ou no *site* do {r5r}, disponível em: <[https://ipeagit.github.io/r5r/reference/travel\\_time\\_matrix.html](https://ipeagit.github.io/r5r/reference/travel_time_matrix.html)>.



```

    verbose = FALSE,
    progress = FALSE
  )

  head(matriz)

```

	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

Na prática, a função `travel_time_matrix()` encontra a rota mais rápida partindo de cada ponto de origem para todos os possíveis pontos de destino, considerando o modo de viagem, o horário de partida e os demais parâmetros passados pelo usuário. Para isso, o `{r5r}` considera tempos de viagem de porta a porta: no caso de uma viagem por transporte público, por exemplo, o tempo total de viagem inclui: i) o tempo de caminhada até a parada de transporte público; ii) o tempo de espera pelo veículo na parada; iii) o tempo de deslocamento dentro do veículo; e iv) o tempo de viagem a pé da parada de desembarque até o destino. Em casos em que mais de uma rota de transporte público é utilizada, o `{r5r}` também contabiliza o tempo gasto nas conexões, considerando a caminhada entre paradas e o tempo de espera pelo próximo veículo.

## BOX 2

### Velocidade de roteamento com a `travel_time_matrix()`

A função `travel_time_matrix()` utiliza uma extensão do algoritmo de roteamento RAPTOR (Conway, Byrd e Van Der Linden, 2017), o que torna o R5 extremamente rápido. A depender da quantidade de pares origem-destino, o `{r5r}` pode ser entre seis e duzentas vezes mais rápido do que outros *softwares* de roteamento multimodal no cálculo de matrizes de tempo de viagem (Higgins *et al.*, 2022).

Elaboração dos autores.

## 3.2 Cálculo de acessibilidade

Calculada a matriz de tempo de viagem entre as origens e os destinos da área de estudo, precisamos utilizá-la para calcular os níveis de acessibilidade do local. Para isso, utilizaremos o pacote `{accessibility}`,<sup>8</sup> também desenvolvido pela equipe do projeto AOP/Ipea, que disponibiliza funções para o cálculo de vários indicadores de acessibilidade. Como *input* básico, todas as funções requerem uma matriz de custo pré-calculada (no nosso caso, a matriz de tempo de viagem calculada

8. Disponível em: <<https://github.com/ipeaGIT/accessibility>>.

na seção anterior) e dados de uso do solo, como o número de determinados tipos de oportunidades em cada célula da grade que cobre a área de estudo.

### 3.2.1 Medida de oportunidades cumulativas

A função `cumulative_cutoff()` é utilizada para calcular o número de oportunidades que podem ser alcançadas em um determinado limite de custo de viagem. No exemplo a seguir, primeiro carregamos a biblioteca `{accessibility}` e adequamos a nossa matriz para que ela possa ser utilizada no cálculo da acessibilidade. Em seguida, calculamos o número de escolas que podem ser alcançadas em até trinta minutos de viagem a partir de cada origem presente em nossa matriz de tempo de viagem.

```
library(accessibility)

# renomeia coluna para usar o pacote {accessibility}
data.table::setnames(matriz, "travel_time_p50", "travel_time")

oportunidades_cumulativas <- cumulative_cutoff(
  travel_matrix = matriz,
  land_use_data = pontos,
  opportunity = "schools",
  travel_cost = "travel_time",
  cutoff = 30
)

head(oportunidades_cumulativas)
```

	id	schools
1:	89a901291abfffff	23
2:	89a9012a3cfffff	0
3:	89a901295b7fffff	18
4:	89a901284a3fffff	4
5:	89a9012809bfffff	20
6:	89a901285cfffff	84

### 3.2.2 Custo mínimo de viagem

A função `cost_to_closest()`, por sua vez, calcula o custo mínimo de viagem necessário para alcançar um determinado número de oportunidades. Com o código a seguir, por exemplo, calculamos o tempo de viagem mínimo para alcançar a escola mais próxima a partir de cada origem.

```

custo_minimo <- cost_to_closest(
  travel_matrix = matriz,
  land_use_data = pontos,
  opportunity = "schools",
  travel_cost = "travel_time"
)

head(custo_minimo)

```

	id	travel_time
1:	89a9012124fffff	0
2:	89a9012126bffff	16
3:	89a9012127bffff	14
4:	89a90128003ffff	7
5:	89a90128007ffff	15
6:	89a9012800bffff	0

### 3.2.3 Medidas gravitacionais

A função `gravity()` calcula medidas gravitacionais de acessibilidade, aquelas nas quais o peso de cada oportunidade diminui gradualmente com o aumento do custo de viagem. Existe, no entanto, uma gama de diferentes tipos de funções de decaimento que podem ser utilizadas, como funções de decaimento exponenciais negativas, de potências inversas, entre outras. Por isso, essa função recebe um *input* adicional: a função de decaimento a ser utilizada no cálculo. O exemplo adiante apresenta o cálculo de acessibilidade a estabelecimentos de educação usando uma medida gravitacional exponencial negativa com parâmetro de decaimento igual a 0,2.

```

grav_exp_negativa <- gravity(
  travel_matrix = matriz,
  land_use_data = pontos,
  opportunity = "schools",
  travel_cost = "travel_time",
  decay_function = decay_exponential(0.2)
)

head(grav_exp_negativa)

```

	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 Medidas com competição

Por fim, a função `floating_catchment_area()` calcula níveis de acessibilidade levando em consideração a competição por oportunidades usando diferentes indicadores do tipo FCA. Como diversos métodos de FCA podem ser utilizados, a função requer que o método desejado seja explicitamente assinalado. Adicionalmente, assim como na função de acessibilidade gravitacional, a função de decaimento utilizada também deve ser definida pelo usuário. O código a seguir mostra um exemplo de cálculo de acessibilidade a escolas usando o método BFCA (Páez, Higgins e Vivona, 2019), levando em consideração os efeitos de competição entre a população como um todo e uma função de decaimento exponencial com parâmetro de decaimento igual a 0,05.

```

competicao_bfca <- floating_catchment_area(
  travel_matrix = matriz,
  land_use_data = pontos,
  opportunity = "schools",
  travel_cost = "travel_time",
  demand = "population",
  method = "bfca",
  decay_function = decay_exponential(0.05)
)

head(competicao_bfca)

```

```

      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

```

As funções apresentadas nesta seção também podem receber outros *inputs* não explicitamente mencionados aqui. Para mais informações sobre cada um dos parâmetros, pode-se consultar a documentação do pacote `{accessibility}` em seu *site*.

### 3.2.5 Cálculo de acessibilidade com o {r5r}

Nos último itens, mostramos como calcular níveis de acessibilidade passo a passo. Para fins didáticos, é importante entender que o cálculo de estimativas de acessibilidade tem como primeiro passo a geração de uma matriz de custos de viagens que, em seguida, é utilizada para estimar níveis de acessibilidade. No entanto, o {r5r} inclui também uma função chamada `accessibility()`, que calcula os níveis de acessibilidade com uma única chamada, sem etapas intermediárias.

De forma parecida com a função de cálculo de matriz de tempo de viagem, a função `accessibility()` recebe como *inputs* uma conexão com o R5, as origens, os destinos, os modos de transporte e o tempo de partida, entre outros argumentos. Adicionalmente, devem ser listadas também quais oportunidades devem ser consideradas e a função de decaimento que deve ser utilizada, bem como o valor do limite de custo e do parâmetro de decaimento. O exemplo a seguir mostra uma aplicação dessa função.

```
acessibilidade_r5r <- accessibility(
  conexao_r5r,
  origins = pontos,
  destinations = pontos,
  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(acessibilidade_r5r)
```

	id	opportunity	percentile	cutoff	accessibility
1:	89a901291abfffff	schools	50	30	21
2:	89a9012a3cfffff	schools	50	30	0
3:	89a901295b7fffff	schools	50	30	16
4:	89a901284a3fffff	schools	50	30	4
5:	89a9012809bfffff	schools	50	30	17
6:	89a901285cfffff	schools	50	30	78

Uma pequena diferença entre o comportamento da função `r5r::accessibility()` e o da `cumulative_cutoff()` do pacote `{accessibility}` está no fato de que, na função do `{r5r}`, pares origem-destino cujos tempos de viagem são iguais ao valor definido como limite são excluídos do cálculo da acessibilidade, enquanto na função do pacote `{accessibility}` eles são incluídos. Ou seja, para simularmos o cálculo anterior com a `cumulative_cutoff()`, precisamos estabelecer um tempo de viagem limite de 29 minutos, e não trinta. No código a seguir, comparamos os resultados das duas funções.

```
ops_cumulativas_29 <- cumulative_cutoff(
  travel_matrix = matriz,
  land_use_data = pontos,
  opportunity = "schools",
  travel_cost = "travel_time",
  cutoff = 29
)

# compara os níveis de acessibilidade calculados das duas
maneiras distintas
comparacao_acessibilidade <- merge(
  acessibilidade_r5r,
  ops_cumulativas_29,
  by = "id"
)

# renomeia colunas com níveis de acessibilidade
data.table::setnames(
  comparacao_acessibilidade,
  old = c("accessibility", "schools"),
  new = c("acesso_r5r", "acesso_acessibilidade")
)

head(comparacao_acessibilidade[, .(id, acesso_r5r, acesso_
  acessibilidade)])
```

	id	acesso_r5r	acesso_acessibilidade
1:	89a9012124fffff	1	1
2:	89a9012126bffff	12	12
3:	89a9012127bffff	14	14
4:	89a90128003ffff	30	30
5:	89a90128007ffff	21	21
6:	89a9012800bffff	29	29

Como podemos observar, fora a pequena diferença de comportamento, o resultado das duas funções é o mesmo. A principal diferença entre os dois métodos, no entanto, é que a informação “intermediária” do tempo de viagem entre origens e destinos não fica disponível ao usuário com o uso da função `accessibility()` do pacote `{r5r}`. Ainda assim, esse fluxo de trabalho pode ser uma boa alternativa para pessoas que estejam interessadas unicamente nos níveis de acessibilidade, não dependendo do tempo de viagem em suas análises. Note também que o pacote `{accessibility}` possui uma gama mais ampla de indicadores de acessibilidade e permite que os usuários definam funções de decaimento personalizadas.

### BOX 3

#### Considerando outros tipos de custo de viagem no cálculo da acessibilidade

Outra diferença entre a função de acessibilidade do `{r5r}` e as funções do pacote `{accessibility}` está no fato de que estas podem trabalhar com variados tipos de custo de viagem, como tempo, custo monetário, conforto etc. A função do `{r5r}`, porém, é menos flexível, limitando-se a considerar apenas restrições de tempo de viagem.

Elaboração dos autores.

## 3.3 Análises de acessibilidade

Calculados os níveis de acessibilidade, seguimos então para sua análise. Existe uma grande variedade de análises que podem ser feitas usando esses dados: por exemplo, diagnósticos das condições de acessibilidade urbana de diferentes bairros, pesquisas sobre desigualdades de acesso a oportunidades entre diferentes grupos sociais, análises sobre exclusão social e *accessibility poverty* (insuficiência de acessibilidade) etc. Nesta seção, no entanto, apresentaremos apenas duas análises relativamente simples e de fácil comunicação: a distribuição espacial da acessibilidade e sua distribuição entre diferentes grupos de renda.

### 3.3.1 Distribuição espacial de acessibilidade urbana

Para compreendermos a distribuição espacial da acessibilidade urbana de uma determinada cidade ou região, primeiro precisamos obter as informações espaciais dos pontos que foram utilizados como origens e destinos no cálculo da matriz. Os pontos que usamos nos exemplos anteriores correspondem aos centroides de células de uma grade hexagonal baseadas no índice [H3](#), desenvolvido pela Uber (Brodsky, 2018). A grade de Porto Alegre e algumas informações sociodemográficas e de uso do solo da cidade são disponibilizadas pela equipe do projeto AOP pelo pacote de R `{aopdata}`. O pacote e suas funções são apresentados em detalhes na seção 5. Com o código adiante, carregamos a biblioteca de visualização de dados, baixamos as informações espaciais da grade e as apresentamos em forma de mapa.

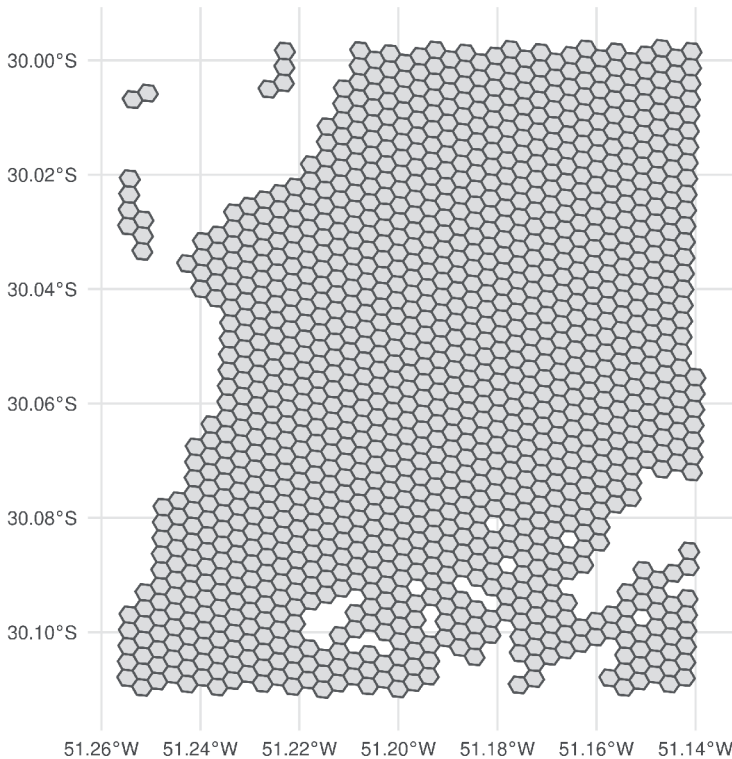
```
library(ggplot2)

# baixa a grade espacial
grade_poa <- aopdata::read_grid(city = "Porto Alegre")

# mantém na grade apenas os hexagonos utilizados na analise
grade_poa <- subset(grade_poa, id_hex %in% pontos$id)

# visualizando o mapa
ggplot(grade_poa) + geom_sf() + theme_minimal()
```

FIGURA 1

**Grade hexagonal cobrindo a região central de Porto Alegre**

Fonte: Figura gerada pelo código supracitado.

Para visualizarmos os dados de acessibilidade espacialmente, precisamos unir a tabela de estimativas de acessibilidade (considerando os níveis calculados com a medida de oportunidades cumulativas) com a tabela que contém os dados espaciais da grade, usando as colunas de identificação dos hexágonos como colunas-chave. Essa operação e seu resultado em formato de mapa são apresentados a seguir.

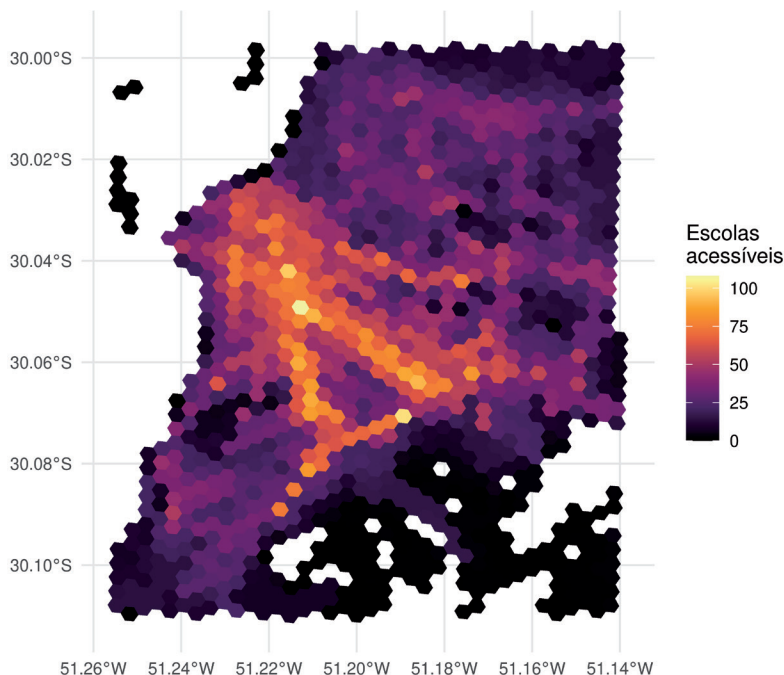


```
# junta as tabelas de dados espaciais e de níveis de acessibilidade
acesso_espacial <- merge(
  grade_poa,
  oportunidades_cumulativas,
  by.x = "id_hex",
  by.y = "id"
)

# configura mapa
ggplot(acesso_espacial) +
  geom_sf(aes(fill = schools), color = NA) +
  scale_fill_viridis_c(option = "inferno") +
  labs(fill = "Escolas\nacessíveis") +
  theme_minimal()
```

FIGURA 2

Distribuição espacial da acessibilidade a escolas na região central de Porto Alegre



Fonte: Figura gerada pelo código supracitado.

Como podemos ver, os níveis de acessibilidade tendem a se concentrar de forma mais acentuada no centro da cidade, onde existe maior concentração de empregos, e próximos aos grandes corredores de transporte da cidade. Por terem

fácil acesso a modos de alta capacidade e velocidade, pessoas que moram mais perto desses corredores tendem a acessar locais distantes de forma relativamente rápida. Em contraste, pessoas que moram mais afastadas desses corredores dependem de modos de menor frequência e velocidade operacional (como os ônibus municipais, por exemplo) e precisam gastar mais tempo para alcançar os corredores de média e alta capacidade. Como consequência, os níveis de acessibilidade de pessoas que moram afastadas do centro e de corredores de alta capacidade tendem a ser menores.

### 3.3.2 Distribuição socioeconômica de acessibilidade urbana

A figura 2, embora seja reveladora quanto aos locais em que estão dispostas as maiores concentrações de acessibilidade, nada mostra sobre os grupos socioeconômicos que possuem os maiores potenciais de acesso a oportunidades na região. Para isso, precisamos cruzar as informações demográficas e econômicas das pessoas que moram em cada um dos pontos de origem com os dados de acessibilidade previamente calculados.

No exemplo a seguir, juntamos aos dados de acessibilidade a informação do decil de renda de cada uma das origens, considerando a renda média de cada uma das pessoas que as habitam (dado também proveniente do pacote {aopdata}). Assim, conseguimos identificar se um hexágono é de baixa, média ou alta renda.

```
populacao_poa <- aopdata::read_population("Porto Alegre",
showProgress = FALSE)

# renomeia colunas com contagem populacional e decil de renda
data.table::setnames(
  populacao_poa,
  old = c("P001", "R003"),
  new = c("contagem_pop", "decil")
)

# junta as tabelas de níveis de acessibilidade espacializados
e dados sociodem.
acesso_sociodemografico <- merge(
  acesso_espacial,
  populacao_poa,
  by = "id_hex"
)

head(acesso_sociodemografico[, c("id_hex", "schools",
"contagem_pop", "decil")])
```

```
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
```

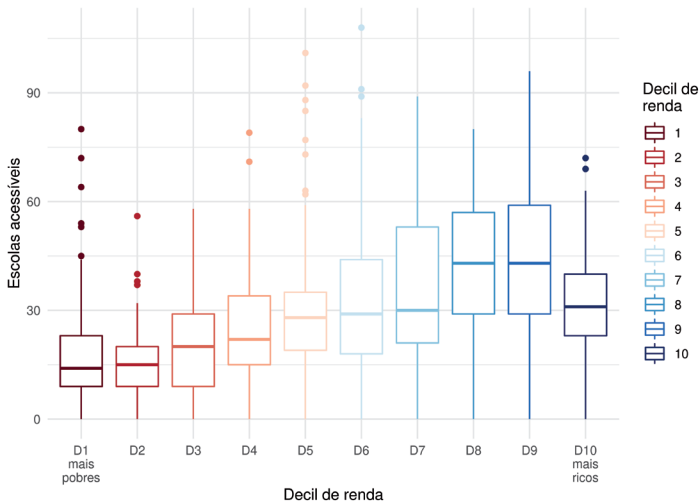
	id_hex	schools	contagem_pop	decil	geometry
1	89a9012124fffff	1	733	9	POLYGON ((-51.25083 -30.111...
2	89a9012126bffff	13	355	9	POLYGON ((-51.25369 -30.106...
3	89a9012127bffff	14	996	10	POLYGON ((-51.2538 -30.1094...
4	89a90128003ffff	34	1742	4	POLYGON ((-51.19446 -30.071...
5	89a90128007ffff	23	477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

Tendo a informação do decil de renda em que cada hexágono se encontra, podemos calcular a distribuição da acessibilidade da população dentro de cada um desses níveis de renda. Para isso, precisamos ponderar o nível de acessibilidade de cada origem pela quantidade de pessoas que residem ali – daí o porquê de termos também trazido a informação da contagem populacional em cada hexágono. Fazendo a ponderação, obtemos a distribuição da acessibilidade das pessoas localizadas em origens de um determinado decil de renda. Caso não ponderássemos, no entanto, teríamos a distribuição de acessibilidade não das pessoas localizadas em cada hexágono, mas dos hexágonos em si. Como em nossa análise nos importamos com as pessoas, e não com as unidades espaciais em que elas estão agregadas, precisamos fazer a ponderação. Podemos visualizar a distribuição de acessibilidade de cada decil usando um *box plot*, como mostrado a seguir.

```
ggplot(subset(acesso_sociodemografico, !is.na(decil))) +
  geom_boxplot(
    aes(
      x = as.factor(decil),
      y = schools,
      color = as.factor(decil),
      weight = contagem_pop
    )
  ) +
  labs(
    color = "Decil de\renda",
    x = "Decil de renda",
    y = "Escolas acessíveis"
  ) +
```

```
scale_color_brewer(palette = "RdBu") +
scale_x_discrete(
  labels = c("D1\nmais\npobres", paste0("D", 2:9),
    "D10\nmais\nricos")
) +
theme_minimal()
```

FIGURA 3  
Distribuição da acessibilidade a escolas na região central de Porto Alegre entre decis de renda



Fonte: Figura gerada pelo código supracitado.

A figura 3 é muito clara em seu conteúdo: pessoas de mais baixa renda tendem a ter níveis de acessibilidade consideravelmente menores do que as de alta renda. Esse é um padrão comum em praticamente todas as cidades brasileiras (Pereira *et al.*, 2020) que ocorre, em larga medida, devido à localização espacial das comunidades de baixa e alta renda no território: os mais ricos costumam morar em áreas mais valorizadas, próximas das grandes concentrações de empregos (e oportunidades de educação, saúde, lazer etc.) e com maior oferta de transporte público de média e alta capacidade. Os mais pobres, por sua vez, tendem a morar em locais mais afastados, onde o valor da terra é menor. Consequentemente, tendem também a se afastar das grandes concentrações de oportunidades. Junta-se a isso o fato de, na maior parte dos casos, a oferta de serviços de transporte público de média e alta capacidade ser pior em locais com maior concentração de pessoas de baixa renda. Sendo assim, seus níveis de acessibilidade são, em média, muito menores do que os dos mais ricos, como indicado pelos dados apresentados.



## DADOS DE TRANSPORTE PÚBLICO

Os objetivos desta seção são: i) apresentar o formato GTFS de dados de transporte público; ii) mostrar onde baixar dados GTFS de cidades brasileiras e do mundo; e iii) apresentar exemplos de como analisar e manipular dados de GTFS usando R.

Dados de transporte público são peças fundamentais no planejamento de transportes em geral e em análises de acessibilidade em particular. Para serem usados de forma que se tenha segurança no resultado das análises, esses dados precisam ser confiáveis e de simples inspeção e interpretação.

Tentando satisfazer esses critérios, cada vez mais agências de transporte público, tomadores de decisão e pesquisadores têm buscado utilizar dados estruturados conforme especificações abertas e colaborativas, ou seja, cujo formato seja definido por uma comunidade de atores interessados, incluindo partes que produzem esses dados (agências de transporte público, por exemplo) e que os consomem (pesquisadores, desenvolvedores de ferramentas de planejamento etc.). Embora uma especificação aberta não necessariamente resolva o problema da qualidade e da confiabilidade dos dados por ela descritos, seu uso traz várias vantagens que promovem o compartilhamento de conhecimento e a transparência de análises e aplicações que dependem dessa especificação – fatores que, por sua vez, podem levar a substanciais ganhos de qualidade e confiança nos dados.

O uso de um formato padrão de dados para transporte público permite o desenvolvimento e o compartilhamento de ferramentas e programas computacionais para análise desses dados, o que cria um campo comum de distribuição e aprendizado entre atores de diferentes cidades e países. Assim, um programa desenvolvido por uma agência de transporte no Brasil pode ser facilmente utilizado por um pesquisador nos Estados Unidos, por um desenvolvedor no Japão ou por outra agência de transporte na África do Sul – desde que, é claro, eles também organizem seus dados no mesmo formato. Além disso, quanto mais amplamente utilizado é esse formato, maior tende a ser tanto a confiabilidade na especificação em si quanto a facilidade de inspeção e interpretação dos seus dados, visto que múltiplos atores detêm o conhecimento necessário para isso.

A especificação de dados aberta e colaborativa mais amplamente utilizada no contexto do planejamento de transporte público é o formato GTFS, sigla para General Transit Feed Specification (Especificação Geral de Redes de Transporte Público, em tradução nossa). Seus usos abrangem tanto o planejamento quanto a operação de sistemas de transporte público. Como visto no capítulo 3, os dados de GTFS também são uma peça fundamental para calcular estimativas de acessibilidade urbana por transporte público. Nesta seção, iremos aprender em mais detalhes o que são os dados GTFS, como eles são estruturados e como trabalhar com esses dados em R.

## 4 DADOS GTFS

O formato GTFS é uma especificação aberta e colaborativa que visa descrever os principais componentes de uma rede de transporte público. Originalmente criada em meados dos anos 2000 por uma parceria entre Google e TriMet, a agência de transporte de Portland, em Oregon, nos Estados Unidos, a especificação hoje é utilizada por agências de transporte em milhares de cidades, espalhadas por todos os continentes do globo (McHugh, 2013). Atualmente, essa especificação é dividida em dois componentes distintos:

- GTFS Schedule, ou GTFS Static, que contém o cronograma planejado de linhas de transporte público, informações sobre suas tarifas e informações espaciais sobre os seus itinerários; e
- GTFS Realtime, que contém informações de localização de veículos em tempo real e alertas de possíveis atrasos, de mudanças de percurso e de eventos que possam interferir no cronograma planejado.

