

ebook

# 40ª Jornada de Atualização em Informática (JAI 2021)

## Organização

Aline M. S. Andrade (UFBA)

Raul S. Wazlawick (UFSC)



# CSBC

2021

XLI CONGRESSO DA  
SOCIEDADE BRASILEIRA  
DE COMPUTAÇÃO



ALINE MARIA SANTOS ANDRADE  
RAUL SIDNEI WAZLAWICK

**40ª JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA (JAI 2021)**

Florianópolis  
Sociedade Brasileira de Computação – SBC  
2021

# **40ª Jornada de Atualização em Informática (JAI 2021)**

## **Editora**

Sociedade Brasileira de Computação (SBC)

## **Coordenação Geral do SBC 2021**

Frank Siqueira (FSC)

Michele Wangham (UNIVALI)

## **Coordenação da JAI**

Aline Maria Santos Andrade (UFBA)

Raul Sidnei Wazlawick (UFSC)

## **Realização**

Sociedade Brasileira de Computação (SBC)

## **Organização**

Universidade Federal de Santa Catarina

Universidade do Vale do Itajaí

## **Comitê de Programa da JAI 2021**

Aline Marins Paes Carvalho (UFF)

Alirio Sá (UFBA)

Daniel Avila Vecchiato (UFMT)

Daniel Macedo Batista (USP)

Eliana Almeida (UFAL)

Fabiola Nakamura (UFAM)

Flávio Assis Silva (UFBA)

Marcelo Walter (UFRGS)

Márcio Bastos Castro (UFSC)

Marcio Lopes Cornelio (UFPE)

Noemi Rodriguez (PUC- Rio)

Rita Suzana Pitangueira Maciel (UFBA)

Soraia Musse (PUCRS)

Thais Vasconcelos Batista (UFRN)

Vania Bogorny (UFSC)

Dados Internacionais de Catalogação na Publicação (CIP)

J82 Jornada de Atualização em Informática (40. : 2021: Florianópolis, SC)  
40º Jornada de Atualização em Informática [recurso eletrônico] / Organizadores: Aline M. S. Andrade, Raul S. Wazlawick – Porto Alegre : SBC, 2021.

ISBN 978-65-87003-57-3

1. Computação - Congresso. I. Sociedade Brasileira de Computação. II. Universidade Federal de Santa Catarina. III. Universidade do Vale do Itajaí.

CDU 004

## Sinopse

Mantendo sua tradição em apresentar temas relevantes em pesquisa e desenvolvimento, a Jornada de Atualização em Informática (JAI), nesta edição, trata de tecnologias e conceitos da área de computação com bastante impacto no desenvolvimento tecnológico da sociedade moderna. Esta publicação apresenta temas que têm relação com segurança cibernética e internet das coisas (IoT), privacidade de dados, comunicação entre pessoas e sistemas computacionais, e processamento de grandes volumes de dados.

Abrimos este livro com uma apresentação sobre a história da JAI elaborada pelo professor Roberto S. Bigonha (UFMG), que ministrou um curso na I JAI em 1982. Agradecemos ao professor Bigonha por ter aceitado este convite que se tornou um desafio devido à pandemia do Covid-19, uma vez que informações sobre as primeiras versões da JAI, registradas apenas em meio impresso, se encontram inacessíveis em bibliotecas nas universidades. Assim, com os dados que foram possíveis recuperar, o prof. Bigonha faz um prazeroso relato da criação e histórico da Jornada de Atualização em Informática, marcando a comemoração dos seus 40 anos.

Agradecemos ao comitê de programa e a todos os autores que submeteram propostas. Tivemos sete propostas submetidas e selecionamos quatro para serem desenvolvidas como capítulos neste livro. Agradecemos também ao coordenador do CSBC 2021, prof. Frank Siqueira, e à equipe de organização pelo apoio na elaboração deste evento.

Finalmente, gostaríamos de registrar a nossa satisfação e alegria em coordenar a JAI no seu 40º ano de existência.

Aline Maria Santos Andrade (UFBA)

Raul Sidnei Wazlawick (UFSC)

Coordenadores da JAI 2021

## Synopsis

Maintaining its tradition of presenting relevant themes in research and development, the Jornadas de Atualização em Informática (JAI), in this edition, deals with technologies and concepts in Computing with a great impact on the technological development of modern society. This publication presents topics related to cybersecurity and the Internet of Things (IoT), data privacy, communication between people and computer systems, and processing large volumes of data.