Ao longo desta seção, focaremos no formato GTFS Schedule, por ser o mais amplamente utilizado por agências de transporte e em análises de acessibilidade.<sup>9</sup>

Por ser uma especificação aberta e colaborativa, o formato GTFS tenta abarcar em sua definição um grande número de usos distintos que agências de transporte e desenvolvedores de ferramentas possam lhe dar. No entanto, agências e *softwares* podem ainda assim depender de informações que não constem na especificação oficial. Surgem, dessa forma, extensões da especificação, algumas das quais podem eventualmente se tornar parte da especificação oficial, caso isso seja aceito pela comunidade. Nesta seção, focaremos em um subconjunto de informações presentes no formato GTFS Schedule “puro” e, portanto, não cobriremos suas extensões.

### 4.1 Estrutura de arquivos GTFS

Arquivos no formato GTFS Schedule (daqui em diante chamado apenas de GTFS) também são conhecidos pela denominação *feed*.<sup>10</sup> Um *feed* é nada mais do que um arquivo comprimido em formato .zip que contém um conjunto de tabelas, salvas em formato .txt, com algumas informações sobre a rede de transporte público (localização das paradas, frequências das viagens, traçado das rotas, entre outras). Como em uma base de dados relacional, as tabelas de um *feed* possuem colunas-chaves que permitem vincular os dados de rotas, viagens e tabelas de

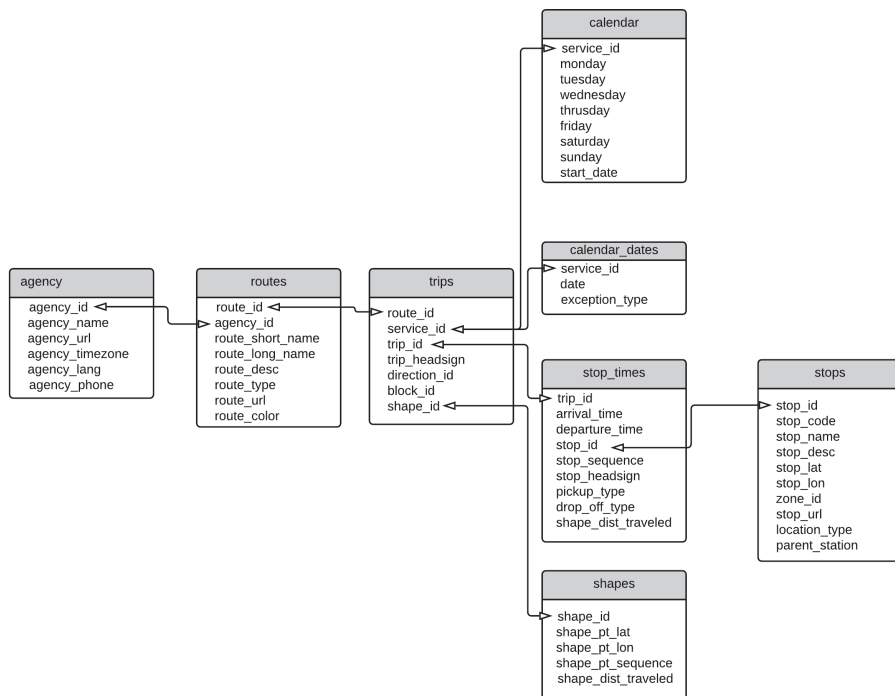
9. Mais informações sobre o formato GTFS Realtime estão disponíveis em: <<https://gtfs.org/realtime/>>.

10. Neste livro, utilizaremos os termos *feed*, arquivo GTFS e dados GTFS como sinônimos.



horários entre si. O esquema geral do formato GTFS é apresentado na figura 4, que mostra algumas das principais tabelas que compõem a especificação e destaca como elas se relacionam a partir de suas colunas-chaves.

FIGURA 4  
Esquema do formato GTFS



Fonte: Pereira, Andrade e Vieira (2022).

Ao todo, 22 tabelas compõem o formato GTFS.<sup>11</sup> Nem todas, no entanto, devem estar obrigatoriamente presentes para que um *feed* seja considerado válido, sendo consideradas, portanto, opcionais. A especificação classifica cada tabela conforme sua obrigatoriedade em três possíveis categorias: obrigatórias, opcionais e condicionalmente obrigatórias (quando a obrigatoriedade de uma tabela depende da existência de uma determinada tabela, coluna ou valor). Para fins de simplicidade, neste livro consideraremos apenas as duas primeiras categorias e faremos comentários quanto à obrigatoriedade de cada tabela quando apropriado. Dessa forma, as tabelas ficam classificadas conforme a seguir.

11. Conforme a [especificação oficial](#), versão da revisão de 9 de maio de 2022.

- 1) Obrigatórias: `agency.txt`; `stops.txt`; `routes.txt`; `trips.txt`; `stop_times.txt`; `calendar.txt`.
- 2) Opcionais: `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`.

Ao longo desta seção, aprenderemos sobre a estrutura básica de um arquivo GTFS e das tabelas que o compõem. Portanto, vamos olhar apenas para as tabelas obrigatórias e para as tabelas opcionais mais frequentemente utilizadas por produtores e consumidores desses arquivos.<sup>12</sup>

Na demonstração que será feita aqui, utilizaremos um subconjunto de dados provenientes do *feed* da cidade de São Paulo criado pela São Paulo Transporte (SPTrans)<sup>13</sup> e baixado em outubro de 2019. O *feed* contém as seis tabelas obrigatórias e mais duas tabelas opcionais bastante utilizadas, a `shapes.txt` e a `frequencies.txt`, o que permite uma boa visão geral sobre o formato GTFS.

#### 4.1.1 `agency.txt`

Arquivo utilizado para descrever as operadoras de transporte que atuam no sistema descrito pelo arquivo GTFS. Embora o termo *agency* (agência) seja usado em lugar de *operators* (operadoras), por exemplo, fica a cargo do produtor do *feed* definir quais instituições serão listadas na tabela.

Por exemplo: múltiplas concessionárias de ônibus atuam em um determinado local, mas todo o planejamento de cronograma e de tarifa é realizado por uma única instituição, em geral uma secretaria de transporte ou empresa pública específica. Essa instituição é também entendida pelos usuários do sistema como a operadora, de fato. Nesse caso, devemos listar a instituição responsável pelo planejamento na tabela.

Agora, imagine um sistema em que a agência de transporte público local transfere a responsabilidade da operação de um sistema multimodal a diversas empresas, por meio de concessões. Cada uma dessas empresas é responsável pelo planejamento de cronogramas e tarifas dos modos que operam, desde que sejam seguidos determinados parâmetros preestabelecidos em contrato. Sendo assim, devemos listar as operadoras (concessionárias) na tabela, e não a agência de transporte público em si.

---

12. Para mais informações sobre as tabelas e as colunas não abordadas neste texto, pode-se verificar a [especificação oficial](#).

13. Disponível em: <<https://www.sptrans.com.br/desenvolvedores/>>.

A tabela 1 mostra o arquivo `agency.txt` do *feed* da SPTrans. Como podemos ver, os responsáveis pelo *feed* optaram por listar a própria empresa no arquivo, e não as concessionárias que operam os ônibus e o metrô da cidade.

TABELA 1

**Exemplo de arquivo `agency.txt`**

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

Fonte: SPTrans.

É necessário notar que, embora estejamos apresentando o `agency.txt` em formato de tabela, o arquivo deve ser formatado como se fosse salvo em formato `.csv`. Ou seja, os valores de cada célula da tabela devem ser separados por vírgulas e cada linha da tabela deve constar em uma linha no arquivo. A tabela 1, por exemplo, é definida da seguinte forma:

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

Por uma questão de comunicação e interpretação dos dados, apresentaremos os exemplos em formato de tabela. É importante ter em mente, porém, que essas tabelas são organizadas como mostrado anteriormente.

4.1.2 `stops.txt`

Arquivo usado para descrever as paradas de transporte público que compõem o sistema. Os pontos listados neste arquivo podem fazer menção a paradas mais simples (como pontos de ônibus), estações, plataformas, entradas e saídas de estações etc. A tabela 2 mostra o `stops.txt` do *feed* da SPTrans.

TABELA 2

**Exemplo de arquivo `stops.txt`**

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

Fonte: SPTrans.

As colunas `stop_id` e `stop_name` servem como identificadores de cada parada, porém cumprem papéis distintos. O principal propósito da `stop_id` é identificar relações entre esta tabela e outras que compõem a especificação (como veremos mais à frente no arquivo `stop_times.txt`, por exemplo). Já a coluna `stop_name` cumpre o papel de um identificador facilmente reconhecido pelo passageiro. Seus valores, portanto, costumam ser nomes de estações, nomes de pontos de interesse da cidade ou endereços (como no caso do *feed* da SPTrans).

A coluna `stop_desc`, presente no *feed* da SPTrans, é opcional e permite à agência de transporte adicionar uma descrição de cada parada e de seu entorno. As colunas `stop_lat` e `stop_lon`, por fim, são as responsáveis por associar cada parada a uma posição espacial, por meio de suas coordenadas geográficas de latitude e longitude.

Entre as colunas opcionais não presentes no `stops.txt` deste *feed* estão a `location_type` e a `parent_station`. A `location_type` é utilizada para denotar o tipo de localização a que cada ponto se refere. Quando ausente, todos os pontos são interpretados como paradas de transporte público, mas valores distintos podem ser usados para distinguir uma parada (`location_type = 0`) de uma estação (`location_type = 1`) ou uma área de embarque (`location_type = 2`), por exemplo. A coluna `parent_station`, por sua vez, é utilizada para descrever relações de hierarquia entre dois pontos. Por exemplo, uma área de desembarque deve dizer a qual parada/plataforma ela pertence, assim como uma parada/plataforma pode também, opcionalmente, listar a qual estação ela pertence.

4.1.3 `routes.txt`

Arquivo usado para descrever as linhas de transporte público que rodam no sistema, incluindo os modos de transporte utilizados em cada uma. A tabela 3 mostra o `routes.txt` do *feed* da SPTrans.

TABELA 3  
Exemplo de arquivo `routes.txt`

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

Fonte: SPTrans.

Assim como no caso do arquivo `stops.txt`, a tabela do `routes.txt` também possui diferentes colunas que apontam o identificador de cada linha (`route_id`) e o seu nome. Nesse caso, no entanto, existem duas colunas de nome: a `route_short_name` e a `route_long_name`. A primeira diz respeito ao nome da linha, usualmente utilizado por passageiros no dia a dia, enquanto o segundo tende a ser um nome mais descritivo. A SPTrans, por exemplo, optou por destacar os pontos finais de cada linha nessa coluna. Podemos notar também que os mesmos valores se repetem nas colunas `route_id` e `route_short_name`, o que não é obrigatório nem proibido – nesse caso, o produtor do *feed* julgou que os nomes das linhas poderiam funcionar satisfatoriamente como identificadores por serem razoavelmente curtos e não se repetirem.

A coluna `agency_id` é a chave que permite relacionar a tabela das rotas com a tabela descrita no `agency.txt`. Ela faz menção a uma agência descrita naquele arquivo, a agência de id 1 (a própria SPTrans). Essa coluna é opcional no caso de *feeds* em que existe apenas uma agência, porém é obrigatória nos casos em que existe mais de uma. Imaginemos, por exemplo, um *feed* que descreve um sistema multimodal que conta com um corredor de metrô e diversas linhas de ônibus: uma configuração possível de `routes.txt` descreveria as linhas de metrô como de responsabilidade da operadora do metrô, e as de ônibus como de responsabilidade da empresa responsável pelo planejamento das linhas de ônibus, por exemplo.

A coluna `route_type` é utilizada para descrever o modo de transporte utilizado em cada linha. Essa coluna aceita diferentes números, cada um representando um determinado modo. Esse exemplo descreve linhas de trem, cujo valor numérico correspondente é 2. Os valores correspondentes para outros modos de transporte são listados na especificação.

#### 4.1.4 `trips.txt`

Arquivo usado para descrever as viagens realizadas no sistema. A viagem é a unidade básica de movimento do formato GTFS: cada viagem é associada a uma linha de transporte público (`route_id`), a um serviço que opera em determinados dias da semana (como veremos mais à frente, no arquivo `calendar.txt`) e a uma trajetória espacial (como será mostrado no arquivo `shapes.txt`). A tabela 4 mostra o `trips.txt` do *feed* da SPTrans.

TABELA 4  
Exemplo de arquivo `trips.txt`

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

Fonte: SPTrans.

A coluna `trip_id` identifica cada uma das viagens descritas na tabela, assim como a `route_id` faz referência a uma linha de transporte público identificada no arquivo `routes.txt`. A coluna `service_id` identifica serviços que determinam os dias da semana em que cada uma das viagens opera (dias úteis, finais de semana, uma mistura dos dois etc.), descritos detalhadamente no arquivo `calendar.txt`. A última coluna à direita na tabela 4 é a `shape_id`, que identifica a trajetória espacial de cada uma das viagens, descrita em detalhes no arquivo `shapes.txt`.

As duas colunas restantes, `trip_headsign` e `direction_id`, são opcionais e devem ser utilizadas para descrever o sentido/destino da viagem. A primeira, `trip_headsign`, é utilizada para ditar o texto que aparece no letreiro de veículos (no caso de um ônibus, por exemplo) ou em painéis informativos (como em metrô e trens) destacando o destino da viagem. Já a coluna `direction_id` é frequentemente utilizada em conjunto com a primeira para dar uma conotação de ida ou volta para cada viagem, em que 0 representa ida e 1, volta, ou vice-versa (assim como ida e volta são conceitos que mudam conforme o referencial, os valores 0 e 1 podem ser usados como desejado, desde que um represente um sentido e o outro, o contrário). No exemplo, as duas primeiras linhas são viagens que fazem menção à mesma rota de transporte público (CPTM L07), porém em sentidos opostos: uma corre em direção a Jundiaí e a outra, à estação da Luz.

4.1.5 `calendar.txt`

Arquivo usado para descrever os diferentes tipos de serviço existentes no sistema. Um serviço, nesse contexto, denota um conjunto de dias da semana em que viagens são realizadas. Cada serviço também é definido pela data em que começa a valer e pela data a partir da qual ele não é mais válido. A tabela 5 mostra o `calendar.txt` do *feed* da SPTrans.

TABELA 5

**Exemplo de arquivo `calendar.txt`**

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

Fonte: SPTTrans.

A coluna `service_id` identifica cada um dos serviços descritos na tabela. Como mostrado anteriormente, esse identificador é usado também no arquivo `trips.txt` e é o responsável por associar cada viagem a um determinado serviço.

As colunas `monday`, `tuesday`, `wednesday`, `thursday`, `friday`, `saturday` e `sunday` (segunda-feira a domingo, em inglês) são utilizadas para delimitar os dias em que cada serviço funciona. O valor 1 significa que o serviço opera naquele dia, enquanto o valor 0 significa que ele não opera. Como podemos ver na tabela 5, o serviço USD opera em todos os dias da semana. Já o serviço U\_\_ opera apenas em dias úteis.

Por fim, as colunas `start_date` e `end_date` delimitam o intervalo em que cada serviço é válido. As datas do formato GTFS são sempre formatadas segundo a regra YYYYMMDD, em que os primeiros quatro números definem o ano, os dois subsequentes definem o mês e os últimos dois, o dia. O valor 20220428, por exemplo, representa o dia 28 de abril de 2022.

4.1.6 `shapes.txt`

Arquivo usado para descrever a trajetória espacial de cada viagem listada no *feed*. Esse arquivo é opcional, mas é fortemente recomendado que agências de transporte o incluam em seus arquivos GTFS. A tabela 6 mostra o `shapes.txt` do *feed* da SPTTrans.

TABELA 6

**Exemplo de arquivo `shapes.txt`**

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

Fonte: SPTTrans.

A coluna `shape_id` identifica cada uma das trajetórias descritas na tabela. Como mostrado anteriormente, esse identificador é usado também no arquivo `trips.txt` e é o responsável por associar cada viagem à sua trajetória espacial. Diferentemente de todos os outros identificadores que vimos até então, no entanto, o identificador `shape_id` se repete em diversas observações da tabela. Isso porque o arquivo associa cada `shape_id` a uma série de pontos espaciais, cujas coordenadas geográficas são descritas nas colunas `shape_pt_lat` e `shape_pt_lon`. A coluna `shape_pt_sequence` lista a sequência na qual os pontos se conectam para formar a trajetória de cada `shape_id`. Os valores listados nessa coluna devem ser ordenados de forma crescente.

4.1.7 `stop_times.txt`

Arquivo usado para descrever a tabela de horários de cada viagem, incluindo o horário de chegada e partida em cada uma das paradas. A formatação desse arquivo depende da existência ou não de um arquivo `frequencies.txt`, detalhe que cobriremos mais adiante. Por enquanto, olharemos para o `stop_times.txt` do *feed* da SPTrans, que também conta com um `frequencies.txt`, na tabela 7.

TABELA 7  
Exemplo de arquivo `stop_times.txt`

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

Fonte: SPTrans.

A viagem cuja tabela de horários está sendo descrita é identificada pela coluna `trip_id`. De forma análoga ao que acontece na tabela de trajetórias, um mesmo `trip_id` se repete em muitas observações da tabela. Isso porque, assim como a trajetória é composta por uma sequência de pontos espaciais, a tabela de horários é composta por uma sequência de diversos horários de partida/chegada em diversas paradas de transporte público.

As colunas seguintes, `arrival_time`, `departure_time` e `stop_id`, são as responsáveis por descrever o cronograma de cada viagem, associando um horário de chegada e um horário de partida a cada uma das paradas da viagem. As colunas de horário são formatadas segundo a regra HH:MM:SS, em que os dois primeiros números definem a hora, os dois seguintes, os minutos e os últimos dois, os



segundos. Essa formatação aceita valores de hora maiores que 24: por exemplo, se uma viagem parte às 23h, mas só chega a uma determinada estação 1h da manhã do dia seguinte, seu horário de chegada deve ser registrado como 25:00:00, e não 01:00:00. A coluna `stop_id`, por sua vez, associa os horários de chegada e partida a uma parada descrita no arquivo `stops.txt`. Por fim, a coluna `stop_sequence` lista a sequência na qual cada parada se conecta às demais para formar o cronograma da viagem. Seus valores devem ser sempre ordenados de forma crescente.

Vale destacar aqui a diferença entre os arquivos `shapes.txt` e `stop_times.txt`. Embora os dois descrevam uma viagem espacialmente, eles o fazem de forma diferente. O `stop_times.txt` descreve a sequência de paradas e horários que compõem um cronograma, mas nada diz sobre o trajeto percorrido pelo veículo entre cada uma das paradas. Já o `shapes.txt` traz a trajetória detalhada da viagem como um todo, mas não descreve em que ponto do espaço estão as paradas da viagem. Quando usamos os dois arquivos em conjunto, portanto, sabemos tanto o cronograma de cada viagem quanto a trajetória espacial da viagem entre paradas.

#### 4.1.8 `frequencies.txt`

Arquivo opcional usado para descrever a frequência de cada viagem dentro de um determinado período do dia. A tabela 8 mostra o `frequencies.txt` do *feed* da SPTrans.

TABELA 8  
Exemplo de arquivo `frequencies.txt`

<code>trip_id</code>	<code>start_time</code>	<code>end_time</code>	<code>headway_secs</code>
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

Fonte: SPTrans.

A viagem cuja frequência está sendo descrita é identificada pela coluna `trip_id`. Novamente, um mesmo identificador pode aparecer em várias observações da tabela, pois a especificação prevê que uma mesma viagem pode ter frequências diferentes ao longo do dia (como em horários de pico e fora do pico, por exemplo). Assim, cada linha da tabela se refere à frequência de uma determinada viagem dentro de um intervalo de tempo especificado pelas colunas `start_time` e `end_time`.

Dentro do período especificado por essas duas colunas, a viagem possui um *headway* detalhado na coluna *headway\_secs*. O *headway* é o tempo que separa a passagem de dois veículos que operam a mesma linha de transporte público. No caso da tabela *frequencies.txt*, esse tempo deve ser especificado em segundos. Um valor de 720 entre 4h e 5h, portanto, significa que a viagem CPTM L07-0 ocorre de doze em doze minutos dentro desse período.

*Usando as tabelas frequencies.txt e stop\_times.txt conjuntamente*

É importante entender, agora, como a presença da tabela *frequencies.txt* altera a especificação da tabela *stop\_times.txt*. Como podemos ver no exemplo da tabela *stop\_times.txt* (tabela 7), a viagem CPTM L07-0 parte da primeira parada às 4h e chega na segunda às 4h08. O cronograma de chegada e saída de uma mesma parada de uma viagem, no entanto, não pode ser definido mais de uma vez na tabela. Como então definir o cronograma das viagens que partem às 4h12, 4h24, 4h36 etc. (lembrando que o *headway* dessa viagem é de doze minutos)?

No caso em que a frequência de uma viagem é especificada no *frequencies.txt*, o cronograma (a tabela de horários) de uma viagem definido no *stop\_times.txt* deve ser entendido como uma referência que descreve o tempo entre paradas. Isto é, os horários ali definidos não devem ser interpretados à risca. Por exemplo, o cronograma listado estabelece que o tempo de viagem entre a primeira e a segunda parada é de oito minutos, e o tempo entre a segunda e a terceira também. Ou seja, a viagem que parte da primeira parada às 4h chega na segunda às 4h08, e, na terceira, às 4h16. A próxima viagem, que parte da primeira parada às 4h12, por sua vez, chega na segunda parada às 4h20, e, na terceira, às 4h28.

Entretanto, poderíamos descrever as mesmas viagens no *stop\_times.txt* sem fazer uso do arquivo *frequencies.txt*. Para isso, poderíamos adicionar um sufixo que identificasse cada uma das viagens referentes à linha CPTM L07 no sentido 0 ao longo do dia. A viagem (*trip\_id*) com identificador CPTM L07-0\_1, por exemplo, seria a primeira viagem no sentido 0 do dia e partiria da primeira parada às 4h e chegaria na segunda às 4h08. A viagem CPTM L07-0\_2, por sua vez, seria a segunda viagem e partiria da primeira parada às 4h12 e chegaria na segunda às 4h20, e assim por diante. Cada uma dessas viagens deveria ser também adicionada ao arquivo *trips.txt* e a quaisquer outros que possuam a coluna *trip\_id* como identificador.

Outro elemento que influencia na forma como o *frequencies.txt* afeta as tabelas de horários na tabela *stop\_times.txt* é a coluna opcional *exact\_times*. Um valor de 0 nesta coluna (ou quando ela está ausente do *feed*, como no caso do arquivo GTFS da SPTrans) indica que a viagem não necessariamente segue um cronograma fixo ao longo do período. Em vez disso, operadores tentam se ater a um determinado *headway* durante o período. Usando o mesmo exemplo de

uma viagem cujo *headway* é de doze minutos entre 4h e 5h, isso significa que não necessariamente a primeira partida sairá exatamente às 4h, a segunda às 4h12 e por aí em diante. A primeira pode, por exemplo, sair às 4h02. A segunda, às 4h14 ou 4h13 etc. Caso desejemos definir um cronograma que é seguido à risca, obtendo o mesmo resultado que seria obtido se definíssemos diversas viagens semelhantes partindo em diferentes horários no `stop_times.txt` (como mostrado no parágrafo anterior), devemos utilizar o valor 1 na coluna `exact_times`.

## 4.2 Onde encontrar dados GTFS de cidades brasileiras

Os dados de GTFS de diversas cidades do mundo podem ser baixados com o pacote de R `{tidytransit}` ou no *site* [Transitland](https://transitland.com/). No Brasil, diversas cidades usam dados GTFS no planejamento e operação de seus sistemas de transportes. Em muitos casos, no entanto, esses dados são de propriedade de empresas operadoras e concessionárias, e não do poder público. Infelizmente, esses arquivos raramente são disponibilizados abertamente e publicamente, contrariando boas práticas de gestão e compartilhamento de dados de interesse público. A tabela 9 mostra as fontes dos dados GTFS de algumas das poucas cidades do Brasil que disponibilizam seus *feeds* abertamente.<sup>14</sup>

TABELA 9

### Fontes de dados GTFS publicamente disponíveis no Brasil

Cidade	Fonte	Informações
Belo Horizonte	Empresa de Transportes e Trânsito de Belo Horizonte (BHTrans).	Dado aberto: <a href="#">transporte convencional</a> e <a href="#">transporte suplementar</a> .
Fortaleza	Empresa de Transporte Urbano de Fortaleza (Etufor).	Dado aberto, disponível em: <a href="https://dados.fortaleza.ce.gov.br/dataset/gtfs">https://dados.fortaleza.ce.gov.br/dataset/gtfs</a> .
Fortaleza	Metrô de Fortaleza (Metrofor).	Dado aberto, disponível em: <a href="https://www.metrofor.ce.gov.br/gtfs/">https://www.metrofor.ce.gov.br/gtfs/</a> .
Porto Alegre	Empresa Pública de Transporte e Circulação de Porto Alegre (EPTC).	Dado aberto, disponível em: <a href="https://dados.portoalegre.rs.gov.br/dataset/gtfs">https://dados.portoalegre.rs.gov.br/dataset/gtfs</a> .
Rio de Janeiro	Secretaria Municipal de Transportes (SMTR).	Dado aberto, disponível em: <a href="https://www.data.rio/datasets/gtfs-do-rio-de-janeiro/about">https://www.data.rio/datasets/gtfs-do-rio-de-janeiro/about</a> .
São Paulo	Empresa Metropolitana de Transportes Urbanos de São Paulo (EMTU).	<i>Download</i> disponível em: <a href="https://www.emtu.sp.gov.br/emtu/dados-abertos/dados-abertos-principal/gtfs.fss">https://www.emtu.sp.gov.br/emtu/dados-abertos/dados-abertos-principal/gtfs.fss</a> . Necessário cadastro.
São Paulo	SPTans.	<i>Download</i> disponível em: <a href="https://www.sptrans.com.br/desenvolvedores/perfil-desenvolvedor/">https://www.sptrans.com.br/desenvolvedores/perfil-desenvolvedor/</a> . Necessário cadastro.

Elaboração dos autores.

Obs.: Os dados de GTFS disponibilizados pela SMTR não incluem os dados dos sistemas de trem e de metrô.

14. Levantamento realizado durante a elaboração deste livro.

## 5 MANIPULAÇÃO E VISUALIZAÇÃO DE DADOS GTFS

Usualmente, arquivos GTFS provenientes de fontes oficiais são utilizados para desenvolver análises e pesquisas que possuem diversos elementos comuns. Visando facilitar a leitura, o processamento e a análise desses dados, a equipe do projeto AOP vem desenvolvendo o pacote de R `{gtfstools}`, que disponibiliza diversas funções que facilitam a manipulação e a exploração de *feeds*.

Neste capítulo, passaremos por algumas das funcionalidades mais frequentemente utilizadas do pacote. Para isso, vamos utilizar uma amostra do *feed* da SPTrans apresentado no capítulo anterior, disponível dentro do `{gtfstools}`.