We opened this book with a presentation on the history of JAI prepared by Professor Roberto S. Bigonha (UFMG), who gave a course at the 1<sup>st</sup> JAI in 1982. We thank Professor Bigonha for accepting this invitation, which became a challenge due to the Covid-19 pandemic, since information about the first versions of JAI, recorded only in print, is inaccessible in libraries in universities. Thus, with the data that were possible to recover, Prof. Bigonha makes a pleasant account on the creation and history of the JAI, marking the celebration of its 40 years.

We thank the program committee and all the authors who submitted proposals. We had seven proposals submitted and selected four to be developed as chapters in this book. We also thank the coordinator of CSBC 2021, Prof. Frank Siqueira, and the organizing committee for their support in the preparation of this event.

Finally, we would like to record our satisfaction and joy in coordinating JAI in its 40th year of existence.

Aline Maria Santos Andrade (UFBA)

Raul Sidnei Wazlawick (UFSC)

COORDINATORS OF JAI 2021



# Apresentação

A **Jornada de Atualização em Informática** da Sociedade Brasileira de Computação foi criada há 40 anos para prover um fórum acadêmico de atualização científica e tecnológica da comunidade de Ciência da Computação do Brasil. Esse evento consiste em minicursos básicos e avançados ministrados por pesquisadores sêniores da nossa comunidade, e que visam oferecer uma oportunidade especial para acadêmicos, estudantes e profissionais de informática se atualizarem em temas diversos de pesquisa e é vista como um dos mais importantes eventos acadêmicos de atualização científica e tecnológica da nossa comunidade de Computação.

A Sociedade Brasileira de Computação foi criada em 1978, e, em 1981, realizou seu primeiro Congresso Nacional, o I CSBC, na cidade de Florianópolis, Santa Catarina, hospedando, entre outros, dois importantes eventos, o SECOMU e o SEMISH, que, de forma independente, já reuniam pesquisadores da área de Computação desde o início da década de 1970.

A Informática teve seu início no Brasil em 1957 com a importação de um computador Univac-120, adquirido pelo Governo do Estado de São Paulo para calcular o consumo de água na capital paulista. Na sequência, universidades, bancos e empresas estatais importaram computadores cuja programação ficava a cargo de profissionais com formação em áreas tão diversas como Engenharia, Economia, Administração e Ciências Exatas. Como naquela época ainda não havia no País cursos formais de Informática de nível superior, os profissionais atuantes na área adquiriam competência técnica diretamente do exercício profissional, do autodidatismo, de cursos no exterior ou por treinamentos oferecidos pelos fabricantes dos computadores importados.

Somente no fim da década de 1960 é que foram criados os primeiros cursos formais em Computação no Brasil: na PUC-Rio, um curso de Mestrado em Informática, o primeiro curso de pós-graduação dessa área no Brasil, e na Universidade Federal da Bahia e na Universidade Estadual de Campinas, bacharelados em Ciência da Computação. A partir daí, outras universidades seguiram o exemplo e algumas dezenas de cursos de graduação e pós-graduação em Informática entraram em funcionamento na década de 1970.

Dessa forma, nessa década, a informática brasileira consolidou-se, atingindo um patamar de grande importância em seu desenvolvimento, tendo até provocado, mais tarde, a promulgação, pelo Congresso Nacional, de uma política industrial para o setor de informática, visando o atingimento de seu completo domínio tecnológico.

Esse rápido desenvolvimento criou uma demanda por profissionais qualificados para atuar em Tecnologia da Informação acima do que nossas universidades eram capazes de produzir, e, por isso, esse setor continuava dependente de um grande contingente de profissionais oriundos de áreas de domínios conexos.

Nesse contexto, a Sociedade Brasileira de Computação, com o objetivo de contribuir para melhorar a qualificação dos profissionais atuantes e abrir novas perspectivas para os nossos universitários e seus professores para o desenvolvimento de novas áreas de pesquisa e de atuação profissional, decidiu criar em 1982 a **Jornada de Atualização em Informática**, para ser um evento satélite dos congressos nacionais da SBC.