### 5.1 Leitura e manipulação básica de arquivos GTFS

A leitura de arquivos GTFS com o `{gtfstools}` é feita com a função `read_gtfs()`, que recebe uma *string* com o caminho do arquivo. Após sua leitura, o *feed* é representado como uma lista de `data.tables`, uma versão de alta *performance* da classe `data.frame`. Ao longo deste capítulo, vamos nos referir a essa lista de tabelas como um *objeto GTFS*. Por padrão, a função a seguir lê todas as tabelas `.txt` do *feed*:

```
# carrega biblioteca
library(gtfstools)

# aponta para o endereço do arquivo gtfs dentro do {gtfstools}
endereco <- system.file("extdata/spo_gtfs.zip", package =
"gtfstools")

# le o gtfs
gtfs <- read_gtfs(endereco)

# consulta o nome das tabelas dentro da lista
names(gtfs)
```

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

Como podemos ver, cada `data.table` dentro do objeto GTFS é nomeado de acordo com a tabela que ele representa, sem a extensão `.txt`. Isso nos permite selecionar e manipular cada uma das tabelas separadamente. O código adiante, por exemplo, mostra os seis primeiros registros da tabela `trips`:

```
head(gtfs$strips)
```

	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

As tabelas dentro de um objeto GTFS podem ser facilmente manipuladas usando a sintaxe dos pacotes `{dplyr}` ou `{data.table}`. Neste livro, optamos por utilizar a sintaxe do `{data.table}`, pois esse pacote oferece diversas funcionalidades para a manipulação de tabelas com grande quantidade de registros, tal como a edição de colunas por referência, filtros de linhas muito rápidos e agregação de dados eficiente.<sup>15</sup> Para adicionar cem segundos a todos os *headways* listados na tabela `frequencies` e reverter essa mudança em seguida, por exemplo, podemos usar o código a seguir:

```
# salva o headway original
headway_original <- 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

# modifica o headway
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

# restitui o headway original
gtfs$frequencies[, headway_secs := headway_original]
head(gtfs$frequencies, 3)
```

15. Mais detalhes sobre o uso e a sintaxe do `{data.table}` estão disponíveis em: <https://rdatatable.gitlab.io/data.table/index.html>.

	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

Após editarmos um objeto GTFS no R, frequentemente vamos querer utilizá-lo para fazer análises de diferentes tipos. Para isso, é comum que precisemos do arquivo GTFS em formato .zip novamente, e não como uma lista de tabelas dentro do R. O pacote {gtfstools} disponibiliza a função `write_gtfs()` exatamente com a finalidade de transformar objetos GTFS que existem apenas dentro do R em arquivos GTFS armazenados no seu computador. Para usarmos essa função, precisamos apenas listar o objeto e o endereço no qual o arquivo deve ser salvo:

```
# aponta para o endereço onde arquivo deve ser salvo
endereco_destino <- tempfile("novo_gtfs", fileext = ".zip")

# salva o GTFS no endereco
write_gtfs(gtfs, path = endereco_destino)

# lista arquivos dentro do feed recém-salvo
zip::zip_list(endereco_destino)[, c("filename", "compressed_
size", "timestamp")]
```

	filename	compressed_size	timestamp
1	agency.txt	112	2023-06-13 21:17:54
2	calendar.txt	129	2023-06-13 21:17:54
3	frequencies.txt	2381	2023-06-13 21:17:54
4	routes.txt	659	2023-06-13 21:17:54
5	shapes.txt	160470	2023-06-13 21:17:54
6	stop_times.txt	7907	2023-06-13 21:17:54
7	stops.txt	18797	2023-06-13 21:17:54
8	trips.txt	717	2023-06-13 21:17:54

## 5.2 Cálculo de velocidade das linhas

Arquivos GTFS são frequentemente utilizados em estimativas de roteamento de transporte público e para informar passageiros sobre a tabela de horários das diferentes rotas que operam em uma região. Dessa forma, é extremamente importante que o cronograma das viagens e a velocidade operacional de cada linha estejam adequadamente descritos no *feed*.

O {gtfstools} disponibiliza a função `get_trip_speed()` para facilitar o cálculo da velocidade de cada viagem presente no *feed*. Por padrão, a função calcula a velocidade (em km/h) de todas as viagens do objeto GTFS, mas viagens individuais também podem ser especificadas:

```
# calcula a velocidade de todas as viagens
velocidades <- get_trip_speed(gtfs)

head(velocidades)
```

	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(velocidades)
```

```
[1] 36
```

```
# calcula a velocidade de duas viagens específicas
velocidades <- get_trip_speed(gtfs, trip_id = c("CPTM L07-0",
"2002-10-0"))

velocidades
```

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

Calcular a velocidade de uma viagem requer que saibamos o seu comprimento e em quanto tempo ela foi realizada. Para isso, a `get_trip_speed()` utiliza duas outras funções do {gtfstools} por trás dos panos: a `get_trip_length()` e a `get_trip_duration()`. O funcionamento das duas é muito parecido com o mostrado anteriormente, calculando o comprimento/duração de todas as viagens por padrão, ou de apenas algumas selecionadas, caso desejado. A seguir, mostramos seus comportamentos padrões.

```
# calcula a distância percorrida de todas viagens
distancias <- get_trip_length(gtfs, file = "shapes")
```

```
head(distancias)
```

```
      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
```

```
# calcula a duração de todas viagens
duracao <- get_trip_duration(gtfs)
```

```
head(duracao)
```

```
      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
```

Assim como a `get_trip_speed()` calcula as velocidades em km/h por padrão, a `get_trip_length()` e a `get_trip_duration()` calculam os comprimentos e as durações em quilômetros e em minutos, respectivamente. Essas unidades podem ser ajustadas com o argumento `unit`, presente nas três funções.

### 5.3 Combinando e filtrando *feeds*

Muitas vezes, o processamento e a edição de arquivos GTFS são realizados, em grande medida, manualmente. Por isso, pequenas inconsistências podem passar batidas pelos responsáveis por esse processamento. Um problema comumente observado em *feeds* é a presença de registros duplicados em uma mesma tabela. O *feed* da SPTrans, por exemplo, possui registros duplicados tanto no `agency.txt` quanto no `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

```

O `{gtfstools}` disponibiliza a função `remove_duplicates()` para remover essas duplicatas. Essa função recebe como *input* um objeto GTFS e retorna o mesmo objeto, porém sem registros duplicados:

```

# remove valores duplicados
gtfs_sem_dups <- remove_duplicates(gtfs)

gtfs_sem_dups$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

```

```
gtfs_sem_dups$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

```

Frequentemente, também, lidamos com múltiplos *feeds* em uma mesma área de estudo. Por exemplo, quando os dados dos sistemas de ônibus e de trens de uma mesma cidade estão salvos em arquivos GTFS separados. Nesse caso, muitas vezes gostaríamos de uni-los em um único arquivo, diminuindo assim o esforço de manipulação e processamento dos dados. Para isso, o {gtfstools} disponibiliza a função `merge_gtfs()`. O exemplo a seguir mostra o resultado da combinação de dois *feeds* distintos, o da SPTrans (sem duplicatas) e o da EPTC, de Porto Alegre:

```

# lê GTFS de Porto Alegre
endereco_poa <- system.file("extdata/poa_gtfs.zip", package =
"gtfstools")
gtfs_poa <- read_gtfs(endereco_poa)

gtfs_poa$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

```

```
gtfs_sem_dups$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

```

```

# combina objetos GTFS de Porto Alegre e São Paulo
gtfs_combinado <- merge_gtfs(gtfs_sem_dups, gtfs_poa)

# checa resultados
gtfs_combinado$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

```

Como podemos ver, os registros das tabelas de ambos os *feeds* foram combinados em uma única tabela. Esse é o caso quando os dois (ou mais, se desejado) objetos GTFS possuem registros de uma mesma tabela (a *agency*, no exemplo). Caso apenas um dos objetos possua uma das tabelas, a operação copia essa tabela para o resultado final. É o caso, por exemplo, da tabela *frequencies*, que existe no *feed* da SPTrans, mas não no da EPTC:

```
names(gtfs_poa)
```

```

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

```

```
names(gtfs_sem_dups)

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

names(gtfs_combinado)

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

identical(gtfs_sem_dups$frequencies, gtfs_combinado$frequencies)

[1] TRUE
```

Um outro tipo de operação muito utilizada no tratamento de dados GTFS é o de filtragem desses arquivos. Frequentemente, *feeds* são usados para descrever redes de transporte público de grande escala, o que pode transformar sua edição, sua análise e seu compartilhamento em operações complexas. Por esse motivo, pesquisadores e planejadores muitas vezes precisam trabalhar com um subconjunto de dados descritos nos *feeds*. Por exemplo, caso desejemos estimar a *performance* da rede de transporte em uma determinada região no horário de pico da manhã, podemos filtrar o nosso arquivo GTFS de modo a manter apenas os registros referentes a viagens que ocorrem nesse intervalo do dia.

O pacote `{gtfstools}` traz diversas funções para facilitar a filtragem de arquivos GTFS. São elas:

- `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();` e
- `filter_by_sf().`

### 5.3.1 Filtro por identificadores

As sete primeiras funções mencionadas anteriormente são utilizadas de forma muito similar. Devemos especificar um vetor de identificadores que é usado para manter no objeto GTFS apenas os registros relacionados a esses identificadores. O exemplo a seguir demonstra essa funcionalidade com a `filter_by_trip_id()`:

```
# checa tamanho do feed antes do filtro
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

```
# mantém apenas registros relacionados a duas viagens
gtfs_filtrado <- filter_by_trip_id(
  gtfs,
  trip_id = c("CPTM L07-0", "CPTM L07-1")
)
```

```
# checa tamanho do feed após o filtro
utils::object.size(gtfs_filtrado)
```

71592 bytes

```
head(gtfs_filtrado$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

```
unique(gtfs_filtrado$shapes$shape_id)
```

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

O código mostra que a função não filtra apenas a tabela `trips`, mas também as outras tabelas que possuem algum tipo de relação com os identificadores especificados. Por exemplo, a trajetória das viagens CPTM L07-0 e CPTM L07-1 é descrita pelos `shape_ids` 17846 e 17847, respectivamente. Esses são, portanto, os únicos identificadores da tabela `shapes` mantidos no objeto GTFS filtrado.

A função também pode funcionar com o comportamento diametralmente oposto: em vez de definirmos os identificadores cujos registros devem ser *mantidos* no *feed*, podemos especificar os identificadores que devem ser *retirados* dele. Para isso, usamos o argumento `keep` com valor `FALSE`:

```
# remove duas viagens do feed
gtfs_filtrado <- filter_by_trip_id(
  gtfs,
  trip_id = c("CPTM L07-0", "CPTM L07-1"),
  keep = FALSE
)

head(gtfs_filtrado$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

```
head(unique(gtfs_filtrado$shapes$shape_id))
```

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

Como podemos ver, as viagens especificadas, bem como suas trajetórias, não estão presentes no objeto GTFS filtrado. A mesma lógica aqui demonstrada com a `filter_by_trip_id()` é válida para as funções que filtram objetos GTFS pelos identificadores `agency_id`, `route_id`, `service_id`, `shape_id`, `stop_id` e `route_type`.

### 5.3.2 Filtro por dia e hora

Outra operação que recorrentemente aparece em análises que envolvem dados GTFS é a de manter serviços que funcionem apenas em determinados horários do dia ou dias da semana. Para isso, o pacote disponibiliza as funções `filter_by_weekday()` e `filter_by_time_of_day()`.

A `filter_by_weekday()` recebe os dias da semana (em inglês) cujos serviços que neles operam devem ser mantidos. Adicionalmente, a função também inclui o argumento `combine`, que define como filtros de dois ou mais dias funcionam. Quando este recebe o valor "and", apenas serviços que operam em todos os dias especificados são mantidos. Quando recebe o valor "or", serviços que operam em pelo menos um dos dias são mantidos:

```
# mantém apenas serviços que operam no sábado E no domingo
gtfs_filtrado <- filter_by_weekday(
  gtfs = gtfs_sem_dups,
  weekday = c("saturday", "sunday"),
  combine = "and"
)

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

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

```
# mantém apenas serviços que operam OU no sábado OU no domingo
gtfs_filtrado <- filter_by_weekday(
  gtfs = gtfs_sem_dups,
  weekday = c("sunday", "saturday"),
  combine = "or"
)

gtfs_filtrado$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

A `filter_by_time_of_day()`, por sua vez, recebe o começo e o final de uma janela de tempo e mantém os registros relacionados a viagens que rodam dentro dessa janela. O funcionamento da função depende da presença ou não da tabela `frequencies` no objeto GTFS: o cronograma descrito na `stop_times` das viagens listadas na tabela `frequencies` não deve ser filtrado, pois, como comentado no capítulo anterior, ele serve como um modelo que dita o tempo de viagem entre uma parada e outra. Caso a `frequencies` esteja ausente, no entanto, a `stop_times` é filtrada segundo o intervalo de tempo especificado. Vamos ver como isso funciona com um exemplo:

```
# mantém apenas viagens dentro do período de 5 às 6 da manhã
gtfs_filtrado <- filter_by_time_of_day(gtfs, from = "05:00:00",
to = "06:00:00")
```

```
head(gtfs_filtrado$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(gtfs_filtrado$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

```
# salva a tabela frequencies e a remove do objeto gtfs
frequencies <- gtfs$frequencies
gtfs$frequencies <- NULL
```

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



```
head(gtfs_filtrado$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

O filtro da tabela `stop_times` pode funcionar de duas formas distintas: mantendo intactas todas as *viagens* que *cruzam* a janela de tempo especificada; ou mantendo no cronograma apenas as *paradas* que são visitadas *dentro* da janela (comportamento padrão da função). Esse comportamento é controlado com o parâmetro `full_trips`, como mostrado a seguir (atenção aos horários e aos segmentos presentes em cada exemplo):

```
# mantém apenas viagens inteiramente dentro do período de 5 às 6 da manhã
gtfs_filtrado <- filter_by_time_of_day(
  gtfs,
  from = "05:00:00",
  to = "06:00:00",
  full_trips = TRUE
)

head(
  gtfs_filtrado$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

```
# mantém apenas paradas que são visitadas entre 5 e 6 da manhã
gtfs_filtrado <- filter_by_time_of_day(
  gtfs,
  from = "05:00:00",
  to = "06:00:00",
  full_trips = FALSE
)

head(
  gtfs_filtrado $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 Filtro espacial

Por fim, o {gtfstools} também disponibiliza uma função que permite filtrar o objeto GTFS usando um polígono espacial. A `filter_by_sf()` recebe um objeto do tipo `sf/sfc` (representação espacial criada pelo pacote {sf}), ou sua *bounding box*, e mantém os registros cujas viagens são selecionadas por uma operação espacial que também deve ser especificada. Embora aparentemente complicado, esse processo de filtragem é compreendido com facilidade quando apresentado visualmente. Para isso, vamos filtrar a GTFS da SPTrans pela *bounding box* da trajetória 68962. Com o código a seguir, apresentamos a distribuição espacial dos dados não filtrados, com a *bounding box* destacada em vermelho na figura 5.

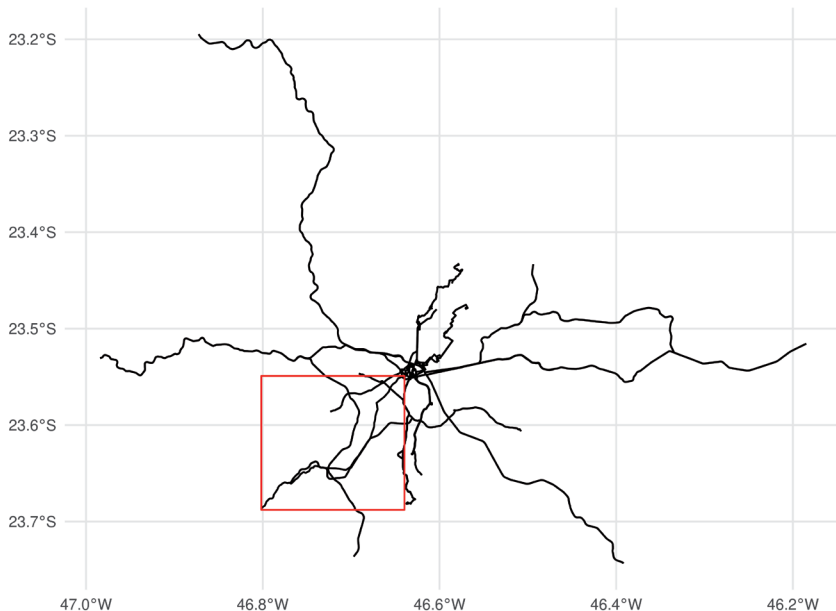
```
# carrega biblioteca ggplot2 para visualização de dados
library(ggplot2)

# cria poligono com a bounding box da trajetoria 68962
trajetoria_68962 <- convert_shapes_to_sf(gtfs, shape_id = "68962")
bbox <- sf::st_bbox(trajetoria_68962)
geometria_bbox <- sf::st_as_sfc(bbox)

# gera geometria de todas as trajetorias do gtfs
todas_as_trajetorias <- convert_shapes_to_sf(gtfs)
```

```
ggplot() +
  geom_sf(data = todas_as_trajetorias) +
  geom_sf(data = geometria_bbox, fill = NA, color = "red") +
  theme_minimal()
```

FIGURA 5  
Distribuição espacial das trajetórias com a *bounding box* da trajetória 68962 em destaque



Fonte: Figura gerada pelo código supracitado.

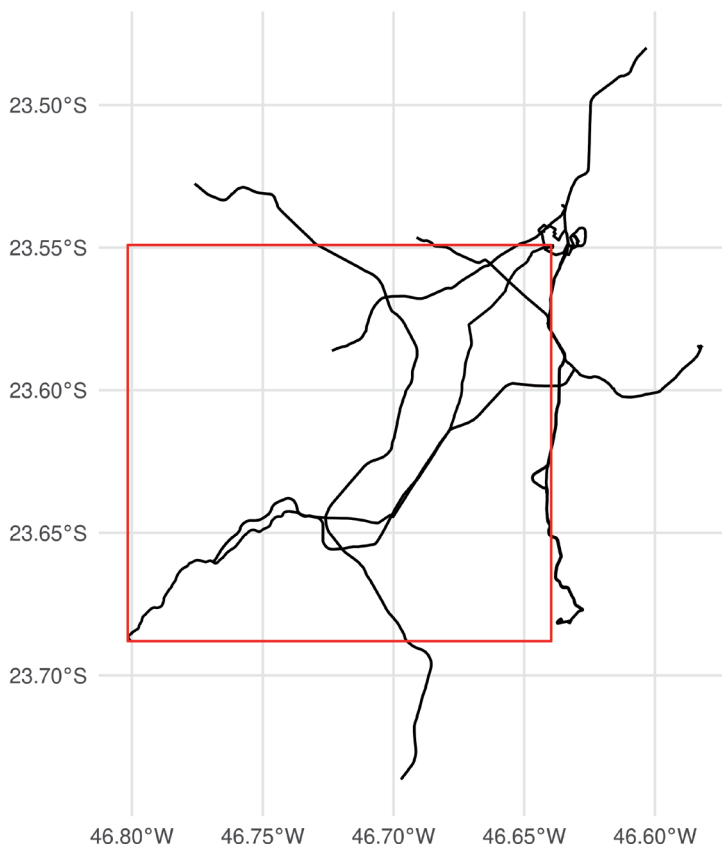
Note que usamos a função `convert_shapes_to_sf()`, também disponibilizada pelo `{gtfstools}`, que converte uma determinada trajetória descrita no arquivo GTFS em um objeto espacial do tipo `sf`. Por padrão, a `filter_by_sf()` mantém os dados relacionados aos registros de viagens cujas trajetórias possuem alguma interseção com o polígono espacial selecionado:

```
gtfs_filtrado <- filter_by_sf(gtfs, bbox)
trajetorias_filtradas <- convert_shapes_to_sf(gtfs_filtrado)

ggplot() +
  geom_sf(data = trajetorias_filtradas) +
  geom_sf(data = geometria_bbox, fill = NA, color = "red") +
  theme_minimal()
```

FIGURA 6

Distribuição espacial das trajetórias com interseções com a *bounding box* da trajetória 68962



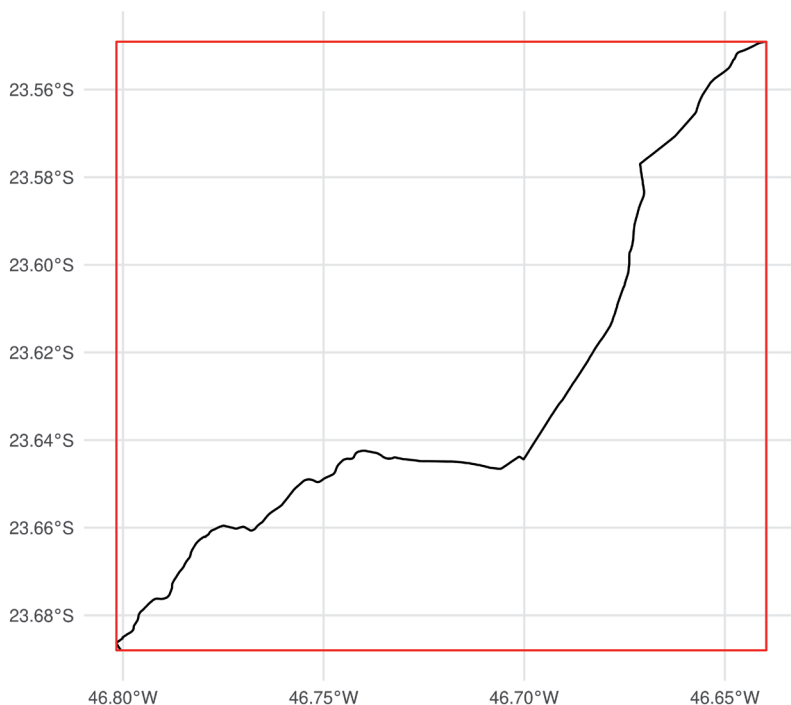
Fonte: Figura gerada pelo código supracitado.

Podemos, no entanto, controlar a operação espacial usada no processo de filtragem. Por exemplo, o código adiante mostra como podemos manter os dados relacionados a viagens que estão *contidas* dentro do polígono espacial:

```
gtfs_filtrado <- filter_by_sf(gtfs, bbox, spatial_operation =  
sf::st_contains)  
trajetorias_filtradas <- convert_shapes_to_sf(gtfs_filtrado)  
  
ggplot() +  
  geom_sf(data = trajetorias_filtradas) +  
  geom_sf(data = geometria_bbox, fill = NA, color = "red") +  
  theme_minimal()
```

FIGURA 7

Distribuição espacial das trajetórias contidas na *bounding box* da trajetória 68962



Fonte: Figura gerada pelo código supracitado.

## 5.4 Validação de arquivos GTFS

Planejadores e pesquisadores frequentemente querem avaliar a qualidade dos arquivos GTFS que estão utilizando em suas análises ou que estão produzindo. Os *feeds* estão estruturados conforme boas práticas adotadas pela comunidade que usa a especificação? As tabelas e colunas estão formatadas corretamente? A informação descrita pelo *feed* parece verossímil (velocidade das viagens, localização das paradas etc.)? Essas são algumas das perguntas que podem surgir ao lidar com dados no formato GTFS.

Para responder a essas e outras perguntas, o `{gtfstools}` inclui a função `validate_gtfs()`, que serve como interface entre o R e o Canonical GTFS Validator, *software* desenvolvido pela MobilityData. O uso do validador requer que o Java versão 11 ou superior esteja instalado.<sup>16</sup>

16. Para informações sobre como checar a versão do Java instalado em seu computador e como instalar a versão correta, caso necessário, conferir o capítulo 3.

Usar a `validate_gtfs()` é muito simples. Primeiro, precisamos baixar o *software* de validação. Para isso, podemos usar a função `download_validator()`, também disponível no pacote, que recebe o endereço de uma pasta na qual o validador deve ser salvo e a versão do validador desejada (por padrão, a mais recente). Como resultado, a função retorna o endereço do arquivo baixado:

```
pasta_temporaria <- tempdir()

endereco_validador <- download_validator(pasta_temporaria)
endereco_validador

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

A segunda (e última) etapa consiste em de fato rodar a função `validate_gtfs()`. Essa função aceita que os dados GTFS a serem validados sejam passados de diferentes formas: i) como um objeto GTFS existente apenas no R; ii) como o endereço de um arquivo GTFS salvo localmente em formato `.zip`; iii) como a URL para um *feed*; ou iv) como uma pasta que contém os dados GTFS não compactados. A função também recebe o endereço para uma pasta onde o resultado da validação deve ser salvo e o endereço para o validador que deve ser usado. Nesse exemplo, vamos fazer a validação a partir do endereço do arquivo GTFS da SPTrans, lido anteriormente:

```
pasta_resultado <- tempfile("validacao_com_endereco")

validate_gtfs(
  endereco,
  output_path = pasta_resultado,
  validator_path = endereco_validador
)

list.files(pasta_resultado)

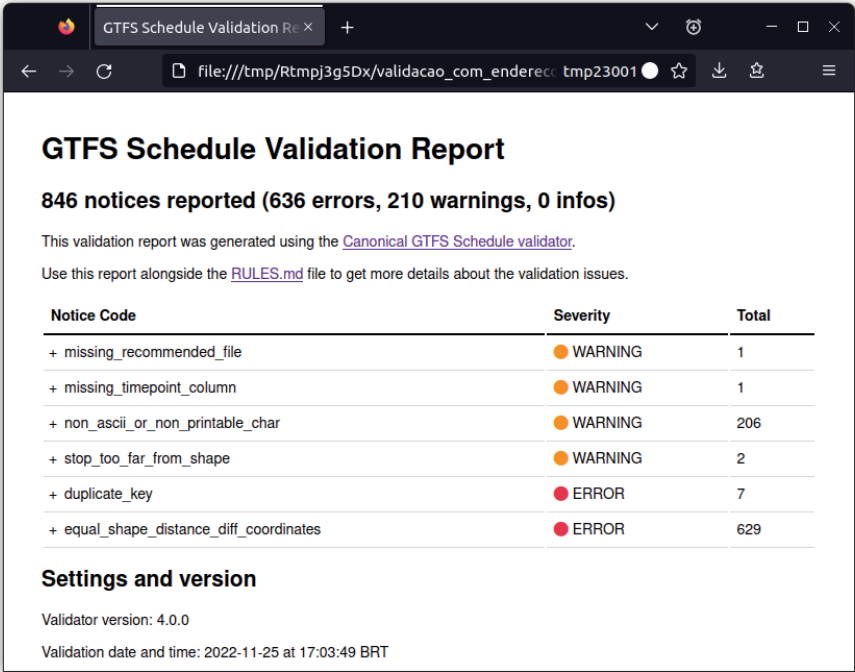
[1] "report.html"          "report.json"          "system_errors.json"
[4] "validation_stderr.txt"
```

Como podemos ver, a validação gera alguns arquivos como resultado:

- `report.html`, mostrado na figura 8, que resume os resultados da validação em uma página HTML (apenas disponível quando utilizado o validador v3.1.0 ou superior);
- `report.json`, que resume a mesma informação apresentada na página HTML, porém em formato JSON, que pode ser usado para processar e interpretar os resultados de forma programática;

- `system_errors.json`, que apresenta eventuais erros de sistema que tenham ocorrido durante a validação e que podem comprometer os resultados; e
- `validation_stderr.txt`, que lista mensagens informativas enviadas pelo validador, incluindo uma lista dos testes realizados, eventuais mensagens de erro etc.<sup>17</sup>

FIGURA 8  
Exemplo de relatório gerado durante a validação



Elaboração dos autores.  
Obs.: Figura cujos leiaute e textos não puderam ser padronizados e revisados em virtude das condições técnicas dos originais (nota do Editorial).