Os responsáveis por essa iniciativa foram os membros da então diretoria da SBC (1981-1983), composta pelo presidente Luiz de Castro Martins (PUC-Rio), vice-presidente Silvio Davi Paciornik (USP), secretária geral Sueli Mendes dos Santos (UFRJ), 1º secretário Estevam Gilberto de Simone (UFRJ), 2º secretário Ivan Moura Campos (UFMG) e tesoureira geral Therezinha da Costa Ferreira Chaves (PUC-Rio), sendo a proposta de criação da JAI aprovada pelo seu conselho, cujos membros eram Claudio Zamitti Mammana (USP), Clésio Saraiva dos Santos (UFRGS), Henrique Pacca Loureiro Luna (UFMG), Carlos Ignácio Zamitti Mammana (CTI/SEI), Mario Dias Ripper (CDS), Wilson de Pádua Paula Filho (UFMG), Ivan da Costa Marques (DIGIBRAS) e João Antonio Zuffo (USP).

A JAI foi idealizada para oferecer minicursos de quatro ou seis horas de duração cada um, abordando um leque variado de temas avançados, mas ao mesmo tempo consolidados, que normalmente não faziam parte dos currículos dos cursos de graduação em computação, na expectativa de despertar o interesse de seus profissionais, estudantes e professores.

A primeira JAI ocorreu como parte do II Congresso da Sociedade Brasileira de Computação, na Universidade Federal de Ouro Preto, cidade de Ouro Preto, MG, tendo sido composto de três cursos: Redes de Computadores, Construção de Compiladores e Tendências em Arquitetura de Computadores. As outras trilhas desse congresso foram o SEMISH e o SECOMU, sendo os temas desse SECOMU “A Formação de Recursos Humanos para o Parque Nacional de Informática” e “Política Nacional de Informática”. E do II CSBC, participaram cerca de 550 pessoas, inclusive representantes do Governo Federal.

A partir de então em todo congresso anual da SBC foram oferecidos de quatro a onze minicursos, de quatro ou seis horas de duração cada, com dezenas de alunos por curso, envolvendo, em média, de 10 a 20 professores de nossas melhores universidades. Esses números traduzem-se na oferta, nesses primeiros quarenta anos, de cerca de 250 cursos em muitas importantes áreas da Computação, envolvendo igual número de professores.

No início, os cursos eram mais informais, montados a partir de convites elaborados pelas coordenações dos eventos, mas, rapidamente, em decorrência do seu sucesso, passou-se a fazer chamadas de propostas de cursos a serem selecionados por comitês de programa e a exigir submissão de textos-didáticos de apoio às aulas, que seriam avaliados e aprovados por comitês editoriais compostos por especialistas nas áreas dos cursos pré-selecionados.

Esse processo tornou-se sistemático e deu origem em 2006 a uma série de livros da SBC, denominada **Atualizações em Informática**, que perdura até hoje, e na qual cada curso é apoiado por um capítulo de cerca de 40 páginas.

Os temas abordados pelas JAIs sempre foram na fronteira do conhecimento da época de sua oferta. No início dos tempos, o estudo de Compiladores, Redes e Arquitetura de Computadores estava sendo introduzido em nossos cursos de graduação, e assim atraíam uma grande audiência. Ao longo dos anos, os temas foram tornando-se mais específicos como sugerem os títulos dos seguintes cursos, selecionados dentre os que foram

ministrados nas três últimas ofertas:

- Programação de computadores quânticos
- Algoritmos de consenso e implementação na plataforma de corrente de blocos
- Fundamentos, aplicações e desafios na autenticação usando sinais biométricos
- Protocolos inovadores criados e adotados em escala mundial
- Conceitos e técnicas para coleta, armazenamento, tratamento e visualização de dados geoespaciais
- Avanços e desafios da computação social
- Métodos experimentais em interação humano computador
- Blockchain e contratos inteligentes para aplicações em IoT
- Privacidade de dados de localização
- Ciência de dados com reprodutibilidade.

Esses são apenas alguns bons exemplos de cursos cuja boa qualidade e atualidade dos temas abordados sempre foram reconhecidas pela comunidade da Computação. E isso justifica, que, no quadragésimo aniversário de sua criação, a Jornada de Atualização em Informática (JAI) seja vista como um dos mais importantes eventos acadêmicos de atualização científica e tecnológica da comunidade de Ciência da Computação do Brasil.