5.5 Fluxo de trabalho com o {gtfstools}: mapeando o *headway* das linhas

Como mostrado nas seções anteriores, o {gtfstools} disponibiliza uma grande caixa de ferramentas que podem ser usadas no processamento e na análise de arquivos GTFS. O pacote, no entanto, oferece diversas outras funções que não puderam ser apresentadas neste livro, por limitação de espaço.<sup>18</sup>

17. Mensagens informativas podem também ser listadas no arquivo `validation_stdout.txt`. As mensagens listadas no `validation_stderr.txt` e no `validation_stdout.txt` dependem da versão do validador utilizada.  
18. A lista completa de funções está disponível em: <<https://ipeagit.github.io/gtfstools/reference/index.html>>.

A apresentação das funções feita até aqui tem um importante caráter demonstrativo, porém não mostra como elas podem ser usadas de forma conjunta na análise de um arquivo GTFS. Esta seção preenche essa lacuna, mostrando como o pacote pode ser usado, por exemplo, para responder à seguinte pergunta: como se distribuem espacialmente os tempos entre veículos de uma mesma linha (os *headways*) no arquivo GTFS da SPTrans?

A primeira etapa é definir o escopo da nossa análise. Para exemplificar, vamos considerar o *headway* no pico da manhã, entre 7h e 9h, em uma típica terça-feira de operação. Para isso, precisamos filtrar o nosso *feed*:

```
# lê o arquivo GTFS
gtfs <- read_gtfs(endereco)

# filtra o GTFS
gtfs_filtrado <- gtfs |>
  remove_duplicates() |>
  filter_by_weekday("tuesday") |>
  filter_by_time_of_day(from = "07:00:00", to = "09:00:00")

# checa resultado do filtro
gtfs_filtrado$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

```
gtfs_filtrado$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

Em seguida, precisamos calcular o *headway* dentro do período estabelecido. Essa informação pode ser encontrada na tabela *frequencies*, mas há um elemento complicador: cada viagem está associada a mais de um *headway*, como vimos anteriormente (um registro para o período entre 7h e 7h59 e outro para o período entre 8h e 8h59). Para resolver essa questão, portanto, vamos calcular o *headway* médio no intervalo entre 7h e 9h.



Os primeiros registros da tabela `frequencies` do arquivo GTFS da SPTrans parecem sugerir que os períodos do dia estão listados sempre de uma em uma hora, porém essa não é uma regra estabelecida na especificação nem é a prática adotada em outros *feeds*. Por isso, vamos calcular a *média ponderada* do *headway* no período especificado. Para isso, precisamos multiplicar cada *headway* pelo intervalo de tempo em que ele é válido e dividir o total dessa soma pelo intervalo de tempo total (duas horas). Para calcular o intervalo de tempo em que cada *headway* é válido, calculamos o começo e o fim do intervalo com a função `convert_time_to_seconds()` e subtraímos o valor do fim pelo do começo, como mostrado a seguir:

```
gtfs_filtrado <- convert_time_to_seconds(gtfs_filtrado)

gtfs_filtrado$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
```

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

Em seguida, calculamos o *headway* médio:

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

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

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

```
head(headway_medio)
```

```
      trip_id headway_medio
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
```

Precisamos agora gerar a trajetória espacial de cada viagem e juntar essa informação à do *headway* médio. Para isso, vamos utilizar a função `get_trip_geometry()`, que, dado um objeto GTFS, gera a trajetória espacial de suas viagens. Essa função nos permite especificar as viagens cujas trajetórias queremos gerar, logo vamos calcular apenas as trajetórias daquelas que estão presentes na tabela de *headways* médios:

```
viagens_selecionadas <- headway_medio$trip_id

trajetorias <- get_trip_geometry(
  gtfs = gtfs_filtrado,
  trip_id = viagens_selecionadas,
  file = "shapes"
)

head(trajetorias)
```

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

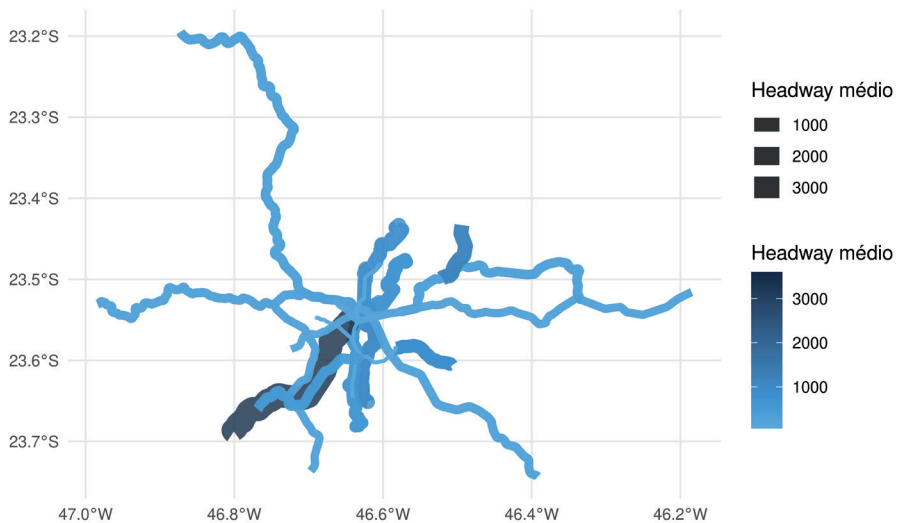
	trip_id	origin_file	geometry
1	CPTM L07-0	shapes	LINESTRING (-46.63535 -23.5...
2	CPTM L07-1	shapes	LINESTRING (-46.87255 -23.1...
3	CPTM L08-0	shapes	LINESTRING (-46.64073 -23.5...
4	CPTM L08-1	shapes	LINESTRING (-46.98404 -23.5...
5	CPTM L09-0	shapes	LINESTRING (-46.77604 -23.5...
6	CPTM L09-1	shapes	LINESTRING (-46.69711 -23.7...

Geradas as trajetórias espaciais de cada viagem, precisamos juntá-las à informação de *headway* médio e, em seguida, configurar o nosso mapa como desejado. No exemplo a seguir, usamos cores e espessuras de linhas que variam de acordo com o *headway* de cada viagem:

```
traj_com_headways <- merge(
  trajetorias,
  headway_medio,
  by = "trip_id"
)
```

```
# configura figura
ggplot(traj_com_headways) +
  geom_sf(aes(color = headway_medio, size = headway_medio),
    alpha = 0.8) +
  scale_color_gradient(high = "#132B43", low = "#56B1F7") +
  labs(color = "Headway médio", size = "Headway médio") +
  theme_minimal()
```

FIGURA 9  
Distribuição espacial dos *headways* no arquivo GTFS da SPTrans



Fonte: Figura gerada pelo código supracitado.

Como podemos ver, o pacote `{gtfstools}` torna o desenvolvimento de análises de *feeds* de transporte público algo fácil e que requer apenas o conhecimento básico de pacotes de manipulação de tabelas (como o `{data.table}` e o `{dplyr}`). O exemplo apresentado nesta seção mostra como muitas de suas funções podem ser usadas conjuntamente para revelar aspectos importantes de sistemas de transporte público descritos no formato GTFS.

## AVALIAÇÃO DE IMPACTO DE PROJETOS DE TRANSPORTE

O objetivo desta seção é apresentar o passo a passo de um exemplo de avaliação do impacto de um projeto de infraestrutura de transportes sobre as condições de acessibilidade urbana locais usando R.

Embora análises de acessibilidade sejam frequentemente utilizadas na literatura científica há mais de duas décadas, apenas recentemente agências de transporte e tomadores de decisão começaram a focar em questões de acessibilidade urbana nas suas atividades de planejamento e operação de sistemas de transporte público (Papa *et al.*, 2015; Boisjoly e El-Geneidy, 2017). Em grande medida, isso é resultado da dificuldade de incorporar análises de acessibilidade a métodos de avaliação de projetos e a atividades de planejamento do dia a dia (Silva *et al.*, 2017; Büttner, 2021).

Nesta seção, usamos o projeto de expansão do metrô de Fortaleza como estudo de caso para ilustrar como realizar uma avaliação de impacto de projetos de infraestrutura de transporte sobre a acessibilidade urbana em R utilizando os pacotes apresentados nos capítulos anteriores. O capítulo 6 apresenta um método tanto para medir o impacto de investimentos de transporte sobre o nível médio de acessibilidade da população quanto para investigar como esse impacto se distribui espacialmente e entre grupos socioeconômicos, afetando as desigualdades de acesso a oportunidades. A aplicação do método envolve o uso e a manipulação de diferentes arquivos GTFS, o cálculo de matrizes de tempo de viagem, a tomada de decisão por trás da escolha de qual medida de acessibilidade utilizar, a estimativa dos níveis de acessibilidade, a visualização espacial desses níveis e o cálculo e a análise de indicadores de desigualdade. Assim, esse estudo de caso engloba muitos dos pontos abordados neste livro e serve como uma aplicação prática dos conceitos até então apresentados.

É importante mencionar que uma avaliação de projeto, investimento ou política pública deve abranger um grande leque de critérios de diferentes naturezas. Esses critérios vão desde aspectos de participação social na formulação das políticas e projetos até os seus impactos, considerando dimensões ambientais, econômicas, sociais, entre outras. Embora uma avaliação de impacto de acessibilidade seja muito importante para a caracterização dos potenciais benefícios e do desempenho de uma rede de transportes como um todo, ela oferece uma perspectiva limitada dos efeitos de uma dada política. Avaliações desse tipo devem, portanto, complementar e ser acompanhadas de outras análises que investiguem os demais impactos dos projetos mais a fundo.



## 6 COMPARANDO A ACESSIBILIDADE ENTRE DOIS CENÁRIOS DE TRANSPORTE

Neste capítulo, vamos ilustrar como usar de forma conjunta o material ensinado nos capítulos anteriores para medir o impacto de um projeto de infraestrutura de transporte sobre a acessibilidade urbana. Para avaliar o impacto de projetos de transporte, precisaremos comparar os níveis de acessibilidade antes e depois da implementação do projeto. Para isso, vamos:

- utilizar diferentes conjuntos de arquivos GTFS e fazer as edições necessárias para representar os cenários antes e depois da implementação do projeto;
- calcular duas matrizes de tempo de viagem, uma antes e outra depois do investimento;
- calcular os níveis de acessibilidade, tanto antes quanto depois do investimento; e
- comparar as condições de acessibilidade em cada cenário, examinando como os impactos se distribuem espacialmente e entre populações de diferentes níveis socioeconômicos.

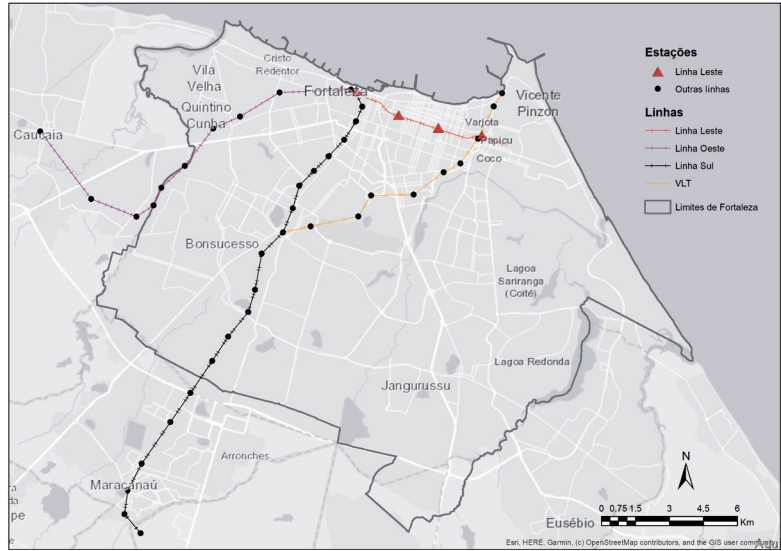
Vamos cobrir esse passo a passo em detalhes, começando por uma breve apresentação do nosso estudo de caso.

### 6.1 Apresentação do estudo de caso

Como estudo de caso, vamos fazer uma avaliação do projeto de construção da linha Leste do metrô de Fortaleza (figura 10). O município de Fortaleza é a capital do estado do Ceará, localizado na região Nordeste do Brasil. Com uma população estimada de 2,7 milhões habitantes, Fortaleza é a quinta cidade mais populosa do país.

Um dos grandes investimentos recentes no sistema de transporte de Fortaleza é a linha Leste do metrô. O traçado da linha Leste possui 7,3 km de extensão e liga o centro da cidade ao bairro Papicu, permitindo a integração das linhas de metrô Sul e Oeste com corredores de veículo leve sobre trilhos (VLT) e com o terminal de ônibus no Papicu (figura 11). Como a linha Leste ainda se encontrava em construção durante a elaboração deste livro, a análise feita neste capítulo trata de uma avaliação *ex ante*, ou seja, em que avaliamos o futuro impacto de um projeto sobre a acessibilidade urbana. Esse tipo de análise se contrapõe a avaliações *ex post*, usadas para avaliar o impacto causado por projetos já implementados.

FIGURA 10  
Sistema de transporte urbano de média e alta capacidade de Fortaleza



Fonte: Braga *et al.* (2022).  
Obs.: Figura reproduzida em baixa resolução e cujos leiaute e textos não puderam ser padronizados e revisados em virtude das condições técnicas dos originais (nota do Editorial).

FIGURA 11  
Detalhamento da futura linha Leste do metrô de Fortaleza



Fonte: Braga *et al.* (2022).  
Obs.: Figura reproduzida em baixa resolução e cujos leiaute e textos não puderam ser padronizados e revisados em virtude das condições técnicas dos originais (nota do Editorial).

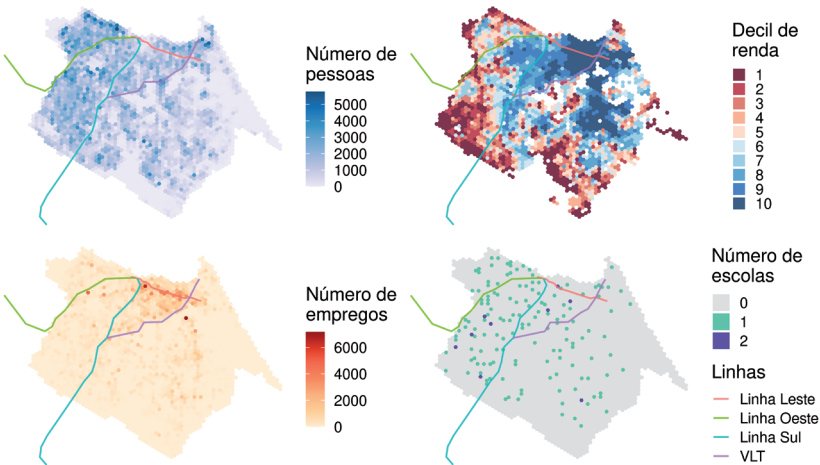
BOX 4  
O cenário de intervenção analisado

É importante notar que a implementação desse projeto também será acompanhada de mudanças nas frequências das linhas Sul e Oeste do metrô e no corredor de VLT Parangaba-Mucuripe, além do racionamento do sistema de ônibus municipais, conforme o Plano de Acessibilidade Sustentável de Fortaleza (Pasfor).<sup>1</sup> Para fins didáticos, no entanto, não incorporamos as mudanças relativas ao racionamento do sistema de ônibus aos cenários da nossa análise. Portanto, o estudo de caso que apresentamos aqui representa um cenário simplificado, que considera apenas a implementação da linha Leste e as mudanças nas frequências das demais linhas do metrô e do VLT.<sup>2</sup>

Elaboração dos autores.  
Notas: <sup>1</sup> Disponível em: <<https://www.pasfor.com.br/>>.  
<sup>2</sup> Uma avaliação mais completa, que leva em consideração todas as mudanças previstas no Pasfor, pode ser encontrada em Braga et al. (2022).

Como mostra a figura 12, a população de Fortaleza está majoritariamente distribuída na região central e na porção oeste da cidade, embora haja alguns pontos de alta densidade populacional no sudeste da cidade. Via de regra, a população de mais alta renda (em tons de azul no mapa da distribuição de decis de renda) está localizada na região do centro expandido e no sudeste do município, enquanto as populações de menor renda (em tons de vermelho) estão principalmente localizadas nas regiões periféricas ao sul e a oeste. Os empregos formais na cidade se distribuem principalmente ao longo de grandes avenidas, com maior concentração no centro da cidade. Em contraste, as escolas públicas de nível médio têm uma distribuição espacial mais equilibrada por todo o território.

FIGURA 12  
Distribuição sociodemográfica, de oportunidades de emprego e de educação e de corredores de transporte público em Fortaleza



Elaboração dos autores.  
Obs.: Figura cujos leiaute e textos não puderam ser padronizados e revisados em virtude das condições técnicas dos originais (nota do Editorial).



## 6.2 Dados GTFS usados na análise

Nesta análise, usaremos os arquivos GTFS disponibilizados pela Etufor e pelo Metrofor. Esses dados descrevem o sistema de transporte público operante na cidade de Fortaleza em outubro de 2019. Para ter acesso a esses dados, usamos o código a seguir, que baixa os *feeds* usando o pacote {httr}:

```
# cria o endereço de arquivos temporários
end_metrofor <- tempfile("metrofor", fileext = ".zip")
end_etufor <- tempfile("etufor", fileext = ".zip")

# baixa dados da metrofor
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/gtfs_for_metrofor_2021-01.zip",
  httr::write_disk(end_metrofor)
)

# baixa dados da etufor
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/gtfs_for_etufor_2019-10.zip",
  httr::write_disk(end_etufor)
)
```

Para simularmos a implementação da nova linha Leste do metrô, precisamos também de um *feed* que descreva como deverá ser a sua operação. Esse *feed* deve conter algumas informações-chave, como o traçado da linha, a localização de suas estações, o tempo de viagem entre elas e a frequência planejada dos serviços. Nesse exemplo, vamos utilizar o arquivo GTFS do projeto da linha Leste criado anteriormente por Braga *et al.* (2022). Assim como os *feeds* da Etufor e do Metrofor, este arquivo está disponível para *download* no repositório do livro no GitHub e pode ser baixado com o código a seguir:

```
# cria o endereço do arquivo temporário
end_linha_leste <- tempfile("linha_leste", fileext = ".zip")

# baixa o GTFS da Linha Leste
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/gtfs_linha_leste.zip",
  httr::write_disk(end_linha_leste)
)
```

Os *feeds* da Etufor e do Metrofor não refletem as mudanças do sistema de transporte público previstas no Pasfor. Para levá-las em consideração no cenário pós-implementação, portanto, precisamos editar esses arquivos usando o pacote {gtfstools}.

Em nosso cenário simplificado, vamos considerar as mudanças de frequência no metrô e no VLT relatadas em Braga *et al.* (2022), com base no Pasfor: i) aumento da frequência da linha Sul do metrô de quatro para dez viagens por hora; ii) aumento da frequência da linha Oeste do metrô de duas para cinco viagens por hora; e iii) aumento da frequência do VLT Parangaba-Mucuripe de duas para oito viagens por hora. Como estamos considerando apenas mudanças em linhas de metrô e VLT, vamos nos ater a edições no *feed* do Metrofor. Em primeiro lugar, precisamos ler esse arquivo com a função `read_gtfs()` e entender como as linhas estão descritas. Para isso, vamos olhar como as tabelas `routes`, `trips` e `calendar` estão estruturadas.

```
library(gtfstools)
```

```
gtfs_metrofor <- read_gtfs(end_metrofor)
```

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

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

```
gtfs_metrofor$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

```
gtfs_metrofor$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						

O *feed* descreve três linhas distintas: os dois corredores de metrô e o corredor de VLT. Como o *feed* não contém uma tabela *frequencies*, cada rota é descrita por diversas viagens, que partem em diferentes horários do dia. Essas viagens são divididas em viagens de ida e de volta e estão todas associadas a um mesmo serviço, que opera em dias úteis e no sábado.

A estratégia que vamos usar para fazer as mudanças necessárias no *feed* é composta pelas etapas descritas a seguir.

- 1) Filtrar o *feed* de forma a manter apenas uma viagem por sentido para cada linha, que servirá para nos dizer o tempo que cada viagem leva entre as suas paradas.
- 2) Adicionar uma tabela *frequencies* ao objeto GTFS, em que descreveremos a frequência de cada viagem.
- 3) “Converter” os registros da recém-adicionada tabela *frequencies* para cronogramas descritos na tabela *stop\_times*. Essa conversão servirá para manter a característica do *feed* original, que descreve as viagens usando apenas a *stop\_times*.

Para manter apenas uma viagem por sentido para cada linha, precisamos filtrar o *feed* usando a `filter_by_trip_id()`. Vamos usar a função para manter apenas o primeiro registro de viagem por linha e por sentido.

```
# identifica a linha na tabela trips em que os primeiros
# registros por rota e
# por sentido estão localizados
indice <- gtfs_metrofor$trips[, .I[1], by = .(route_id,
direction_id)]$V1

# seleciona o identificador de cada linha acima
viagens_selecionadas <- gtfs_metrofor$trips[indice]$trip_id

# filtra o gtfs para manter apenas as viagens acima
gtfs_filtrado <- filter_by_trip_id(
  gtfs_metrofor,
  trip_id = viagens_selecionadas
)

gtfs_filtrado$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

A fim de facilitar nossa edição, vamos mudar o id das viagens, identificando o corredor e o sentido em que elas operam. Essa mudança precisa ser feita tanto na tabela trips quanto na stop\_times.

```
gtfs_filtrado$stop_times[
  ,
  trip_id := data.table::fcase(
    trip_id == "4", "metro_oeste_0",
    trip_id == "19", "metro_oeste_1",
    trip_id == "34", "metro_sul_0",
    trip_id == "96", "metro_sul_1",
    trip_id == "159", "vlt_0",
    trip_id == "181", "vlt_1"
  )
]

gtfs_filtrado$trips[
  ,
  trip_id := data.table::fcase(
    trip_id == "4", "metro_oeste_0",
    trip_id == "19", "metro_oeste_1",
    trip_id == "34", "metro_sul_0",
    trip_id == "96", "metro_sul_1",
    trip_id == "159", "vlt_0",
    trip_id == "181", "vlt_1"
  )
]

gtfs_filtrado$trips
```

	trip_id	trip_headsign	direction_id	block_id	shape_id	service_id
1:	metro_oeste_0	Caucaia	0			4
2:	metro_oeste_1	Moura Brasil	1			4
3:	metro_sul_0	Carlito Benevides	0			4
4:	metro_sul_1	Chico da Silva	1			4
5:	vlt_0	Iate	0			4
6:	vlt_1	Parangaba	1			4

```

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

```

Precisamos agora adicionar uma tabela `frequencies` que descreva a frequência de cada viagem. A especificação requer, no entanto, que listemos o *headway* de cada linha, e não a sua frequência. Como o *headway* é o inverso da frequência, precisamos dividir o intervalo de uma hora (3.600 segundos) pela frequência de cada linha (10 viagens/hora para a linha Sul, 5 viagens/hora para a linha Oeste e 8 viagens/hora para o VLT). Como resultado, temos que os *headways* da linha Sul, da linha Oeste e do VLT passarão a ser, respectivamente, 360, 720 e 450 segundos. Com o código a seguir, criamos uma tabela `frequencies` com os pacotes `{tibble}` e `{data.table}`.

```

frequencias <- tibble::tribble(
  ~trip_id,      ~start_time, ~end_time, ~headway_secs, ~exact_times,
  "metro_oeste_0", "06:00:00", "09:00:00", 720L,      1,
  "metro_oeste_1", "06:00:00", "09:00:00", 720L,      1,
  "metro_sul_0",   "06:00:00", "09:00:00", 360L,      1,
  "metro_sul_1",   "06:00:00", "09:00:00", 360L,      1,
  "vlt_0",         "06:00:00", "09:00:00", 450L,      1,
  "vlt_1",         "06:00:00", "09:00:00", 450L,      1
)

# converte tabela para data.table
data.table::setDT(frequencias)

# cria tabela frequencies no objeto gtfs com base na tabela acima
gtfs_filtrado$frequencies <- frequencias

```

A fim de simplificar este estudo de caso, assumimos que os *headways* listados são válidos no período de 6h às 9h. Essa premissa é válida no nosso caso, pois iremos calcular as matrizes de tempo de viagem apenas durante o pico da manhã. No entanto, caso desejássemos calcular os tempos de viagens em outros períodos do dia, ou mesmo utilizar este arquivo GTFS na operação do dia a dia desses corredores, precisaríamos listar os *headways* para os demais períodos do dia, como fora do pico, madrugada etc. O valor 1 da coluna `exact_times` estabelece que o cronograma das viagens durante o período especificado deve seguir o *headway* à risca, e não de forma aproximada.<sup>19</sup>

19. Para mais detalhes, conferir a descrição da tabela `frequencies` no capítulo 4.

O objeto GTFS que resulta das modificações feitas até aqui já pode perfeitamente ser utilizado para o cálculo de matrizes de tempo de viagem. Buscando retomar a característica inicial do *feed* de não possuir uma tabela *frequencies*, no entanto, vamos “converter” os registros dessa tabela em cronogramas descritos na tabela *stop\_times*. Para isso, usamos a função *frequencies\_to\_stop\_times()*. Como todas as viagens do *feed* são convertidas, a tabela *frequencies* é removida do objeto.

```
gtfs_filtrado <- frequencies_to_stop_times(gtfs_filtrado)

gtfs_filtrado$frequencies
```

NULL

Para verificar se a operação deu certo, vamos olhar para as viagens da linha Oeste no sentido Caucaia (cujo *direction\_id* é 0). Como essa linha deve ter uma frequência de cinco viagens por hora entre 6h e 9h, a tabela *trips* deve conter exatamente dezesseis viagens relacionadas a ela (cinco viagens por hora durante três horas, mais uma viagem começando às 9h).

```
# seleciona apenas as viagens da linha oeste do metrô
metro_linha_oeste <- gtfs_filtrado$trips[grepl("metro_oeste_0",
trip_id)]

# checa número de viagens
nrow(metro_linha_oeste)
```

[1] 16

```
# checa identificador de cada viagem
metro_linha_oeste$trip_id
```

```
[1] "metro_oeste_0_1" "metro_oeste_0_2" "metro_oeste_0_3" "metro_oeste_0_4"
[5] "metro_oeste_0_5" "metro_oeste_0_6" "metro_oeste_0_7" "metro_oeste_0_8"
[9] "metro_oeste_0_9" "metro_oeste_0_10" "metro_oeste_0_11" "metro_oeste_0_12"
[13] "metro_oeste_0_13" "metro_oeste_0_14" "metro_oeste_0_15" "metro_oeste_0_16"
```

A tabela *stop\_times*, por sua vez, deve listar essas viagens partindo a cada doze minutos (o equivalente a um *headway* de 450 segundos). Vamos verificar, portanto, o primeiro registro do cronograma de cada uma das viagens listadas anteriormente.

```

viagens_metro_oeste <- metro_linha_oeste$trip_id

# identifica a linha na tabela stop_times em que estão os
# primeiros registros de
# cada uma das viagens acima
indice_viagens <- gtfs_filtrado$stop_times[
  trip_id %in% viagens_metro_oeste,
  .I[1],
  by = trip_id
]$V1

gtfs_filtrado$stop_times[indice_viagens, .(trip_id, departure_time)]

```

	trip_id	departure_time
1:	metro_oeste_0_1	06:00:00
2:	metro_oeste_0_2	06:12:00
3:	metro_oeste_0_3	06:24:00
4:	metro_oeste_0_4	06:36:00
5:	metro_oeste_0_5	06:48:00
6:	metro_oeste_0_6	07:00:00
7:	metro_oeste_0_7	07:12:00
8:	metro_oeste_0_8	07:24:00
9:	metro_oeste_0_9	07:36:00
10:	metro_oeste_0_10	07:48:00
11:	metro_oeste_0_11	08:00:00
12:	metro_oeste_0_12	08:12:00
13:	metro_oeste_0_13	08:24:00
14:	metro_oeste_0_14	08:36:00
15:	metro_oeste_0_15	08:48:00
16:	metro_oeste_0_16	09:00:00

A operação de “conversão” da tabela frequencies para a stop\_times, portanto, funcionou corretamente, e podemos utilizar o nosso *feed* modificado no cálculo da matriz de tempo de viagem. Para isso, no entanto, precisamos salvar o objeto GTFS na memória em formato .zip, assim como estão salvos os demais dados GTFS que serão utilizados no estudo de caso. Para isso, usamos a função `write_gtfs()`.

```

end_metrofor_modificado <- tempfile("metrofor_modificado",
  fileext = ".zip")

write_gtfs(gtfs_filtrado, end_metrofor_modificado)

```

Temos, agora, quatro arquivos de GTFS distintos:

- o *feed* da Etufor, que descreve o sistema de ônibus que operava em outubro de 2019;
- o *feed* do Metrofor, que descreve a operação em outubro de 2019 das linhas Sul e Oeste do metrô e do VLT Parangaba-Mucuripe;
- o *feed modificado* do Metrofor, que descreve a futura operação das linhas Sul e Oeste do metrô e do VLT Parangaba-Mucuripe, como previsto no Pasfor; e
- o *feed* da linha Leste, que descreve a futura operação dessa linha.

Esses quatro arquivos GTFS serão usados em conjunto para calcular as condições de acessibilidade de Fortaleza antes e depois da implementação da linha Leste. No cenário pré-linha Leste, vamos calcular as matrizes de tempo de viagem com base apenas nos *feeds* de outubro de 2019 do Metrofor e da Etufor, que representam a típica operação de transporte público da cidade antes da implementação do novo corredor. No cenário pós-implementação, consideraremos o *feed* da Etufor, o *feed modificado* do Metrofor, com as frequências do metrô e do VLT alteradas, e o arquivo GTFS da linha Leste, incorporando à análise a operação planejada dessa linha após sua finalização.

### 6.3 Cálculo das matrizes de tempo de viagem

Feitas as edições necessárias ao *feed* do Metrofor e definidos os dados GTFS que vamos usar em cada um dos cenários de transporte público, o próximo passo é calcular as matrizes de tempo de viagem, que posteriormente serão utilizadas para estimar os níveis de acessibilidade. Para isso, vamos utilizar a função `travel_time_matrix()` do pacote `{r5r}`, como apresentado anteriormente no capítulo 3.

Antes de calcular as matrizes, contudo, precisamos organizar os nossos arquivos na estrutura que o `{r5r}` requer. Com o código a seguir, criamos uma pasta separada para cada um dos nossos cenários (antes e depois) e salvamos nelas os dados necessários para o roteamento:

```
# cria pasta raiz da análise de dados
pasta_analise <- "analise_de_impacto"
dir.create(pasta_analise)

# cria pasta dos cenários
pasta_antes <- file.path(pasta_analise, "antes")
pasta_depois <- file.path(pasta_analise, "depois")
```



```

dir.create(pasta_antes)
dir.create(pasta_depois)

# copia os arquivos para pasta do cenário "antes"
file.copy(from = end_etufor, to = file.path(pasta_antes,
"etufor.zip"))
file.copy(from = end_metrofor, to = file.path(pasta_antes,
"metrofor.zip"))

# copia os arquivos para pasta do cenário "depois"
file.copy(from = end_etufor, to = file.path(pasta_depois,
"etufor.zip"))
file.copy(
  from = end_metrofor_modificado,
  to = file.path(pasta_depois, "metrofor_modificado.zip")
)
file.copy(
  from = end_linha_leste,
  to = file.path(pasta_depois, "linha_leste.zip")
)

# visualiza esquema de arquivos na pasta
fs::dir_tree(pasta_analise)

```

```

analise_de_impacto
+-- antes
|   +-- etufor.zip
|   \-- metrofor.zip
\-- depois
    +-- etufor.zip
    +-- linha_leste.zip
    \-- metrofor_modificado.zip

```

Para estimarmos o tempo de viagem na nossa área de estudo, precisamos ainda de um arquivo com os dados do OSM representando a rede viária local, em formato .pbff. Opcionalmente, iremos utilizar também um arquivo representando a topografia local, em formato .tif. Esses arquivos, assim como os dados GTFS, estão disponíveis para *download* no repositório do livro. Partindo do pressuposto de que a implementação da linha Leste não vai afetar o traçado das ruas e calçadas na região, bem como a topografia local, podemos usar os mesmos arquivos para o cálculo das duas matrizes de tempo de viagem. Com o código

apresentado a seguir, baixamos esses dados e copiamos os arquivos para as pastas dos dois cenários de transporte:

```
# cria endereço temporário dos arquivos na máquina local
end_pbf <- tempfile("rede_viaria", fileext = ".osm.pbf")
end_tif <- tempfile("topografia", fileext = ".tif")

# download dos dados de OSM
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/fortaleza.osm.pbf",
  httr::write_disk(end_pbf)
)

# download dos dados de topografia
httr::GET(
  "https://github.com/ipeaGIT/intro_access_book/releases/
  download/data_1st_edition/topografia3_for.tif",
  httr::write_disk(end_tif)
)

# copia arquivo para as pastas dos cenários antes e depois
file.copy(from = end_pbf, to = file.path(pasta_antes,
"rede_viaria.osm.pbf"))
file.copy(from = end_pbf, to = file.path(pasta_depois,
"rede_viaria.osm.pbf"))

file.copy(from = end_tif, to = file.path(pasta_antes,
"topografia.tif"))
file.copy(from = end_tif, to = file.path(pasta_depois,
"topografia.tif"))

fs::dir_tree(pasta_analise)

analise_de_impacto
+-- antes
|   +-- etufor.zip
|   +-- metrofor.zip
|   +-- rede_viaria.osm.pbf
|   \-- topografia.tif
```

```
\-- depois
+-- etufor.zip
+-- linha_leste.zip
+-- metrofor_modificado.zip
+-- rede_viaria.osm.pbf
\-- topografia.tif
```

Com os dados organizados nas pastas, podemos começar o cálculo das matrizes de tempo de viagem. A primeira etapa é construir a rede de transporte multimodal usada pelo {r5r} no roteamento a partir dos dados da rede viária, do sistema de transporte público e de topografia. Isso é feito com o comando `setup_r5()`, que também retorna uma conexão com o R5. Com o código a seguir, criamos duas redes, uma para cada cenário:

```
# aloca a memória disponível para o Java
options(java.parameters = "-Xmx4G")

# carrega a biblioteca
library(r5r)

# cria a rede de transporte multimodal de cada cenário
con_r5r_antes <- setup_r5(pasta_antes, verbose = FALSE)
con_r5r_depois <- setup_r5(pasta_depois, verbose = FALSE)
```

Criadas as redes de transporte de cada cenário, prosseguimos para o cálculo das matrizes de tempo de viagem. Nesta etapa, vamos utilizar como origens e destinos os centroides de uma grade espacial de hexágonos de Fortaleza, disponibilizada pelo pacote {aopdata}.<sup>20</sup> Cada hexágono tem uma área de 0,11 km<sup>2</sup>, aproximadamente um quarteirão, o que permite uma análise espacial bem detalhada.

Para comparar adequadamente os dois cenários, precisamos calcular as duas matrizes considerando os mesmos parâmetros de viagem. Aqui, vamos considerar viagens a pé ou por transporte público, permitir distâncias de caminhada de até trinta minutos nas pernas de acesso e egresso das paradas de transporte público e limitar o tempo máximo de viagem em até sessenta minutos. Vamos considerar o horário de partida de 7h, durante o horário de pico de uma típica segunda-feira de operação:

---

20. Mais detalhes sobre o pacote são apresentados na seção 5.

```
# baixa os dados da grade espacial
grade_fortaleza <- aopdata::read_grid(city = "Fortaleza")

# calcula os centroides das células da grade
pontos <- sf::st_centroid(grade_fortaleza)

# renomeia o nome da coluna com o id das células
names(pontos)[1] <- "id"

# calcula a matriz de tempo de viagem do cenário "antes"
matriz_antes <- travel_time_matrix(
  con_r5r_antes,
  origins = pontos,
  destinations = pontos,
  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
)

# calcula a matriz de tempo de viagem do cenário "depois"
matriz_depois <- travel_time_matrix(
  con_r5r_depois,
  origins = pontos,
  destinations = pontos,
  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
)

head(matriz_antes)
```

	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(matriz_depois)
```

	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

À primeira vista, nossas matrizes parecem iguais: todos os tempos de viagem na amostra de pares mostrados anteriormente são idênticos. Isso ocorre porque o projeto de expansão do metrô fica restrito a uma área relativamente pequena, no centro da cidade de Fortaleza, e as mudanças nas frequências do VLT e das linhas Sul e Oeste do metrô afetam principalmente as imediações desses corredores. Assim, muitos deslocamentos entre regiões da cidade de fato não são afetados pela implementação do corredor e das mudanças de frequência. Diversos pares origem-destino, no entanto, têm os tempos de viagem entre eles impactados:

```
# une os tempos de viagem dos dois cenários
comparacao <- merge(
  matriz_antes,
  matriz_depois,
  by = c("from_id", "to_id"),
  suffixes = c("_antes", "_depois")
)

# mostra apenas os pares OD cujo tempo que os distancia diminuiu
comparacao[travel_time_p50_antes < travel_time_p50_depois]
```

	from_id	to_id	travel_time_p50_antes
1:	8980104092fffff	8980104531bffff	48
2:	8980104092fffff	8980104536bffff	47
3:	8980104092fffff	898010453c7ffff	44

4:	8980104092ffffff 8980104e803ffff	48
5:	8980104092ffffff 8980104e807ffff	57
---		
10499:	8980104eecffffff 8980104e80bffff	53
10500:	8980104eecffffff 8980104e80ffffff	55
10501:	8980104eecffffff 8980104e863ffff	56
10502:	8980104eecffffff 8980104e87bffff	51
10503:	8980104f1abffff 8980104e87bffff	58
travel_time_p50_depois		
1:	50	
2:	48	
3:	46	
4:	50	
5:	59	
---		
10499:	54	
10500:	57	
10501:	57	
10502:	53	
10503:	59	

6.4 Cálculo da acessibilidade nos cenários antes e depois

O cálculo dos níveis de acessibilidade nos dois cenários é muito simples, exigindo apenas um processamento básico dos dados e a aplicação de uma das funções de cálculo de acessibilidade do pacote {accessibility}. Para facilitar o tratamento dos dados, vamos empilhar as matrizes de tempo de viagem dos dois cenários em uma única tabela, identificando cada cenário com a coluna `cenario`:

```
# combina as matrizes de tempo de viagem dos cenários antes e depois
matriz <- rbind(matriz_antes, matriz_depois, idcol = "cenario")
matriz[, cenario := factor(cenario, labels = c("antes", "depois"))]

matriz

      cenario      from_id      to_id travel_time_p50
1:  antes 89801040323ffff 89801040323ffff          2
2:  antes 89801040323ffff 89801040327ffff         22
3:  antes 89801040323ffff 8980104032bffff         32
4:  antes 89801040323ffff 8980104032ffff         15
5:  antes 89801040323ffff 89801040333ffff         10
---
```

3773154:	depois	8980107b6dbffff	8980107b6cbffff	8
3773155:	depois	8980107b6dbffff	8980107b6cfffff	15
3773156:	depois	8980107b6dbffff	8980107b6d3ffff	9
3773157:	depois	8980107b6dbffff	8980107b6d7ffff	16
3773158:	depois	8980107b6dbffff	8980107b6dbffff	0

Para calcular os níveis de acessibilidade, precisamos de uma tabela com os dados de uso do solo da cidade de Fortaleza. Podemos baixar uma tabela com esses dados usando a função `read_landuse()` do pacote `{aopdata}`, que traz dados de contagem populacional e de oportunidades em cada um dos hexágonos que compõem a grade espacial baixada anteriormente.

```
# baixa dados de uso do solo em fortaleza
dados_fortaleza <- aopdata::read_landuse(
  city = "Fortaleza",
  showProgress = FALSE
)
```

A fim de demonstração, vamos calcular a acessibilidade a postos de trabalho e a escolas públicas de ensino médio na nossa área de estudo. Os dados do total de empregos e de escolas públicas de nível médio em cada hexágono estão listados nas colunas `T001` e `E004`, respectivamente. Vamos renomeá-las para facilitar sua identificação e manter apenas as colunas necessárias na tabela de dados de uso do solo. Vamos manter também as colunas `P001`, de população total em cada hexágono, e `R003`, do decil de renda em que cada hexágono se encontra, que serão úteis mais à frente:

```
# renomeia colunas
colunas_mantidas <- c("id", "empregos", "escolas", "populacao",
  "decil")
data.table::setnames(
  dados_fortaleza,
  old = c("id_hex", "T001", "E004", "P001", "R003"),
  new = colunas_mantidas
)

# deleta as demais colunas que não serão usadas
dados_fortaleza[, setdiff(names(dados_fortaleza),
  colunas_mantidas) := NULL]

dados_fortaleza
```

	id	populacao	decil	empregos	escolas
1:	89801040323fffff	30	1	0	0
2:	89801040327fffff	318	1	7	0
3:	8980104032bfffff	0	NA	0	0
4:	8980104032fffff	103	1	98	0
5:	89801040333fffff	43	1	0	0
---					
2558:	8980107b6cbfffff	2575	4	124	0
2559:	8980107b6cfffff	2997	3	4	0
2560:	8980107b6d3fffff	1751	8	14	0
2561:	8980107b6d7fffff	2032	4	134	0
2562:	8980107b6dbfffff	1896	9	193	0

Uma decisão-chave no cálculo de acessibilidade é a escolha da medida a ser utilizada. É extremamente importante pesar as vantagens e desvantagens de cada medida e compreender quais indicadores se adequam às oportunidades para as quais desejamos calcular os níveis de acessibilidade. Nesse exemplo, utilizaremos duas medidas distintas.

- 1) No cálculo da acessibilidade a empregos, vamos usar a medida de oportunidades cumulativas. Essa métrica nos permite entender quantos empregos são acessíveis dentro de um determinado custo de viagem. Apesar das suas limitações, discutidas no capítulo 2, esse é um dos indicadores mais amplamente utilizados. Isso se deve, em grande medida, à facilidade de comunicar e interpretar seus resultados. Nesse exemplo, vamos estabelecer como limite de custo um tempo de viagem de sessenta minutos, valor muito próximo do tempo médio de deslocamento casa-trabalho por transporte público de Fortaleza em 2019 (cerca de 58 minutos, segundo o Pasfor).
- 2) No cálculo da acessibilidade a escolas públicas, vamos usar a medida de custo mínimo de viagem. Essa métrica é especialmente útil para avaliar a cobertura de serviços públicos essenciais, como educação e saúde básica. Com ela, podemos, por exemplo, identificar as parcelas da população que estão a uma distância maior do que a considerada razoável para acessar esses serviços essenciais.

Como mostrado anteriormente, no capítulo 3, o cálculo dessas medidas pode ser feito com as funções `cumulative_cutoff()` e `cost_to_closest()`, respectivamente, do pacote `{accessibility}`:



```
# carrega a biblioteca
library(accessibility)

# calcula a medida de oportunidades cumulativas
acesso_a_empregos <- cumulative_cutoff(
  matriz,
  land_use_data = dados_fortaleza,
  opportunity = "empregos",
  travel_cost = "travel_time_p50",
  cutoff = 60,
  group_by = "cenario"
)

acesso_a_empregos
```

	id	cenario	empregos
1:	89801040323fffff	antes	48049
2:	89801040327fffff	antes	26044
3:	8980104032bfffff	antes	25862
4:	8980104032fffff	antes	69361
5:	89801040333fffff	antes	48049
---			
5120:	8980107b6cbfffff	depois	378840
5121:	8980107b6cfffff	depois	286878
5122:	8980107b6d3fffff	depois	339878
5123:	8980107b6d7fffff	depois	359648
5124:	8980107b6dbfffff	depois	372565

```
# calcula a medida de tempo mínimo de viagem
tempo_a_escolas <- cost_to_closest(
  matriz,
  land_use_data = dados_fortaleza,
  opportunity = "escolas",
  travel_cost = "travel_time_p50",
  group_by = "cenario"
)

tempo_a_escolas
```

	id	cenario	travel_time_p50
1:	89801040323fffff	antes	36
2:	89801040323fffff	depois	36
3:	89801040327fffff	antes	41

4:	89801040327ffff	depois	41
5:	8980104032bffff	antes	41
---			
5120:	8980107b6d3ffff	depois	19
5121:	8980107b6d7ffff	antes	14
5122:	8980107b6d7ffff	depois	14
5123:	8980107b6dbffff	antes	15
5124:	8980107b6dbffff	depois	15

O resultado da função de custo mínimo de viagem inclui alguns valores Inf (abreviação de infinito). Esse valor é utilizado para sinalizar origens que não conseguem alcançar nenhuma oportunidade, dadas as viagens que compõem a matriz. Em nosso caso, isso significa que as origens listadas com esse valor não conseguem alcançar nenhuma escola pública de nível médio dentro de sessenta minutos (limite de tempo de viagem imposto no cálculo da matriz). Para simplificar os cálculos daqui em diante, vamos considerar que essas regiões estão a oitenta minutos de viagem de uma escola:

```
# substitui valores Inf por tempos de viagem de 80 minutos
tempo_a_escolas[
    ,
    travel_time_p50 := ifelse(is.infinite(travel_time_p50),
    80, travel_time_p50)
]
```

Feito isso, podemos calcular a diferença dos níveis de acessibilidade entre os dois cenários. Esta informação é útil para comunicar de forma mais direta o efeito da implementação da linha Leste e das mudanças de frequência dos outros corredores sobre as condições de acessibilidade na cidade. Para isso, usamos o código adiante:

```
acesso_a_empregos[
    ,
    diferenca := data.table::shift(empregos, type = "lead") - empregos,
    by = id
]

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

## 6.5 Análise dos níveis de acessibilidade antes e depois

Agora que calculamos os níveis de acessibilidade em cada cenário e a diferença entre eles, precisamos compreender como a futura implementação da linha Leste em conjunto com o aumento da frequência das linhas de metrô e VLT deverá impactar as condições de acessibilidade em nossa área de estudo. Como primeira análise exploratória, podemos investigar o impacto dessas mudanças sobre a acessibilidade média da cidade. Olhando primeiro para a acessibilidade a oportunidades de emprego, vamos calcular a quantidade média de empregos acessíveis em cada cenário. Aqui, é importante calcular a média de acessibilidade ponderada pela população de cada célula da grade espacial que estamos usando, visto que células com maiores populações contribuem mais para o nível médio da população como um todo do que células com poucas pessoas.

```
# carrega bibliotecas de visualização de dados
library(ggplot2)
library(patchwork)

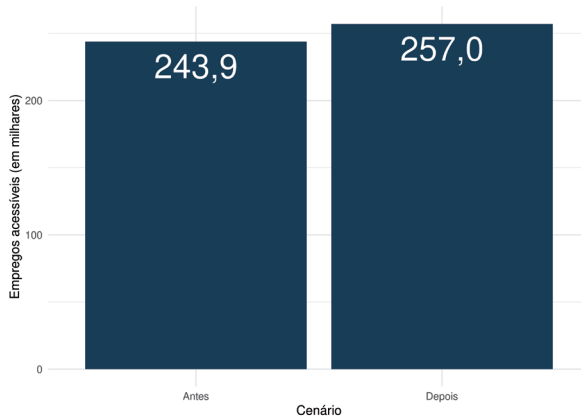
# une tabela de acessibilidade com informações de pessoas
residentes e renda em
# renda nos hexágonos
acesso_a_empregos <- merge(
  acesso_a_empregos,
  dados_fortaleza,
  by = "id"
)

# renomeia colunas com nomes ambíguos pós-união
data.table::setnames(
  acesso_a_empregos,
  old = c("empregos.x", "empregos.y"),
  new = c("acesso_a_empregos", "empregos_no_hexagono")
)

# calcula a média ponderada de acessibilidade em cada cenário
acesso_medio <- acesso_a_empregos[
  ,
  .(acesso = weighted.mean(acesso_a_empregos,
    w = as.numeric(populacao))),
  by = cenario
]
```

```
ggplot(data = acesso_medio, aes(x = cenario, y = acesso / 1000)) +  
  geom_col(fill = "#0f3c53") +  
  geom_text(  
    aes(  
      label = formatC(  
        acesso / 1000,  
        digits = 1,  
        decimal.mark = ",",  
        format = "f"  
      )  
    ),  
    vjust = 1.5,  
    color = "white",  
    size = 10  
  ) +  
  ylab("Empregos acessíveis (em milhares)") +  
  scale_x_discrete(name = "Cenário", labels = c("Antes",  
    "Depois")) +  
  theme_minimal()
```

FIGURA 13  
Média de acessibilidade ao emprego em Fortaleza por cenário de transporte



Fonte: Figura gerada pelo código supracitado.

Os resultados mostram que a população de Fortaleza conseguia acessar em média 243.859 empregos por transporte público em até sessenta minutos de viagem antes da expansão do metrô, em 2019. A implementação da linha Leste e os aumentos nas frequências do metrô e do VLT aumentam esse valor em 5,4%, para em média 257.035 empregos.

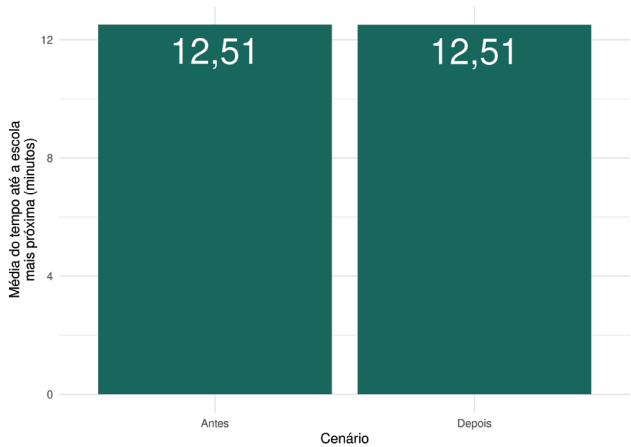
Analisando o tempo médio de acesso à escola pública de ensino médio mais próxima, notamos que as mudanças no sistema de transportes pouco alteram a acessibilidade a essas escolas. Em média, a população de Fortaleza demorava cerca de 12,5 minutos para chegar à escola pública de ensino médio mais próxima de sua casa em 2019. Após a conclusão da extensão do metrô e o aumento da frequência das linhas de metrô e VLT, esse resultado permanece praticamente inalterado.

```
# une tabela de tempo a escolas com informações de pessoas
residentes nos
# hexágonos
tempo_a_escolas <- merge(
  tempo_a_escolas,
  dados_fortaleza,
  by = "id"
)

# calcula a média ponderada de acessibilidade em cada cenário
tempo_medio <- tempo_a_escolas[
  ,
  .(tempo = weighted.mean(travel_time_p50,
    w = as.numeric(populacao))),
  by = cenario
]

ggplot(data = tempo_medio, aes(x = cenario, y = tempo)) +
  geom_col(fill = "#0d6556") +
  geom_text(
    aes(label = formatC(tempo, digits = 2, decimal.mark = ",",
      format = "f")),
    vjust = 1.5,
    color = "white",
    size = 10
  ) +
  ylab("Média do tempo até a escola\nmais próxima (minutos)") +
  scale_x_discrete(name = "Cenário", labels = c("Antes",
    "Depois")) +
  theme_minimal()
```

FIGURA 14  
Média do tempo até a escola pública de nível médio mais próxima em Fortaleza por cenário de transporte



Fonte: Figura gerada pelo código supracitado.

Em síntese, os resultados mostram que o plano de construção da linha Leste do metrô e de ajuste das frequências de outros corredores de transporte público da cidade deverá afetar mais significativamente a acessibilidade a oportunidades de emprego do que a acessibilidade a escolas públicas de nível médio. Isso se explica principalmente pela distribuição espacial desses dois tipos de oportunidade na cidade de Fortaleza: enquanto os empregos se distribuem de forma muito mais concentrada no centro da cidade, as escolas estão muito mais bem distribuídas em todo o território. As mudanças no sistema de transporte público, portanto, devem facilitar o acesso de moradores de regiões mais afastadas aos empregos localizados no centro. Em contrapartida, a distribuição mais equitativa de escolas públicas resulta em condições relativamente boas de acesso a esses locais mesmo antes das mudanças, explicando por que tais mudanças deverão ter impacto tão pequeno no tempo médio de acesso a escolas.

Esses resultados podem ser melhor compreendidos quando observamos a sua distribuição espacial. Antes de fazer isso, no entanto, vamos criar um objeto espacial que descreve o traçado dos corredores de transporte da cidade – o que torna ainda mais evidente o impacto das mudanças no sistema de transportes sobre os níveis de acessibilidade.