Parabéns à comunidade brasileira de computação por essa importante conquista! Parabéns aos nossos pioneiros, que há 40 anos, definiram o caminho que está sendo trilhado. E parabéns à SBC que garantiu a continuidade dessa iniciativa singular.

Para concluir, cumpre-nos informar que o levantamento dos dados a respeito das JAIs oferecidas nos últimos 40 anos não foi uma tarefa fácil, principalmente neste momento crítico da pandemia que assola todo o nosso planeta. Nessa busca pela recuperação desses dados, tivemos o apoio da sede da SBC, e, por essa ajuda, agradecemos à professora Renata Galante, diretora administrativa da SBC, que prontamente viabilizou nosso acesso direto a membros de sua equipe para realizar buscas e consultas aos arquivos da sede de nossa associação. E, em particular, agradecemos imensamente à Camila Luce, coordenadora de eventos da SBC, que, embora forçada a trabalhar em regime de trabalho remoto imposto pela pandemia do coronavírus, conseguiu reunir importantes informações e dados relevantes da história das JAIs, os quais tornaram possível a preparação deste breve relato histórico.

Desejamos a todos um proveitoso Congresso!

Roberto da Silva Bigonha  
Universidade Federal de Minas Gerais

## Sumário

<b>Capítulo 1 - Ciência de Dados com Reprodutibilidade usando Jupyter</b>	<b>13</b>
João F. Pimentel (UFF)	
Gabriel P. Oliveira (UFMG)	
Mariana O. Silva (UFMG)	
Danilo B. Seufitelli (UFMG)	
Mirella M. Moro (UFMG)	
<b>Capítulo 2 - Métodos Experimentais em Interação Humano Computador</b>	<b>63</b>
Carlos H. Morimoto (USP)	
Antonio Diaz-Tula (USP)	
<b>Capítulo 3 - Privacidade de Dados de Localização: Modelos, Técnicas e Mecanismos</b>	<b>107</b>
Javam C. Machado (UFC)	
Eduardo R. Duarte Neto (UFC)	
<b>Capítulo 4 - Blockchain e Contratos Inteligentes para Aplicações em IoT, uma Abordagem Prática</b>	<b>151</b>
Fabíola Greve (UFBA)	
Jauberth Abijaude (UESC e UFBA)	
Péricles Sobreira (UQO)	



## Capítulo

# 1

## Ciência de Dados com Reprodutibilidade usando Jupyter

João Felipe Pimentel, Gabriel P. Oliveira, Mariana O. Silva,  
Danilo B. Seufitelli e Mirella M. Moro

### *Abstract*

*Data Science has become a trending research topic in Computer Science due to the growing interest in extracting knowledge from different data sources. In such a context, Jupyter Notebook has consolidated itself as one of the main tools used by data scientists to perform exploratory data analysis in a fast and straightforward way, with a high potential for code reproduction. Hence, this JAI aims to present Jupyter with reproducibility for developing Data Science projects. The content is tailored for students and professionals with some programming experience. In particular, we first introduce Jupyter and its general use to develop solutions for Data Science. Then, we present Jupyter advanced topics and address ways to promote open science. Finally, this JAI overviews Data Science with Jupyter Notebooks by combining concepts and theoretical foundations with practical examples and real-world data.*

### *Resumo*

*Ciência de Dados tornou-se um tópico de pesquisa emergente na Ciência da Computação devido ao crescente interesse em extrair conhecimento de diferentes fontes de dados. Nesse contexto, o Jupyter Notebook vem se consolidando como uma das principais ferramentas utilizadas por cientistas de dados para realizar análises exploratórias de dados de forma rápida e direta, com alto potencial de reprodução de código. Dessa forma, o objetivo deste capítulo é apresentar o Jupyter com reprodutibilidade para a realização de projetos em Ciência de Dados. O conteúdo é organizado para estudantes e profissionais com alguma experiência em programação. Em particular, primeiro apresentamos o Jupyter e seu uso geral para desenvolver soluções para Ciência de Dados. Em seguida, apresentamos tópicos avançados do Jupyter e abordamos maneiras de promover a ciência aberta. Para concluir, este JAI apresenta uma visão geral de Ciência de Dados com Jupyter Notebooks combinando conceitos e fundamentos teóricos com exemplos práticos e dados do mundo real.*