```
# lê arquivos gtfs necessários para gerar as geometrias de
cada corredor
gtfs_metrofor <- read_gtfs(end_metrofor)
gtfs_linha_leste <- read_gtfs(end_linha_leste)
```

```

# gtfs da metrofor não contém tabela shapes, logo vamos criar
a geometria
# a partir das tabelas stops e stop_times com a função
get_trip_geometry()
viagens_corredores <- c("4", "34", "159")

# a sequência de paradas de uma das viagens não está ordenada
corretamente,
# então precisamos ordená-las manualmente
gtfs_metrofor$stop_times <- gtfs_metrofor$stop_times[
  order(trip_id, stop_sequence)
]
traj_metrofor <- gtfstools::get_trip_geometry(
  gtfs_metrofor,
  trip_id = viagens_corredores
)

# converte o trajeto em um dos sentidos da linha leste em
geometria espacial
traj_linha_leste <- gtfstools::convert_shapes_to_sf(
  gtfs_linha_leste,
  shape_id = "LL_0"
)

# nomeia cada uma das linhas e une as tabelas
traj_linha_leste$corredor <- "Linha Leste"
traj_metrofor$corredor <- data.table::fcase(
  traj_metrofor$trip_id == 4, "Linha Oeste",
  traj_metrofor$trip_id == 34, "Linha Sul",
  traj_metrofor$trip_id == 159, "VLT"
)

traj_metrofor$origin_file <- NULL
traj_metrofor$trip_id <- NULL
traj_linha_leste$shape_id <- NULL

traj_corredores <- rbind(traj_metrofor, traj_linha_leste)

# duplica a tabela, adiciona coluna identificando cada
cenário e remove a linha
# leste do cenário pré-implementação das mudanças
traj_corredores <- rbind(traj_corredores, traj_corredores)
traj_corredores$scenario <- rep(c("antes", "depois"), each = 4)

```

```

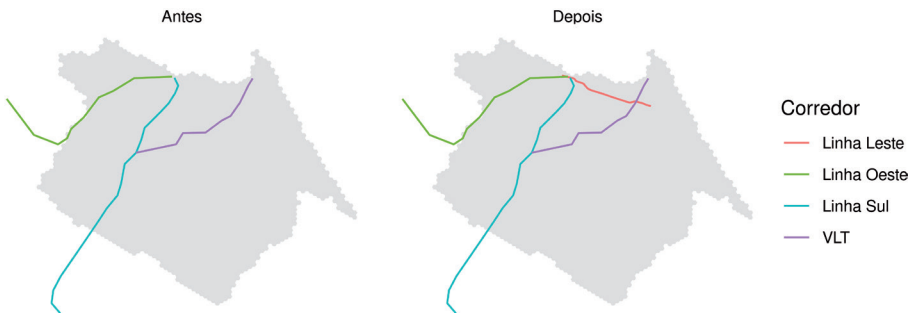
traj_corredores <- subset(
  traj_corredores,
  corredor != "Linha Leste" | cenario != "antes"
)

ggplot() +
  geom_sf(data = grade_fortaleza, fill = "gray90", color = NA) +
  geom_sf(data = traj_corredores, aes(color = corredor)) +
  scale_color_manual(
    name = "Corredor",
    values = c("#F8766D", "#7CAE00", "#00BFC4", "#C77CFF")
  ) +
  facet_wrap(
    ~ cenario,
    nrow = 1,
    labeller = as_labeller(c(antes = "Antes", depois = "Depois"))
  ) +
  theme_void()

```

FIGURA 15

**Distribuição dos corredores de transporte de média e alta capacidade em Fortaleza por cenário de transporte**



Fonte: Figura gerada pelo código supracitado.

Podemos, então, analisar a distribuição espacial dos níveis de acessibilidade nos cenários antes e depois, bem como a diferença de acessibilidade entre eles. Para isso, temos que juntar as estimativas de acessibilidade com a grade espacial da nossa área de estudo. Primeiro, para a acessibilidade a empregos, como mostrado a seguir.



```

# une os dados de acessibilidade com a grade espacial de
# fortaleza e os converte
# em objetos espaciais
acesso_a_empregos <- merge(
  acesso_a_empregos,
  grade_fortaleza,
  by.x = "id",
  by.y = "id_hex"
)
acesso_a_empregos_sf <- sf::st_sf(acesso_a_empregos)

# configura mapas da distribuição de acessibilidade nos
# cenários antes e depois
dist_acesso <- ggplot() +
  geom_sf(
    data = acesso_a_empregos_sf,
    aes(fill = acesso_a_empregos),
    color = NA
  ) +
  facet_wrap(
    ~ cenario,
    nrow = 1,
    labeller = as_labeller(c(antes = "Antes", depois = "Depois"))
  ) +
  scale_fill_viridis_c(
    option = "inferno",
    label = scales::label_number(scale = 1 / 1000)
  ) +
  labs(fill = "Empregos\nnecessíveis\n(em milhares)",
    color = "Corredores") +
  geom_sf(
    data = traj_corredores,
    aes(color = corredor),
    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"))

# configura mapa de diferença entre cenários
dist_diferenca <- ggplot() +
  geom_sf(
    data = subset(acesso_a_empregos_sf, !is.na(diferenca)),

```

```

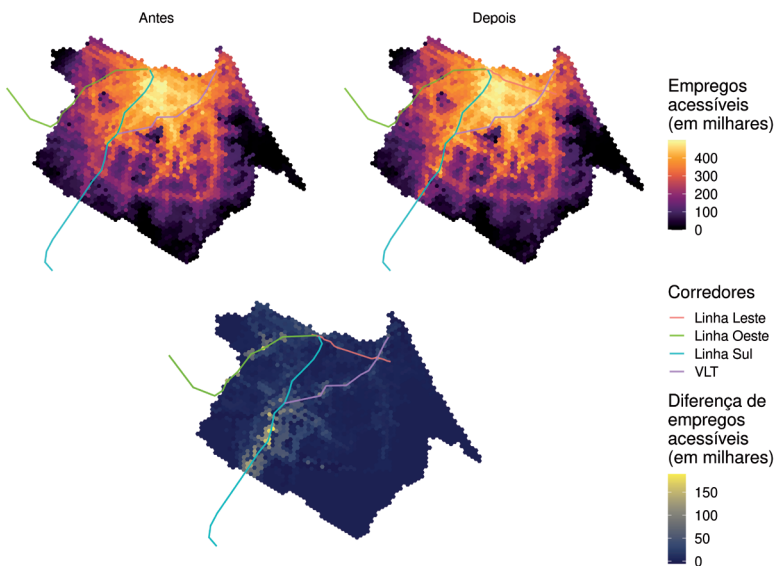
    aes(fill = diferenca),
    color = NA
  ) +
  scale_fill_viridis_c(
    option = "cividis",
    label = scales::label_number(scale = 1 / 1000)
  ) +
  labs(
    fill = "Diferença de\nempregos\nnecessíveis\n(em milhares)",
    color="Corredores"
  ) +
  geom_sf(data = traj_corredores, aes(color = corredor),
    alpha = 0.8) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
    "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# combina as duas figuras
dist_acesso / dist_diferenca + plot_layout(ncol = 1,
  heights = c(1, 1))

```

FIGURA 16

Distribuição espacial dos níveis de acessibilidade ao emprego por cenário de transporte e da diferença entre eles



Fonte: Figura gerada pelo código supracitado.

A figura 16 mostra que as regiões que mais se beneficiam das mudanças no sistema de transportes são aquelas afastadas do centro da cidade, mas que ainda se encontram próximas a estações de média e alta capacidade. Os ganhos de acessibilidade ao emprego se concentram principalmente próximos às estações dos corredores das linhas Sul e Oeste do metrô e, em menor medida, em torno de algumas estações do corredor de VLT Parangaba-Mucuriipe. Mesmo regiões próximas a esses corredores, embora não imediatamente adjacentes a eles, mostram grandes ganhos de acessibilidade, ressaltando o papel da conectividade da rede na garantia de boas condições de acessibilidade. Por sua vez, a região no entorno da linha Leste, que já concentrava os maiores níveis de acessibilidade da cidade mesmo antes da implementação do novo corredor, apresenta melhora nas condições de acessibilidade muito menos expressiva.

Os mapas de distribuição do tempo de acesso até a escola de nível médio mais próxima, no entanto, apresentam uma situação diferente, como mostrado a seguir.

```
# une os dados de tempo de acesso com a grade espacial de
# fortaleza e os
# converte em objetos espaciais
tempo_a_escolas <- merge(
  tempo_a_escolas,
  grade_fortaleza,
  by.x = "id",
  by.y = "id_hex"
)
tempo_a_escolas_sf <- sf::st_sf(tempo_a_escolas)

# configura mapas da distribuição de tempo a escolas nos cenários
# antes e depois
dist_tempo <- ggplot() +
  geom_sf(data = tempo_a_escolas_sf, aes(fill = travel_time_p50),
    color = NA) +
  facet_wrap(
    ~ cenario,
    nrow = 1,
    labeller = as_labeller(c(antes = "Antes", depois = "Depois"))
  ) +
  scale_fill_viridis_c(option = "plasma", direction = -1) +
  labs(fill = "Tempo até\n a escola\n mais próxima\n (em minutos)") +
  geom_sf(
    data = traj_corredores,
    aes(color = corredor),
```

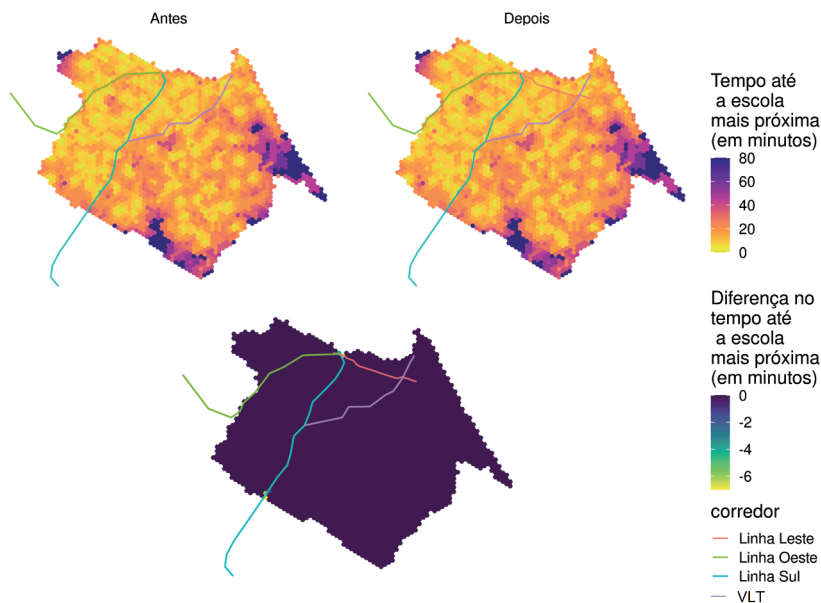
```
    alpha = 0.8,
    show.legend = FALSE
  ) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
    "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# configura mapa de diferença entre cenários
dist_dif_tempo <- ggplot() +
  geom_sf(
    data = subset(tempo_a_escolas_sf, !is.na(diferenca)),
    aes(fill = diferenca),
    color = NA
  ) +
  scale_fill_viridis_c(option = "viridis", direction = -1) +
  labs(
    fill = "Diferença no\ntempo até\n a escola\nmais próxima\n(em\n minutos)"
  ) +
  geom_sf(data = traj_corredores, aes(color = corredor), alpha =
    0.8) +
  scale_color_manual(values = c("#F8766D", "#7CAE00",
    "#00BFC4", "#C77CFF")) +
  theme_void() +
  theme(legend.key.size = unit(0.4, "cm"))

# combina as duas figuras
dist_tempo / dist_dif_tempo + plot_layout(ncol = 1, heights = c(1, 1))
```

FIGURA 17

**Distribuição espacial dos tempos de acesso à escola pública de nível médio mais próxima por cenário de transporte e da diferença entre eles**



Fonte: Figura gerada pelo código supracitado.

A linha Leste e as mudanças de frequência nas outras linhas do metrô e do VLT praticamente não alteram o panorama da acessibilidade a escolas de nível médio em Fortaleza, o que faz com que pouquíssimos hexágonos apresentem algum tipo de ganho de acessibilidade entre os dois cenários – apenas alguns poucos nas imediações de estações de metrô apresentam alguma redução no tempo de acesso a escolas. Como podemos ver, a distribuição da acessibilidade a escolas na cidade segue um padrão muito menos concêntrico que o da distribuição da acessibilidade a empregos. Novamente, isso é uma consequência da distribuição de escolas públicas na cidade: ao contrário da distribuição de empregos, que tende a seguir critérios econômico-financeiros, a distribuição de equipamentos públicos tende a seguir critérios equitativos, buscando priorizar regiões periféricas e de maior concentração de grupos populacionais vulneráveis. Mesmo antes da implementação das mudanças no sistema de transportes, áreas com baixíssima acessibilidade ao emprego, como a região no extremo sul da cidade, já possuíam boas condições de acessibilidade a escolas públicas de nível médio. No entanto, o grau de sucesso das políticas de educação em promover um acesso equitativo pode variar muito entre cidades e para diferentes níveis de ensino, como infantil ou fundamental (Saraiva *et al.*, 2023).

## 6.6 Desigualdade de acessibilidade

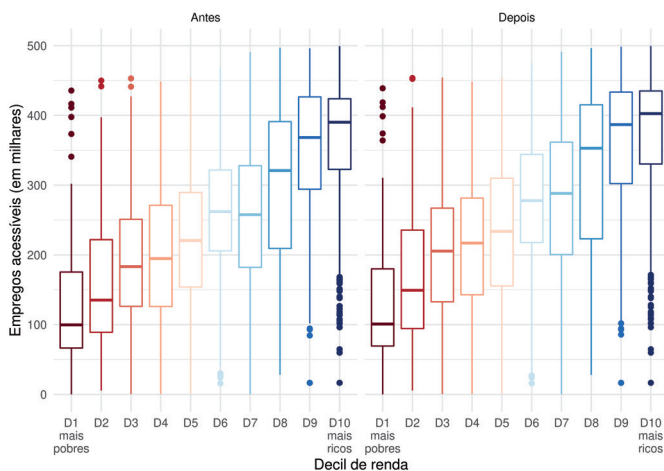
Uma dimensão-chave para a avaliação de políticas de transporte está relacionada aos seus aspectos distributivos. Quem são os principais beneficiados e prejudicados por essas políticas? Sob uma perspectiva de equidade, espera-se que políticas governamentais priorizem melhorar as condições de acessibilidade das pessoas em piores condições socioeconômicas e que mais dependem do transporte público (Pereira, Schwanen e Banister, 2017; Van Wee, 2022).

Nesta seção, vamos examinar como os ganhos de acessibilidade ao emprego decorrentes da construção da linha Leste e do aumento das frequências das linhas do metrô e do VLT estão distribuídos entre pessoas de diferentes níveis de renda. Para isso, precisamos entender como era a distribuição do nível de acesso a empregos entre a população em 2019, antes das mudanças no sistema de transportes, e como ela passará a ser após implementadas essas mudanças. Com o código a seguir, utilizamos a informação do decil de renda *per capita* em que cada célula da nossa grade espacial se encontra para entender a distribuição dos níveis de acessibilidade entre grupos socioeconômicos antes e depois das mudanças.

```
ggplot(data = acesso_a_empregos[populacao > 0]) +  
  geom_boxplot(  
    aes(  
      x = as.factor(decil),  
      y = acesso_a_empregos / 1000,  
      color = as.factor(decil),  
      weight = populacao,  
      group = decil  
    ),  
    show.legend = FALSE  
  ) +  
  facet_wrap(  
    ~ cenario,  
    nrow = 1,  
    labeller = as_labeller(c(antes = "Antes", depois = "Depois"))  
  ) +  
  scale_colour_brewer(palette = "RdBu") +  
  labs(x = "Decil de renda", y = "Empregos acessíveis (em  
    milhares)") +  
  scale_x_discrete(  
    labels = c("D1\nmais\npobres", paste0("D", 2:9),  
      "D10\nmais\nricos")  
  ) +  
  theme_minimal()
```

FIGURA 18

**Distribuição dos níveis de acessibilidade ao emprego entre decis de renda por cenário de transporte**



Fonte: Figura gerada pelo código supracitado.

A figura 18 mostra claramente que as pessoas mais ricas de Fortaleza possuem maiores níveis de acesso a empregos do que seus pares mais pobres, tanto antes quanto depois das mudanças no sistema de transportes. Em Fortaleza, assim como na maior parte das grandes cidades brasileiras, os mais ricos tendem a morar próximo às grandes concentrações de emprego e do centro da cidade, enquanto os mais pobres tendem a ocupar regiões periféricas (Pereira *et al.*, 2022b). Consequentemente, os mais ricos costumam ter melhores condições de acessibilidade urbana do que os mais pobres, não apenas porque moram mais perto dos empregos, mas também porque, geralmente, moram em regiões centrais mais bem servidas de transporte público do que as periferias.

Entretanto, é difícil enxergar na figura 18 grandes mudanças nas condições de acessibilidade entre os dois cenários. Lançando mão do mesmo recurso que usamos anteriormente, apresentamos na figura 19 como os *ganhos* de acessibilidade entre os cenários se distribuem por decil de renda:

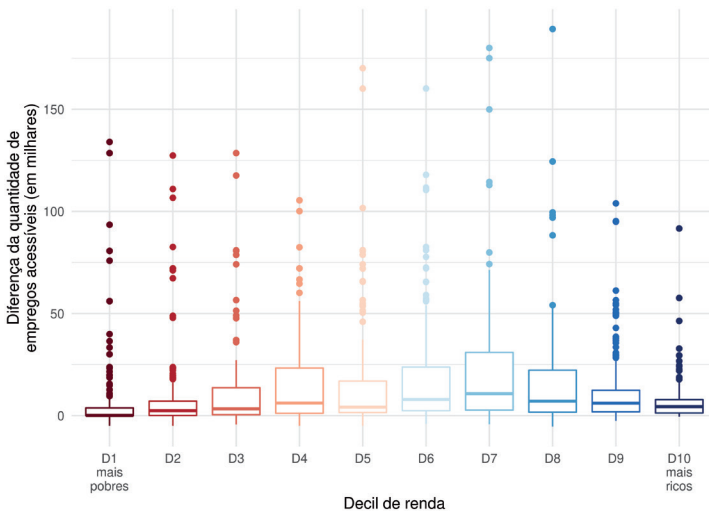
```
ggplot(subset(acesso_a_empregos, populacao > 0 & !is.na(
  diferenca))) +
  geom_boxplot(
    aes(
      x = as.factor(decil),
      y = diferenca / 1000,
      color = as.factor(decil),
      weight = populacao,
```

```

    group = decil
  ),
  show.legend = FALSE
) +
scale_colour_brewer(palette = "RdBu") +
labs(
  x = "Decil de renda",
  y = "Diferença da quantidade de\nempregos acessíveis
(em milhares)"
) +
scale_x_discrete(
  labels = c("D1\nmais\npobres", paste0("D", 2:9),
    "D10\nmais\nricos")
) +
theme_minimal()

```

FIGURA 19  
Distribuição da diferença dos níveis de acessibilidade ao emprego entre cenários de transporte



Fonte: Figura gerada pelo código supracitado.

Como podemos ver, a distribuição de ganhos de acessibilidade segue o formato de U invertido, com maiores ganhos entre as camadas médias da população e menores entre as camadas mais ricas e mais pobres. A célula na qual houve maior aumento de acessibilidade aparece como *outlier* do oitavo decil, apresentando um ganho de quase 200 mil empregos em relação ao cenário antes das mudanças.



Gráficos como os das figuras 18 e 19 não são os mais simples de comunicar, exatamente por serem ricos em nuances e interpretações. Visando facilitar essa comunicação, medidas sintéticas de desigualdade são frequentemente utilizadas como uma maneira simples de medir o impacto de políticas de transporte sobre a desigualdade de acessibilidade. Esse tipo de medida busca resumir a distribuição dos níveis de acessibilidade entre grupos populacionais (aqui, decis de renda) em um único indicador que facilite a compreensão dos resultados e que possa ser usado, por exemplo, na elaboração de planos e metas. As medidas de desigualdade mais frequentemente utilizadas na literatura de acessibilidade são a razão de Palma e o índice de Gini (Lucas, Van Wee e Maat, 2016; Guzman e Oviedo, 2018; Pritchard *et al.*, 2019).

Nesse exemplo, vamos calcular a razão de Palma dos cenários antes e depois das mudanças. Essa medida é o resultado da divisão da acessibilidade média dos 10% mais ricos pela acessibilidade média dos 40% mais pobres:

$$P = \frac{\overline{A_{tp10}}}{\overline{A_{bt40}}} \quad (7)$$

Em que  $P$  é a razão de Palma,  $\overline{A_{tp10}}$  é a acessibilidade média dos 10% mais ricos e  $\overline{A_{bt40}}$  é a acessibilidade média dos 40% mais pobres.

#### BOX 5

##### Por que usar a razão de Palma?

Uma das vantagens da razão de Palma em relação ao índice de Gini é a sua facilidade de comunicação e interpretação. Valores maiores do que 1 indicam um cenário em que os ricos possuem níveis médios de acessibilidade maiores do que os dos pobres, e valores menores do que 1, a situação inversa. Outra vantagem da razão de Palma é que ela reflete claramente como a desigualdade varia entre dois grupos de especial interesse para nós: os mais privilegiados e os mais vulneráveis de uma população. O índice de Gini, por sua vez, estima o quanto uma distribuição desvia de uma situação hipotética em que todos possuem o mesmo nível de acesso, mas nada diz sobre as condições socioeconômicas daqueles que detêm os maiores e menores níveis de acessibilidade. Se uma determinada política aumenta os níveis de acessibilidade de pessoas de alta renda que moram em locais de baixa acessibilidade, por exemplo, o índice de Gini indicaria que houve uma diminuição da desigualdade, mesmo que nenhuma população vulnerável tenha se beneficiado dessa política. Difícilmente uma política como essa poderia ser considerada equitativa.

Elaboração dos autores.

Ao calcularmos a razão de Palma nos cenários antes e depois da implementação da linha Leste e das mudanças de frequência das linhas de metrô e VLT, conseguimos entender o impacto dessas políticas na desigualdade de acesso a empregos em Fortaleza:

```
# calcula a acessibilidade média dos mais ricos nos dois cenários
acesso_mais_ricos <- acesso_a_empregos[
  decil == 10,
  .(acesso = weighted.mean(acesso_a_empregos, w = as.numeric
    (populacao))),
  by = cenario
]

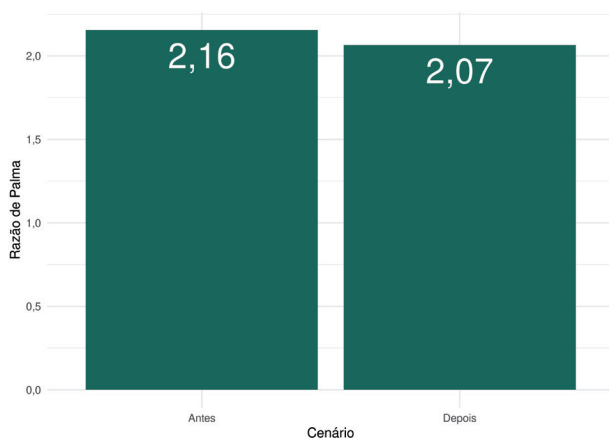
# calcula a acessibilidade média dos mais pobres nos dois cenários
acesso_mais_pobres <- acesso_a_empregos[
  decil %in% 1:4,
  .(acesso = weighted.mean(acesso_a_empregos, w = as.numeric
    (populacao))),
  by = cenario
]

# combina a acessibilidade dos mais ricos e mais pobres
razao_de_palma <- merge(
  acesso_mais_ricos,
  acesso_mais_pobres,
  by = "cenario",
  suffixes = c("_mais_ricos", "_mais_pobres")
)

# calculando Razão de Palma
razao_de_palma[, palma := acesso_mais_ricos / acesso_mais_pobres]

ggplot(data = razao_de_palma, aes(x = cenario, y = palma)) +
  geom_col(fill = "#0d6556") +
  geom_text(
    aes(label = formatC(palma, digits = 2, decimal.mark = ",",
      format = "f")),
    vjust = 1.5,
    color = "white",
    size = 10
  ) +
  scale_y_continuous(
    name = "Razão de Palma",
    labels = scales::label_number(decimal.mark = ",")
  ) +
  scale_x_discrete(name = "Cenário", labels = c("Antes", "Depois")) +
  theme_minimal()
```

FIGURA 20

**Razão de Palma da acessibilidade ao emprego em Fortaleza por cenário de transporte**

Fonte: Figura gerada pelo código supracitado.

A figura 20 aponta que em 2019 a população mais rica de Fortaleza conseguia acessar, em média, 2,16 vezes mais empregos do que a população mais pobre em até sessenta minutos por transporte público. Ela também mostra que a desigualdade medida pela razão de Palma sofreu uma pequena queda entre os cenários pré e pós-intervenções. Ou seja, podemos concluir que, neste cenário simplificado de avaliação da linha Leste, a política de expansão do metrô de Fortaleza combinada com o aumento da frequência das linhas Sul e Oeste do metrô e do VLT seria progressiva. Em outras palavras, essas mudanças, quando analisadas sem considerar as demais intervenções previstas no Pasfor, contribuem positivamente para a redução da desigualdade de acesso a oportunidades de emprego em Fortaleza.<sup>21</sup>

Neste capítulo, focamos na avaliação do impacto de uma política de transporte sobre as condições de acessibilidade da população. Vale ressaltar, no entanto, que a avaliação completa de uma política pública deve considerar também critérios como a participação social em sua formulação, além de outros possíveis impactos ambientais, econômicos e sociais. Embora uma avaliação de impacto de acessibilidade seja muito importante para identificar quais grupos se beneficiam de uma política de transporte e para entender como essa política impacta o desempenho do sistema de transportes como um todo, esse tipo de análise é apenas uma dimensão de avaliação e deve complementar ou ser complementada por outras análises.

21. É importante lembrar que a avaliação apresentada neste capítulo considera um cenário de intervenção simplificado para fins didáticos. Para uma avaliação mais completa da linha Leste do metrô de Fortaleza e das mudanças previstas no Pasfor, que também considera alterações nas linhas de ônibus da cidade, conferir o trabalho de Braga *et al.* (2022).

## DADOS DO PROJETO ACESSO A OPORTUNIDADES

Os objetivos desta seção são: i) apresentar os dados de acessibilidade, de uso do solo e socioeconômicos disponibilizados pelo projeto AOP; e ii) ensinar a baixá-los e utilizá-los com o pacote de R {aopdata}.

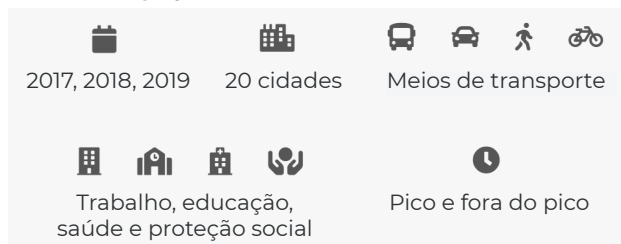
Nos capítulos anteriores, aprendemos sobre o conceito de acessibilidade, como traduzi-lo em medidas quantitativas e como calcular essas medidas usando R. No entanto, frequentemente nos deparamos com situações em que não podemos estimar os níveis de acessibilidade por conta própria, por não termos o tempo ou os dados necessários para isso, ou em que não precisamos fazer esse cálculo porque tais estimativas já foram realizadas por outras pessoas. Ao longo dos próximos capítulos, apresentaremos a base de dados com estimativas de acessibilidade criada e disponibilizada no âmbito do projeto AOP.

O AOP é um projeto de pesquisa liderado pelo Ipea com o objetivo de compreender as condições de transporte e as desigualdades de acesso a oportunidades nas cidades brasileiras. Todos os resultados dos dados produzidos pela equipe do AOP são disponibilizados publicamente. A base disponibilizada pelo AOP contém não apenas estimativas de acessibilidade urbana, mas também informações sobre distribuição e contagem populacional e de atividades econômicas e serviços públicos.<sup>22</sup> Os dados estão agregados espacialmente em uma grade hexagonal que segue o sistema de gradeamento H3, desenvolvido pela Uber (Brodsky, 2018). Cada célula espacial tem cerca de 0,11 km<sup>2</sup>, área similar à coberta por um quarteirão, permitindo análises em alta resolução espacial. Como apresentado na figura 21, as estimativas de acessibilidade estão disponíveis para 2017, 2018 e 2019 e para as vinte maiores cidades do Brasil, considerando diferentes modos de transporte (caminhada, bicicleta, transporte público e automóvel), horários do dia (pico e fora do pico), grupos populacionais (segundo níveis de renda, cor, sexo e idade) e tipos de atividade (empregos, escolas, serviços de saúde e centros de assistência social).

---

22. As metodologias utilizadas para gerar esses dados são apresentadas em detalhe em publicações separadas para os dados populacionais e de uso do solo (Pereira *et al.*, 2022a) e para os dados de acessibilidade (Pereira *et al.*, 2022b).

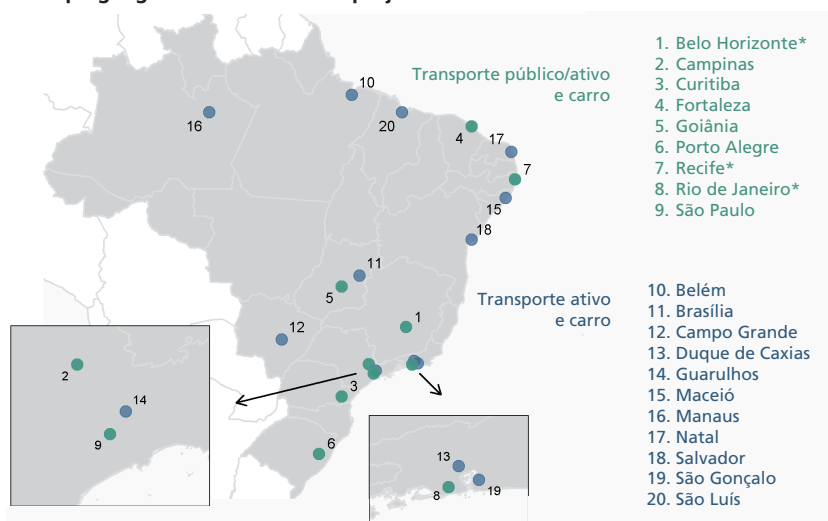
FIGURA 21

**Escopo dos dados do projeto AOP**

Elaboração dos autores.

Os níveis de acessibilidade por transporte público foram estimados apenas para as cidades que disponibilizaram ao projeto dados GTFS de qualidade considerada aceitável. São elas: Belo Horizonte, Campinas, Curitiba, Fortaleza, Goiânia,<sup>23</sup> Porto Alegre, Recife, Rio de Janeiro e São Paulo. Ainda assim, algumas dessas cidades disponibilizaram *feeds* apenas para determinados anos ou compartilharam arquivos considerados não representativos do sistema de transporte público operado na prática, resultando na exclusão das estimativas dessas cidades para alguns anos. A figura 22 indica as cidades para as quais foram calculados os níveis de acessibilidade por modo de transporte.

FIGURA 22

**Escopo geográfico dos dados do projeto AOP**

Elaboração dos autores.

Obs.: As cidades destacadas com asterisco não possuem estimativas por transporte público para todos os anos.

23. O arquivo GTFS de Goiânia descreve a rede de transporte público não apenas do município, mas de toda a sua região metropolitana.

A seguir, duas tabelas resumem os dados disponibilizados pelo projeto. A tabela 10 apresenta os dados de acessibilidade urbana.

**TABELA 10**  
**Indicadores de acessibilidade calculados pelo projeto AOP**

Indicador (código)	Descrição	Tipo de oportunidades	Limites de tempo de viagem
Tempo mínimo de viagem (TMI)	Tempo até a oportunidade mais próxima	Saúde, educação e Centros de Referência de Assistência Social (CRAS)	A pé (60 minutos); bicicleta, transporte público e carro (120 minutos)
Medida cumulativa ativa (CMA)	Quantidade de oportunidades acessíveis em um determinado limite de tempo	Trabalho, saúde, educação e CRAS	A pé e bicicleta (15, 30, 45 e 60 minutos); transporte público e carro (15, 30, 60, 90 e 120 minutos)
Medida cumulativa passiva (CMP)	Quantidade de pessoas que acessam a localidade em um determinado limite de tempo	-	A pé e bicicleta (15, 30, 45 e 60 minutos); transporte público e carro (15, 30, 60, 90 e 120 minutos)

Elaboração dos autores.

Já a tabela 11 apresenta os dados de características socioeconômicas da população e de distribuição espacial de oportunidades.

**TABELA 11**  
**Informações socioeconômicas da população e de distribuição espacial de atividades, segundo ano e fonte de dados**

Dado	Informações	Anos	Fonte
Características sociodemográficas da população	Quantidade de pessoas segundo sexo, faixa de idade e cor/raça; média da renda domiciliar <i>per capita</i>	2010	Censo Demográfico do Instituto Brasileiro de Geografia e Estatística (IBGE)
Estabelecimentos de educação	Quantidade de creches e escolas públicas segundo nível de ensino (infantil, fundamental e médio)	2017 2018 2019	Censo Escolar do Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep)
Estabelecimentos de saúde	Quantidade de estabelecimentos de saúde que atendem pelo Sistema Único de Saúde (SUS) segundo nível de atenção (baixa, média e alta complexidade)	2017 2018 2019	Cadastro Nacional de Estabelecimentos de Saúde (CNES) do Ministério da Saúde
Atividade econômica	Quantidade de empregos formais conforme o nível de instrução dos trabalhadores (baixa, média e alta escolaridade)	2017 2018 2019	Relação Anual de Informações Sociais (Rais) do Ministério da Economia

(Continua)

(Continuação)

Dado	Informações	Anos	Fonte
Estabelecimentos de assistência social	Quantidade de CRAS	2017 2018 2019	Censo do Sistema Único de Assistência Social (Censo SUAS) do Ministério da Cidadania

Elaboração dos autores.

Todas as bases de dados criadas pelo projeto AOP estão disponíveis para *download* em seu [site](#) ou pelo pacote de R {aopdata}. O dicionário de dados pode ser consultado *online*<sup>24</sup> ou em uma sessão de R, com o comando `aopdata::aopdata_dictionary(lang = "pt")`. Os capítulos desta seção apresentam diversos exemplos de como baixar e visualizar esses dados em R.

24. Disponível em: <[https://ipeagit.github.io/aopdata/articles/data\\_dic\\_pt.html](https://ipeagit.github.io/aopdata/articles/data_dic_pt.html)>.

## 7 DADOS SOCIOECONÔMICOS E DE POPULAÇÃO

Os dados de distribuição espacial da população e de suas características em termos de renda *per capita*, cor, sexo e idade, provenientes do Censo 2010 e agregados pelo projeto AOP, podem ser baixados com a função `read_population()`. A função requer que um valor seja passado para o parâmetro `city`, utilizado para indicar a cidade cujos dados devem ser baixados. Para baixar os dados com as informações espaciais de cada célula da grade, o parâmetro `geometry` deve receber o valor `TRUE` (por padrão, esse valor é `FALSE`, fazendo com que a geometria das células não seja baixada).

No exemplo a seguir, mostramos como baixar os dados populacionais e socioeconômicos de Fortaleza:

```
# baixa os dados sociodemográficos do AOP
dados_fortaleza <- aopdata::read_population(
  city = "Fortaleza",
  year = 2010,
  geometry = TRUE,
  showProgress = FALSE
)
```

A tabela baixada inclui o ano de referência do censo demográfico, dados de identificação do hexágono e do município e dados socioeconômicos em colunas cujos nomes estão codificados:

```
names(dados_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"		

A tabela 12 apresenta a descrição de cada uma das colunas da tabela, bem como observações sobre alguns de seus valores. Essa descrição também pode ser consultada na documentação da função, rodando em uma sessão do R o comando `?read_population`.



TABELA 12

**Descrição das colunas da tabela de dados populacionais e socioeconômicos**

Coluna	Descrição	Observação
year	Ano de referência	-
id_hex	Identificador único do hexágono	-
abbrev_muni	Sigla de três letras do município	-
name_muni	Nome do município	-
code_muni	Código de sete dígitos do IBGE do município	-
P001	Quantidade total de pessoas	-
P002	Quantidade de pessoas brancas	-
P003	Quantidade de pessoas negras	-
P004	Quantidade de pessoas indígenas	-
P005	Quantidade de pessoas de cor amarela	-
P006	Quantidade de homens	-
P007	Quantidade de mulheres	-
P010	Quantidade de pessoas de 0 a 5 anos	-
P011	Quantidade de pessoas de 6 a 14 anos	-
P012	Quantidade de pessoas de 15 a 18 anos	-
P013	Quantidade de pessoas de 19 a 24 anos	-
P014	Quantidade de pessoas de 25 a 39 anos	-
P015	Quantidade de pessoas de 40 a 69 anos	-
P016	Quantidade de pessoas de 70 anos ou mais	-
R001	Renda <i>per capita</i> média	Valores de 2010 (em R\$)
R002	Quintil de renda	Valores de 1 (mais pobres) a 5 (mais ricos)
R003	Decil de renda	Valores de 1 (mais pobres) a 10 (mais ricos)
geometry	Geometria espacial	-

Elaboração dos autores.

As subseções a seguir mostram exemplos de visualizações desses dados em forma de mapas.

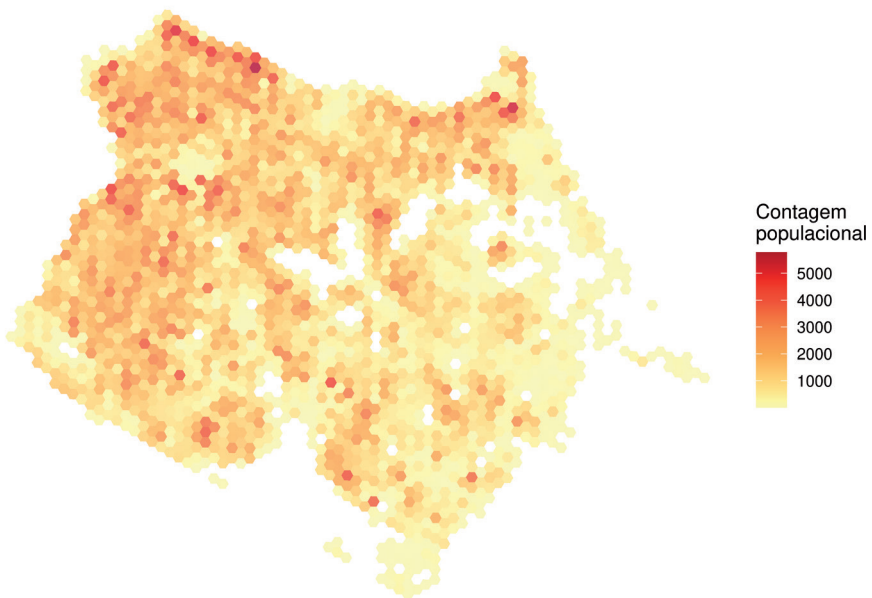
### 7.1 Mapa de população total

No código a seguir, carregamos bibliotecas de visualização de dados e configuramos o mapa. Com um comando, podemos visualizar a distribuição espacial da população de Fortaleza. A figura 23 mostra um mapa coroplético no qual a cor de cada célula da grade espacial é preenchida com base na quantidade total de pessoas que ali residem (variável P001).

```
library(patchwork)
library(ggplot2)

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

FIGURA 23  
Distribuição populacional em Fortaleza



Fonte: Figura gerada pelo código supracitado.

## 7.2 Mapa de população por cor

Além da informação sobre a população total em cada célula, os dados também informam a contagem populacional por classificações de cor (variáveis P002 a P005), sexo (variáveis P006 e P007) e faixa etária (variáveis P010 a P016) em cada unidade espacial. O código a seguir ilustra como é simples calcular a proporção de pessoas negras e brancas em cada hexágono e visualizar essas proporções em um mapa.

```
pop_negra <- ggplot(subset(dados_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 = "Proporção de pessoas negras") +  
  theme_void()  
  
pop_branca <- ggplot(subset(dados_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 = "Proporção de pessoas brancas") +  
  theme_void()  
  
pop_negra + pop_branca
```

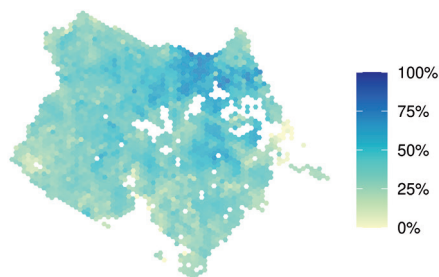
FIGURA 24

**Proporção de pessoas negras e brancas em Fortaleza**

Proporção de pessoas negras



Proporção de pessoas brancas



Fonte: Figura gerada pelo código supracitado.

**7.3 Mapa de população por renda**

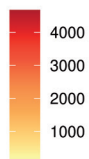
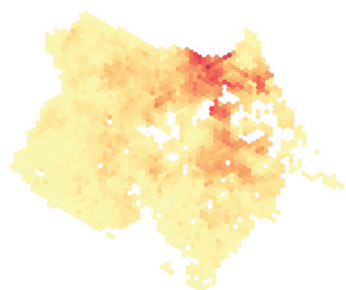
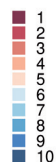
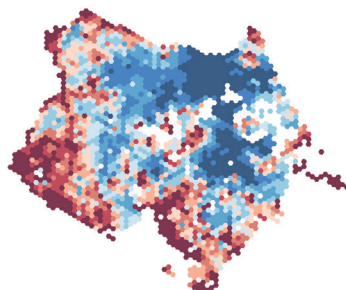
Por fim, os dados também trazem informações sobre a renda *per capita* média de cada hexágono (R001) e sua classificação em termos de quintil (R002) e decil de renda (R003). Com esses dados, podemos visualizar a distribuição espacial dos níveis de renda na cidade.

```
renda <- ggplot(subset(dados_fortaleza, P001 > 0)) +
  geom_sf(aes(fill = R001), color = NA, alpha = 0.8) +
  scale_fill_distiller(name = NULL, palette = "YlOrRd",
    direction = 1) +
  labs(title = "Renda per capita média (R$)") +
  theme_void()

decis <- ggplot(subset(dados_fortaleza, !is.na(R002))) +
  geom_sf(aes(fill = factor(R003)), color = NA, alpha = 0.8) +
  scale_fill_brewer(name = NULL, palette = "RdBu") +
  labs(title = "Decis de renda per capita") +
  theme_void() +
  theme(legend.key.size = unit(0.3, "cm"))

renda + decis
```

FIGURA 25

**Distribuição de renda em Fortaleza**Renda *per capita* média (R\$)Decis de renda *per capita*

Fonte: Figura gerada pelo código supracitado.

## 8 DADOS DE DISTRIBUIÇÃO ESPACIAL DE OPORTUNIDADES

O pacote `{aopdata}` permite baixar, para todas cidades incluídas no projeto, dados de 2017, 2018 e 2019 sobre a distribuição espacial de empregos (de baixa, média e alta escolaridade), estabelecimentos públicos de saúde (de baixa, média e alta complexidade), escolas públicas (ensino infantil, fundamental e médio) e CRAS.

Esses dados podem ser baixados com a função `read_landuse()`, que funciona de forma análoga à `read_population()`. Para isso, basta indicar na chamada a cidade cujos dados são desejados (parâmetro `city`) e o ano de referência (`year`), além de apontar se deseja incluir as informações espaciais dos hexágonos (`geometry`).

No exemplo a seguir, mostramos como baixar os dados de uso do solo de 2019 para Belo Horizonte. Note que essa função resulta em uma tabela que também traz, automaticamente, os dados de população.

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

names(dados_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"
```

A tabela 13 apresenta a descrição das colunas da tabela (excluindo as previamente incluídas na tabela de dados sociodemográficos). Essa descrição também pode ser consultada na documentação da função, rodando em uma sessão de R o comando `?read_landuse`.

TABELA 13  
Descrição das colunas da tabela de dados de distribuição espacial de oportunidades

Coluna	Descrição
year	Ano de referência
id_hex	Identificador único do hexágono
abbrev_muni	Sigla de três letras do município
name_muni	Nome do município
code_muni	Código de sete dígitos do IBGE do município
T001	Quantidade total de empregos
T002	Quantidade de empregos de baixa escolaridade
T003	Quantidade de empregos de média escolaridade
T004	Quantidade de empregos de alta escolaridade
E001	Quantidade total de estabelecimentos públicos de ensino
E002	Quantidade de estabelecimentos públicos de ensino infantil
E003	Quantidade de estabelecimentos públicos de ensino fundamental
E004	Quantidade de estabelecimentos públicos de ensino médio
M001	Quantidade total de matrículas públicas de ensino
M002	Quantidade de matrículas públicas de ensino infantil
M003	Quantidade de matrículas públicas de ensino fundamental
M004	Quantidade de matrículas públicas de ensino médio
S001	Quantidade total de estabelecimentos de saúde
S002	Quantidade de estabelecimentos públicos de saúde de baixa complexidade
S003	Quantidade de estabelecimentos públicos de saúde de média complexidade
S004	Quantidade de estabelecimentos públicos de saúde de alta complexidade
C001	Quantidade total de CRAS
geometry	Geometria espacial

Elaboração dos autores.

As subseções a seguir mostram exemplos de visualizações desses dados em forma de mapas.

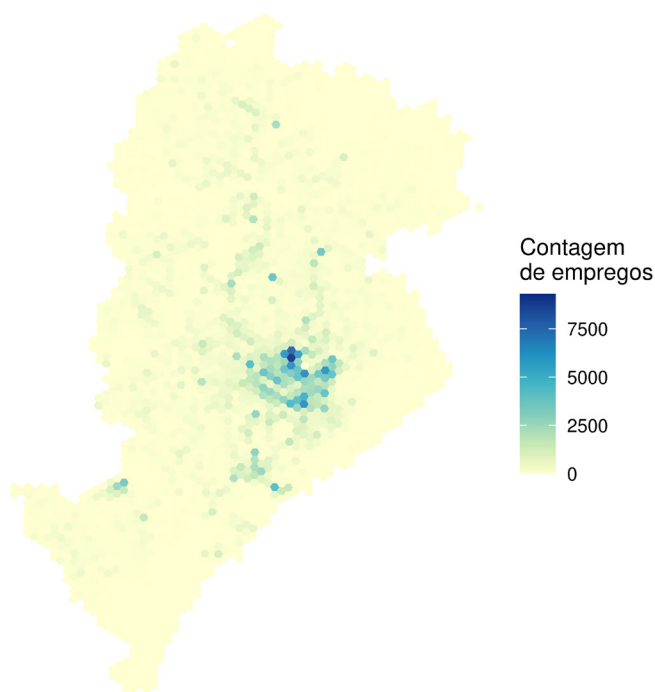
### 8.1 Mapa de empregos

No código adiante, carregamos bibliotecas de visualização de dados e configuramos o mapa. As variáveis iniciadas com a letra T são as que descrevem a distribuição espacial de empregos em cada cidade. A seguir, apresentamos a distribuição espacial do total de empregos em cada hexágono (variável T001) em Belo Horizonte:

```
library(patchwork)
library(ggplot2)

ggplot(dados_bh) +
  geom_sf(aes(fill = T001), color = NA, alpha = 0.9) +
  scale_fill_distiller(palette = "YlGnBu", direction = 1) +
  labs(fill = "Contagem\nde empregos") +
  theme_void()
```

FIGURA 26  
Distribuição espacial de empregos em Belo Horizonte



Fonte: Figura gerada pelo código supracitado.

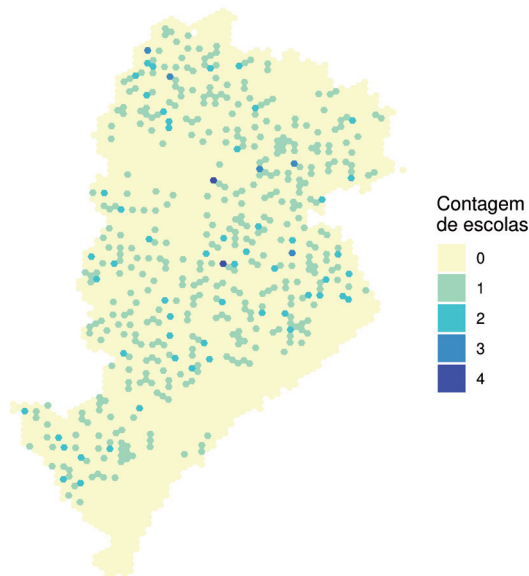
## 8.2 Mapa de escolas

As variáveis que indicam o número de escolas públicas em cada célula, por sua vez, começam com a letra E. No exemplo a seguir, apresentamos a distribuição espacial de todas as escolas públicas de Belo Horizonte (variável E001).



```
ggplot(dados_bh) +
  geom_sf(aes(fill = as.factor(E001)), color = NA, alpha =
    0.9) +
  scale_fill_brewer(palette = "YlGnBu", direction = 1) +
  labs(fill = "Contagem\nde escolas") +
  theme_void()
```

FIGURA 27

**Distribuição espacial de escolas em Belo Horizonte**

Fonte: Figura gerada pelo código supracitado.

**8.3 Mapa de estabelecimentos de saúde**

As variáveis que contêm os dados dos estabelecimentos públicos de saúde em cada célula começam com a letra S. A visualização a seguir compara a distribuição espacial de estabelecimentos públicos de saúde de baixa complexidade (S002) e de alta complexidade (S004).

```
saude_baixa <- ggplot(dados_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(fill = "Contagem de\nestabelecimentos") +
  theme_void()
```

```
saude_alta <- ggplot(dados_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(fill = "Contagem de\nestablecimentos") +
  theme_void()

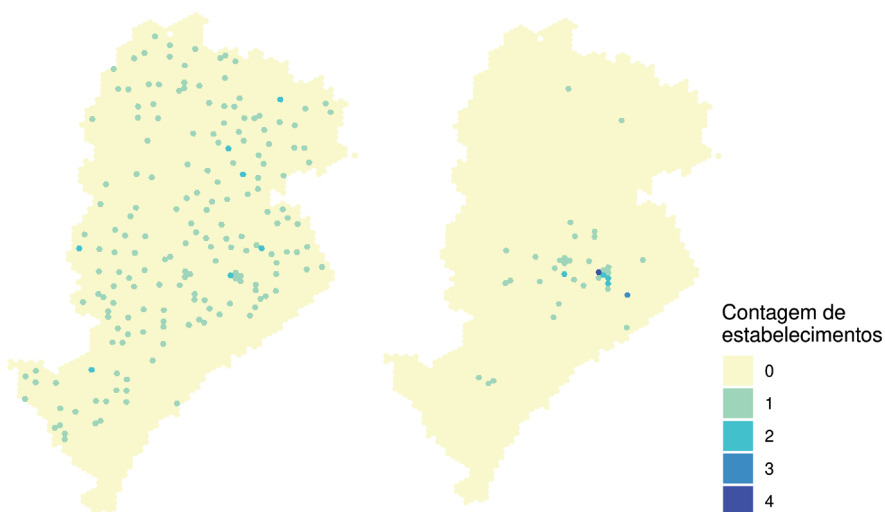
saude_baixa + saude_alta + plot_layout(guides = "collect")
```

FIGURA 28

**Distribuição espacial de estabelecimentos de saúde de baixa e alta complexidade em Belo Horizonte**

Baixa complexidade

Alta complexidade



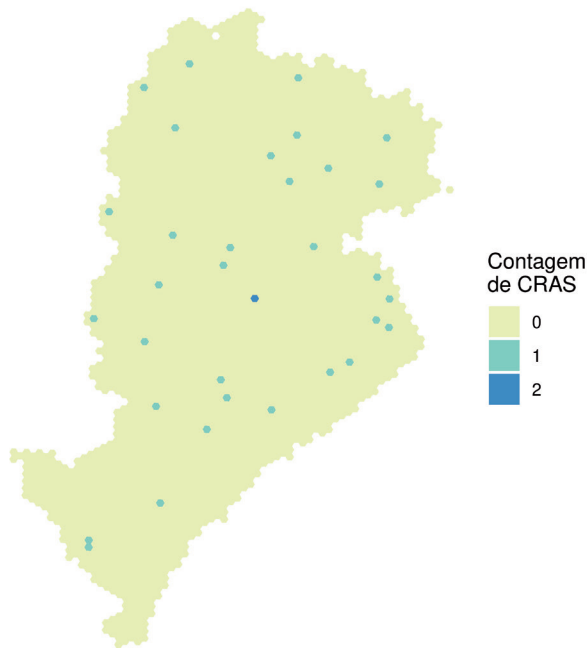
Fonte: Figura gerada pelo código supracitado.

## 8.4 Mapa de CRAS

Por fim, a variável C001 descreve a distribuição espacial de CRAS em cada cidade. A figura 29 apresenta essa distribuição em Belo Horizonte.