## 1.1. Introdução

A Ciência de Dados tem como principal objetivo extrair informações úteis de dados. No processo de extração dessas informações, dados brutos com diversos formatos e estruturas são obtidos de diversas fontes, pré-processados para atingir um nível esperado de integridade, e finalmente analisados através da extração de estatísticas, visualizações e técnicas de mineração de dados e aprendizado de máquina para que as informações desejadas sejam obtidas. Todo esse processo deve ser feito com rigorosidade para que as informações sejam obtidas com qualidade e reprodutibilidade. Ou seja, após a extração das informações, é importante que seja possível reproduzir a análise e chegar a resultados próximos para dados semelhantes. Essa característica ajuda a validar que a informação extraída está correta e possibilita expandir a análise para novos dados e resultados.

O uso de Jupyter Notebooks para publicar pesquisas reprodutíveis é defendido por combinar texto de relatório com código executável [Kluyver et al. 2016]. Apesar do formato em si não garantir a reprodutibilidade [Pimentel et al. 2019], a documentação de processos com resultados pode ser bem valiosa para tal objetivo, e seu formato possibilita e facilita a análise exploratória de dados [Perkel 2018].

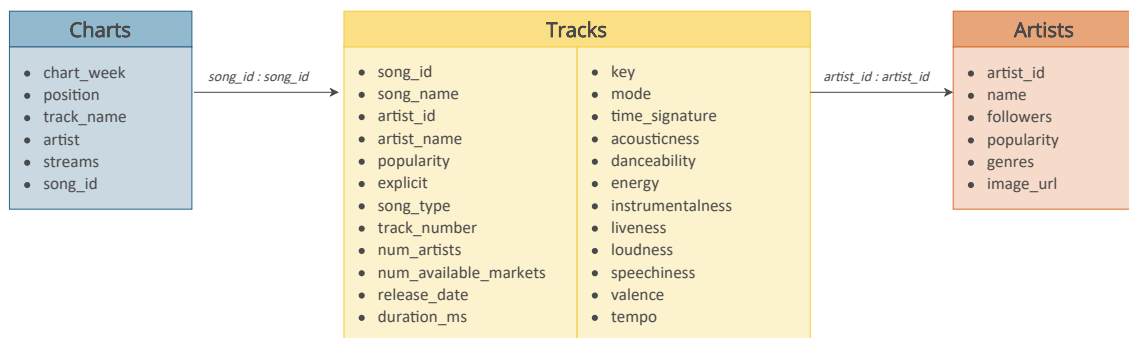
Este capítulo está organizado da seguinte forma. A Seção 1.2 introduz o conjunto de dados utilizado ao longo deste capítulo. A Seção 1.3 apresenta os conceitos iniciais de Jupyter. As Seções 1.4, 1.5 e 1.6 apresentam, respectivamente, a preparação de dados, métodos importantes e visualização do Jupyter para Ciência de Dados com Python. A Seção 1.7 resume aspectos avançados do Jupyter para a sua extensão. A Seção 1.8 apresenta como fazer ciência aberta com o Jupyter, compartilhando notebooks e resultados e garantindo a reprodutibilidade dos mesmos. Por fim, a Seção 1.9 apresenta considerações finais. A Figura 1.1 apresenta uma visão geral do capítulo com as ferramentas e conceitos que são discutidos em cada seção.



Figura 1.1. Conteúdo do curso.

## 1.2. Conjunto de Dados

Esta seção apresenta o conjunto de dados que guia o restante deste capítulo (Seções 1.3 a 1.7). Um conjunto único facilita o entendimento dos tópicos e reduz o tempo de familiarização com os dados em si. Usar dados reais em um projeto de Ciência de Dados também enriquece o aprendizado, pois dados sintéticos podem não ser fiéis aos desafios das etapas