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

FIGURA 29  
Distribuição espacial de CRAS em Belo Horizonte



Fonte: Figura gerada pelo código supracitado.

## 9 ESTIMATIVAS DE ACESSIBILIDADE

Finalmente, o pacote {aopdata} também permite baixar, para todas as cidades incluídas no projeto, estimativas de acesso a empregos e a serviços de saúde, de educação e de assistência social para 2017, 2018 e 2019.

Esses dados podem ser baixados com a função `read_access()`, que funciona de maneira análoga às funções `read_population()` e `read_landuse()`, apresentadas anteriormente. Além de indicar a cidade (parâmetro `city`) e o ano de referência (`year`), no entanto, é necessário também informar o modo de transporte (`mode`) e o intervalo do dia (pico, entre 6h e 8h, ou fora do pico, entre 14h e 16h, controlado pelo parâmetro `peak`).

No exemplo a seguir, mostramos como baixar estimativas de acessibilidade no período de pico em São Paulo referentes a 2019. Nesse exemplo, baixamos as estimativas de acessibilidade tanto por automóvel quanto por transporte público e as unimos em um único `data.frame`. Note que essa função resulta em uma tabela que traz também, automaticamente, os dados de população e de uso do solo.

```
# baixa dados do AOP de acessibilidade por transporte público
acesso_tp <- aopdata::read_access(
  city = "São Paulo",
  mode = "public_transport",
  year = 2019,
  peak = TRUE,
  geometry = TRUE,
  showProgress = FALSE
)

# baixa dados do AOP de acessibilidade por automóvel
acesso_carro <- aopdata::read_access(
  city = "São Paulo",
  mode = "car",
  year = 2019,
  peak = TRUE,
  geometry = TRUE,
  showProgress = FALSE
)

# junta os dados em um único dataframe
dados_sp <- rbind(acesso_tp, acesso_carro)
```

```
names(dados_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"    "CMPP15"     "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"    "CMPP130"
[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] "CMPP160"    "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"    "CMPP190"    "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"   "CMPP1120"   "CMPPN120"   "CMPP0005I120"
[191] "CMPP0614I120" "CMPP1518I120" "CMPP1924I120" "CMPP2539I120" "CMPP4069I120"
[196] "CMPP70I120" "TMIST"      "TMISB"      "TMISM"      "TMISA"
[201] "TMIET"      "TMIEI"     "TMIEF"     "TMIEM"     "TMICT"
[206] "geometry"
```

Como podemos ver, assim como nos casos das tabelas de dados sociodemográficos e de uso do solo, os nomes das variáveis de estimativas de acessibilidade estão organizados em códigos, como CMAEF30, TMISB e CMPPM60. Esses códigos são resultado da combinação de três componentes, conforme descrito a seguir.

- 1) O tipo de indicador de acessibilidade: é indicado pelas três primeiras letras do código. Os dados incluem três categorias de indicadores:
  - a) CMA – indicador de acessibilidade cumulativa ativa;
  - b) CMP – indicador de acessibilidade cumulativa passiva; e
  - c) TMI – indicador de tempo mínimo até a oportunidade mais próxima.
- 2) O tipo de atividade para a qual os níveis de acessibilidade foram calculados ou a que pessoas o indicador se refere: é descrito pelas letras seguintes, que estão no meio do código. Os dados incluem estimativas de acessibilidade para diversos tipos de atividades:
  - a) TT – todos os empregos;
  - b) TB – empregos de baixa escolaridade;
  - c) TM – empregos de média escolaridade;
  - d) TA – empregos de alta escolaridade;
  - e) ST – todos os estabelecimentos públicos de saúde;
  - f) SB – estabelecimentos públicos de saúde de baixa complexidade;
  - g) SM – estabelecimentos públicos de saúde de média complexidade;
  - h) SA – estabelecimentos públicos de saúde de alta complexidade;
  - i) ET – todas as escolas públicas;
  - j) EI – escolas públicas de ensino infantil;
  - k) EF – escolas públicas de ensino fundamental;
  - l) EM – escolas públicas de ensino médio;
  - m) MT – todas as matrículas de escolas públicas;
  - n) MI – matrículas de escolas públicas de ensino infantil;
  - o) MF – matrículas de escolas públicas de ensino fundamental;
  - p) MM – matrículas de escolas públicas de ensino médio; e
  - q) CT – todos os CRAS.

No caso do CMP, as letras no meio do nome da variável indicam a qual grupo populacional os níveis de acessibilidade se referem:

- a) PT – toda a população;
- b) PH – população de homens;
- c) PM – população de mulheres;
- d) PB – população branca;
- e) PN – população negra;
- f) PA – população amarela;
- g) PI – população indígena;
- h) P0005I – população de 0 a 5 anos;
- i) P0614I – população de 6 a 14 anos;
- j) P1518I – população de 15 a 18 anos;
- k) P1924I – população de 19 a 24 anos;
- l) P2539I – população de 25 a 39 anos;
- m) P4069I – população de 40 a 69 anos; e
- n) P70I – população de 70 anos ou mais.

3) O tempo limite de viagem utilizado no cálculo do indicador: é descrito pelos números ao final do código. Esses números somente se aplicam ao CMA e ao CMP e incluem os limites de 15, 30, 45, 60, 90 e 120 minutos, dependendo do modo de transporte.

São exemplos:

- CMAEF30: número de escolas públicas de ensino fundamental acessíveis em até 30 minutos de viagem;
- TMISB: tempo de viagem até o estabelecimento público de saúde com serviços de baixa complexidade mais próximo; e
- CMPPM60: quantidade de mulheres que conseguem acessar determinada célula da grade espacial em até 60 minutos de viagem.

A descrição completa das variáveis também pode ser consultada na documentação da função, rodando em uma sessão de R o comando `?read_access`. As subseções a seguir mostram exemplos de visualizações desses dados em forma de mapas e gráficos.

### 9.1 Mapa do tempo para acessar o hospital mais próximo

Neste exemplo, comparamos o tempo de acesso por automóvel e por transporte público até o hospital mais próximo de cada hexágono. Para analisar o tempo mínimo de viagem (TMI) até hospitais de alta complexidade (SA), utilizamos a variável TMISA. Com o código a seguir, carregamos as bibliotecas de visualização de dados e apresentamos a distribuição espacial do tempo de acesso com os dois modos de transporte. Como os tempos de viagem por transporte público costumam ser muito mais longos do que por automóvel, truncamos a distribuição dos valores da variável em sessenta minutos.

```
library(ggplot2)
library(patchwork)

# trunca os tempos de viagem em 60 minutos
dados_sp$TMISA <- ifelse(dados_sp$TMISA > 60, 60, dados_sp$TMISA)

ggplot(subset(dados_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 = "Tempo (em\nminutos)") +
  facet_wrap(
    ~ mode,
    labeller = as_labeller(
      c(car = "Carro", public_transport = "Transporte público")
    )
  ) +
  theme_void()
```

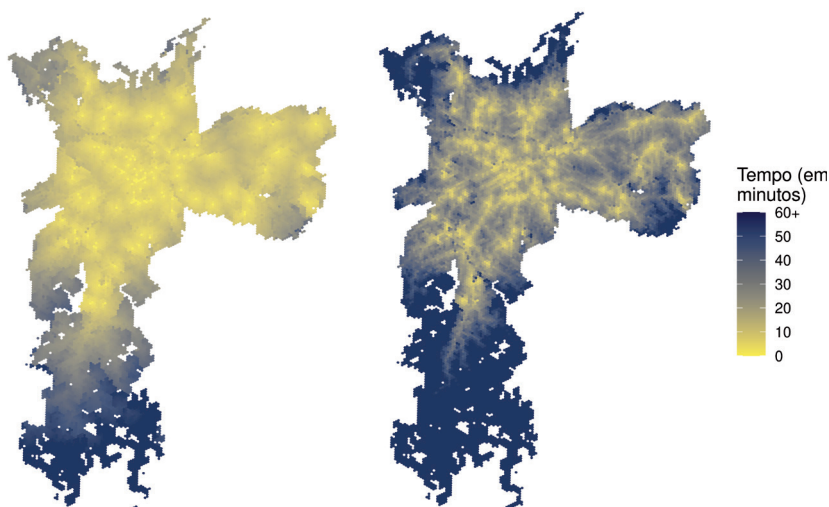


FIGURA 30

**Tempo de viagem até o hospital de alta complexidade mais próximo em São Paulo**

Carro

Transporte público



Fonte: Figura gerada pelo código supracitado.

## 9.2 Mapa da quantidade de empregos acessíveis

Os dados do {aopdata} também tornam muito simples a comparação da quantidade de oportunidades acessíveis em diferentes tempos de viagem. Com o código a seguir, por exemplo, ilustramos como visualizar lado a lado as distribuições espaciais do número de empregos acessíveis em até sessenta e noventa minutos de viagem, respectivamente, por transporte público.

```
# estabelece valores usados na legenda do mapa
limites_legenda <- c(0, max(acesso_tp$CMATT90, na.rm = TRUE) /
1000000)

# configura os mapas

fig60 <- ggplot(subset(acesso_tp, !is.na(mode))) +
  geom_sf(aes(fill = CMATT60 / 1000000), color = NA, alpha = 0.9) +
  scale_fill_viridis_c(option = "inferno", limits = limites_legenda) +
  labs(
    subtitle = "Até sessenta minutos de viagem",
    fill = "Empregos\n(em milhões)"
  ) +
  theme_void()
```

```
fig90 <- ggplot(subset(acesso_tp, !is.na(mode))) +
  geom_sf(aes(fill = CMATT90 / 1000000), color = NA, alpha = 0.9) +
  scale_fill_viridis_c(option = "inferno", limits = limites_legenda) +
  labs(
    subtitle = "Até noventa minutos de viagem",
    fill = "Empregos\n(em milhões)"
  ) +
  theme_void()

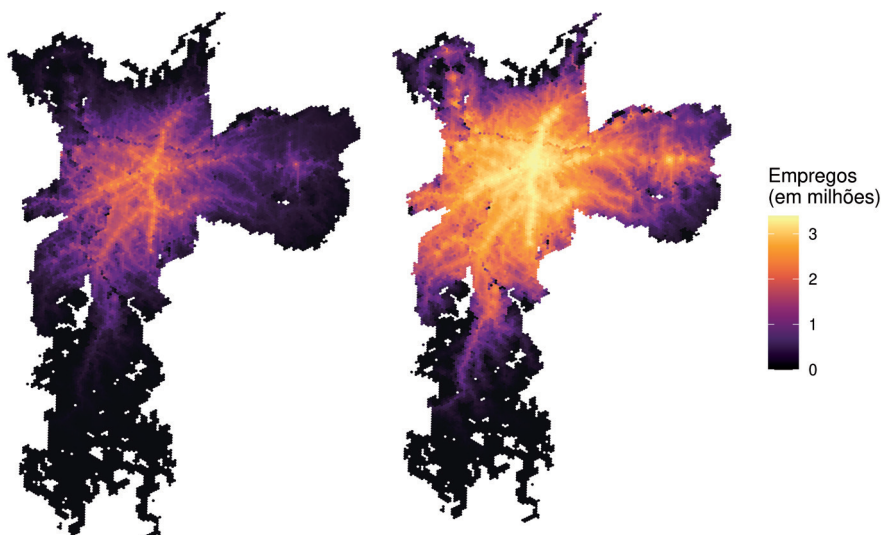
fig60 + fig90 + plot_layout(guides = "collect")
```

FIGURA 31

Quantidade de empregos acessíveis por transporte público em São Paulo

Até sessenta minutos de viagem

Até noventa minutos de viagem



Fonte: Figura gerada pelo código supracitado.

### 9.3 Desigualdades de acesso a oportunidades

Existem diversas maneiras de analisar quão desiguais são as condições de acesso a oportunidades nas cidades brasileiras a partir dos dados do `{aopdata}`. Nesta subseção, apresentamos três exemplos desse tipo de análise.

### 9.3.1 Desigualdade no tempo de acesso a oportunidades

Neste primeiro exemplo, vamos comparar o tempo médio de viagem até o hospital público mais próximo de pessoas de diferentes níveis de renda. Para isso, calculamos o tempo médio de acesso a estabelecimentos de saúde de alta complexidade ponderado pela população de cada célula da nossa grade espacial. Essa ponderação é necessária porque cada célula, por abrigar um número de pessoas diferente das demais, contribui de forma diferente para a média da população como um todo.

Antes de realizar o cálculo, cabe observar que alguns hexágonos da cidade não conseguem acessar nenhum hospital em até duas horas de viagem. Nesses casos, o valor das variáveis de tempo mínimo de viagem é Inf. Para lidar com isso, neste exemplo substituímos todos os valores Inf por um tempo de viagem de 120 minutos.

```
# copia os dados de acesso em um novo dataframe
desigualdade_tp <- data.table::as.data.table(acesso_tp)

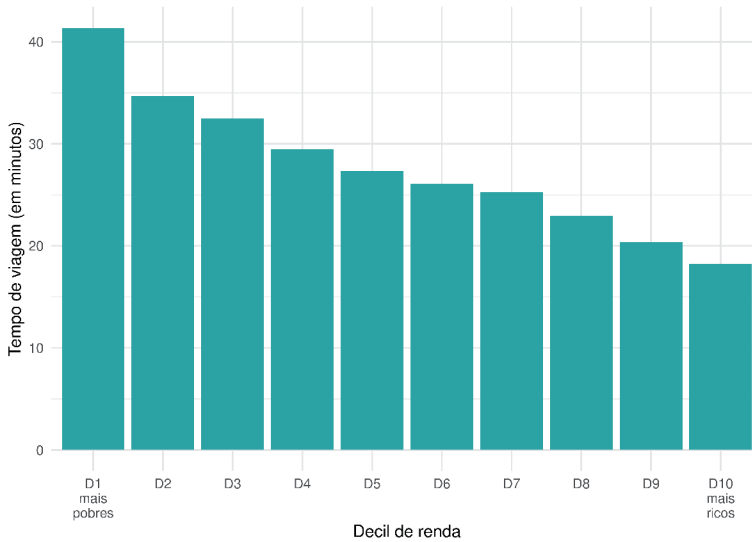
# substitui os valores Inf por 120
desigualdade_tp[, TMISA := ifelse(is.infinite(TMISA), 120, TMISA)]

# calcula o tempo de viagem médio por decil de renda
desigualdade_tp <- desigualdade_tp[
  ,
  .(media = weighted.mean(x = TMISA, w = P001, na.rm = TRUE)),
  by = R003
]
desigualdade_tp <- subset(desigualdade_tp, !is.na(media))

ggplot(desigualdade_tp) +
  geom_col(aes(y = media, x = factor(R003)), fill =
    "#2c9e9e", color = NA) +
  scale_x_discrete(
    labels = c("D1\nmais\npobres", paste0("D", 2:9), "D10\n
    mais\nricos")
  ) +
  labs(x = "Decil de renda", y = "Tempo de viagem (em minutos)") +
  theme_minimal()
```

FIGURA 32

**Média de tempo de viagem por transporte público em São Paulo até o hospital mais próximo**



Fonte: Figura gerada pelo código supracitado.

### 9.3.2 Desigualdade no número de oportunidades acessíveis

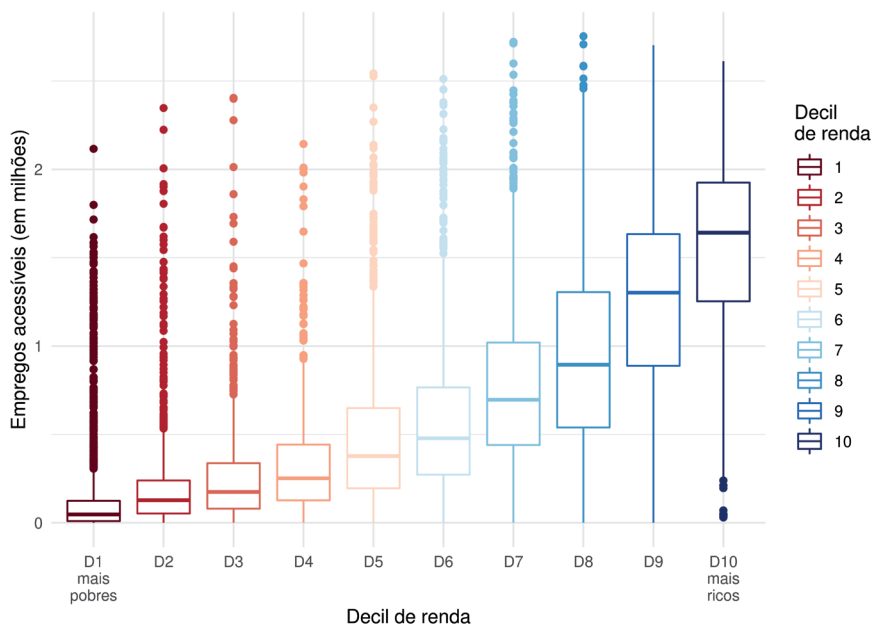
Outra maneira de examinar a desigualdade de acesso a oportunidades é comparar a quantidade de oportunidades acessíveis por diferentes grupos populacionais considerando os mesmos modos de transporte e limites de tempo de viagem. Nesse caso, analisamos o indicador de acessibilidade cumulativa ativa, representado por variáveis cujos códigos começam com CMA na base de dados do {aopdata}. No exemplo a seguir, comparamos a quantidade de empregos acessíveis por pessoas de diferentes decis de renda, considerando viagens de transporte público limitadas em sessenta minutos de viagem.

```
ggplot(subset(acesso_tp, !is.na(R003))) +
  geom_boxplot(
    aes(x = factor(R003), y = CMA60 / 1000000, color = factor(R003))
  ) +
  scale_color_brewer(palette = "RdBu") +
  labs(
    color = "Decil\nde renda",
    x = "Decil de renda",
    y = "Empregos acessíveis (em milhões)"
  ) +
```

```
scale_x_discrete(
  labels = c("D1\nmais\npobres", paste0("D", 2:9), "D10\nmais\nricos")
) +
theme_minimal()
```

FIGURA 33

**Distribuição do número de empregos acessíveis em até sessenta minutos por transporte público em São Paulo**



Fonte: Figura gerada pelo código supracitado.

Por fim, podemos também comparar como o uso de diferentes modos de transporte resulta em diferentes níveis de acessibilidade para a população e como essa diferença varia entre cidades. No exemplo a seguir, comparamos a quantidade de empregos acessíveis em até trinta minutos de viagem a pé e de carro. Para isso, precisamos primeiro baixar os dados de acessibilidade por ambos os modos para todas as cidades do projeto.

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

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

Em seguida, calculamos para cada cidade e para cada modo de transporte a média ponderada do número de empregos acessíveis em até trinta minutos (CMATT30). Feito isso, juntamos essas estimativas em uma única tabela e calculamos a razão entre os níveis de acessibilidade por carro e os níveis de acessibilidade a pé.

```
# calcula a média de empregos acessíveis em 30 minutos

media_carro <- dados_carro[
  ,
  .(acesso_carro = weighted.mean(CMATT30, w = P001,
    na.rm = TRUE)),
  by = name_muni
]

media_caminhada <- dados_caminhada[
  ,
  .(acesso_caminhada = weighted.mean(CMATT30, w = P001,
    na.rm = TRUE)),
  by = name_muni
]

# junta os dados e calcula a razão entre o nível de acesso
# por carro e a pé
media_acesso <- merge(media_carro, media_caminhada)
media_acesso[, razao := acesso_carro / acesso_caminhada]

head(media_acesso)
```

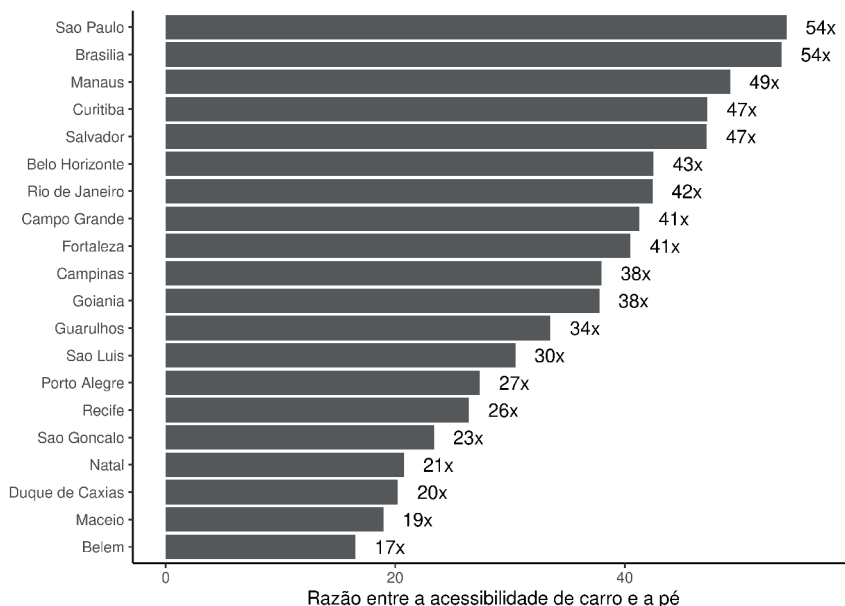
	name_muni	acesso_carro	acesso_caminhada	razao
1:	Belem	155270.4	9392.235	16.53179
2:	Belo Horizonte	529890.0	12464.233	42.51284
3:	Brasilia	220575.9	4110.703	53.65892
4:	Campinas	256333.1	6748.923	37.98133
5:	Campo Grande	172680.5	4181.209	41.29919
6:	Curitiba	494376.9	10471.135	47.21331

Finalmente, podemos visualizar os resultados graficamente:

```
ggplot(media_acesso, aes(x = razao, y = reorder(name_muni,
razao))) +
  geom_bar(stat = "identity") +
  geom_text(aes(x = razao + 3, label = paste0(round(razao),
"x")))) +
  labs(y = NULL, x = "Razão entre a acessibilidade de carro e
a pé") +
  theme_classic()
```

FIGURA 34

**Diferença entre a quantidade de empregos acessíveis por automóvel e por caminhada em até trinta minutos de viagem nas vinte maiores cidades do Brasil**



Fonte: Figura gerada pelo código supracitado.

Como esperado, a figura 34 mostra que é possível acessar muito mais empregos em até trinta minutos com viagens de carro do que com viagens a pé. Essa diferença, porém, varia muito entre cidades. Em São Paulo e em Brasília, viagens de automóvel de até trinta minutos permitem acessar, em média, um número de empregos 54 vezes maior do que viagens a pé de mesma duração. Em Belém, onde observamos a menor diferença, o automóvel permite acessar 17 vezes mais empregos do que a caminhada – razão ainda considerável, porém menor do que a das demais cidades.

## REFERÊNCIAS

- 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.
- 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.
- BANISTER, D. The sustainable mobility paradigm. **Transport Policy**, v. 15, n. 2, p. 73-80, 2008.
- \_\_\_\_\_. The trilogy of distance, speed and time. **Journal of Transport Geography**, v. 19, n. 4, p. 950-959, 2011.
- 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.
- 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.
- 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.
- 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). Disponível em: <[https://repositorio.ipea.gov.br/bitstream/11058/11195/1/td\\_2767.pdf](https://repositorio.ipea.gov.br/bitstream/11058/11195/1/td_2767.pdf)>.
- BRODSKY, I. H3: Uber's hexagonal hierarchical spatial index. **Uber Engineering Blog**, 2018. Disponível em: <<https://www.uber.com/en-BR/blog/h3/>>.
- BÜTTNER, B. Accessibility tools for transport policy and planning. *In*: VICKERMAN, R. (Ed.). **International Encyclopedia of Transportation**. Oxford: Elsevier, 2021. p. 83-86.
- 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.
- 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.

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.

DAMIANI, A. *et al.* Ciência de dados em R. **Curso-R**, 2022. Disponível em: <<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.

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.

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.

FARRINGTON, J.; FARRINGTON, C. Rural accessibility, social inclusion and social justice: towards conceptualisation. **Journal of Transport Geography**, v. 13, n. 1, p. 1-12, 2005.

GEURS, K. T.; 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.

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.

HERSZENHUT, D. *et al.* The impact of transit monetary costs on transport inequality. **Journal of Transport Geography**, v. 99, Feb. 2022.

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.

KANDT, J.; BATTY, M. Smart cities, big data and urban policy: towards urban analytics for the long run. **Cities**, v. 109, Feb. 2021.

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.

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 Pplaces**. Sydney: Committee of the Transport Access Manual/University of Sydney, 2020.

LOVELACE, R.; NOWOSAD, J.; MUENCHOW, J. **Geocomputation with R**. Londres: Chapman and Hall, 2019. Disponível em: <<https://geocompr.robinlovelace.net/>>.

LUCAS, K. *et al.* Transport poverty and its adverse social consequences. **Proceedings of the Institution of Civil Engineers: Transport**, v. 169, n. 6, p. 353-365, 2016.

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.

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.

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.

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.

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.

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.

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.

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.

PAPA, E. *et al.* Accessibility instruments for planning practice: a review of European experiences. **Journal of Transport and Land Use**, v. 9, n. 3, 2015.

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. DOI: <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.

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).

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.

PEREIRA, R. H. M.; SCHWANEN, T.; BANISTER, D. Distributive justice and equity in transportation. **Transport Reviews**, v. 37, n. 2, p. 170-191, 2017.

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. Disponível em: <<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.

VAN WEE, B. Transport modes and accessibility. *In*: VICKERMAN, R. (Ed.). **International encyclopedia of transportation**. Oxford: Elsevier, 2021. p. 32-37.

\_\_\_\_\_. Accessibility and equity: a conceptual framework and research agenda. **Journal of Transport Geography**, v. 104, Oct. 2022.

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.

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.

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



## **EDITORIAL**

### **Coordenação**

Aeromilson Trajano de Mesquita

### **Assistentes da Coordenação**

Rafael Augusto Ferreira Cardoso

Samuel Elias de Souza

### **Supervisão**

Aline Cristine Torres da Silva Martins

### **Revisão**

Bruna Neves de Souza da Cruz

Bruna Oliveira Ranquine da Rocha

Carlos Eduardo Gonçalves de Melo

Elaine Oliveira Couto

Laize Santos de Oliveira

Luciana Bastos Dias

Rebeca Raimundo Cardoso dos Santos

Vivian Barros Volotão Santos

Deborah Baldino Marte (estagiária)

Maria Eduarda Mendes Laguardia (estagiária)

### **Editoração**

Aline Cristine Torres da Silva Martins

Mayana Mendes de Mattos

Mayara Barros da Mota

### **Capa**

Rafael H. M. Pereira

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



## **Missão do Ipea**

Aprimorar as políticas públicas essenciais ao desenvolvimento brasileiro por meio da produção e disseminação de conhecimentos e da assessoria ao Estado nas suas decisões estratégicas.