**Figura 1.2. Esquema do conjunto de dados de sucesso musical utilizado como estudo de caso ao longo das Seções 1.3 a 1.7.**

de tratamento e processamento dos dados [Iguar and Seguí 2017, Skiena 2017]. Aqui, o conjunto de dados refere-se a sucesso na indústria da música, uma das mais dinâmicas e importantes no cenário do entretenimento mundial. Especificamente, utilizamos dados provenientes do Spotify,<sup>1</sup> o serviço de *streaming* de áudio mais popular do mundo, que reúne mais de 345 milhões de usuários em 178 países e territórios.<sup>2</sup>

A Figura 1.2 apresenta o esquema do conjunto de dados. Seguindo a metodologia de [Oliveira et al. 2020], os dados são obtidos a partir das paradas semanais de sucesso do Spotify durante 2020. Tais paradas são um *ranking* das 200 músicas mais ouvidas durante a semana em questão e são disponibilizadas separadamente para cada mercado em que a empresa atua. Porém, por simplificação, consideramos apenas os dados globais agregados, ou seja, obtidos somando-se os *streams* de todos os mercados. Para cada lista, obtém-se informação de suas músicas, incluindo lista de intérpretes, data de lançamento e características acústicas. Também inclui-se dados sobre artistas que interpretam tais músicas, destacando-se seus gêneros musicais e indicadores de popularidade.

### 1.3. Jupyter Básico

Jupyter é o sistema mais usado para programação literária interativa [Shen 2014]. O paradigma de programação literária busca ajudar na comunicação de programas através da alternância de texto em linguagem natural formatada, pedaços de código executáveis, e resultados de computações. O texto em linguagem natural é usado tanto para explicar o código quanto para comentar o resultado obtido [Perkel 2018]. A interatividade do Jupyter permite que este paradigma seja usado em tempo real para análises de dados, com o processo sendo documentado durante o desenvolvimento, resultados sendo exibidos de forma instantânea e discutidos imediatamente em linguagem natural.

Jupyter foi desenvolvido como uma evolução do IPython, um sistema de *REPL* (laço de leitura-avaliação-impressão) para Python [Perkel 2021b]. Apesar de suportar várias linguagens de programação, a linguagem para códigos executáveis mais comum em Notebooks (documentos do Jupyter) ainda é Python [Pimentel et al. 2019]. Python é uma linguagem de programação interpretada multiparadigma com suporte a programação

<sup>1</sup>Spotify API: <https://developer.spotify.com/>

<sup>2</sup>Spotify Company Info: <https://newsroom.spotify.com/company-info/>

imperativa, programação funcional e orientação a objetos. Inclui estruturas de dados e módulos em sua instalação, com flexibilidade para manipular as estruturas e integrar-se com ferramentas escritas em outras linguagens. Assim, cientistas de todas as áreas usam esta linguagem em seus experimentos computacionais [Perkel 2018, Pimentel 2021].

Esta seção tem o objetivo de apresentar a estrutura geral de um Notebook (Seção 1.3.1), bem como introduzir uma biblioteca básica amplamente utilizada em Jupyter Notebooks para projetos de Ciência de Dados (Seção 1.3.3). Além disso, abordamos como adicionar dados de diferentes formatos no Jupyter (Seção 1.3.2).

### 1.3.1. Estrutura Geral de um Notebook

Jupyter armazena Notebooks como documentos JSON com extensão “.ipynb” (para mais detalhes do que é um documento JSON, ver Seção 1.3.3). Esses documentos são compostos por *células* que podem ser de três tipos: uma célula de *código* possui código executável que produz resultados; uma célula de *Markdown* possui texto formatado; e uma célula *bruta* possui texto que não é nem Markdown, nem código executável. Em geral, células brutas são usadas por ferramentas externas que convertem notebooks em outros formatos.

Os resultados de uma execução de uma célula de código são armazenados na própria célula e exibidos interativamente durante a execução do notebook. Jupyter permite a visualização de textos (saída padrão), erros (saída de erros), imagens (PNG, JPG e SVG), HTML com JavaScript e Markdown. Além do resultado da execução, células de código também armazenam um contador de execução, que pode ser usado para indicar aproximadamente a ordem de execução. Por conta da interatividade do Jupyter, um notebook não precisa ser executado inteiramente de cima para baixo – como ocorre normalmente em arquivos de código. Ao invés disso, pode-se executar em qualquer ordem desejada e até mesmo executar uma mesma célula mais de uma vez.

A Figura 1.3 apresenta um exemplo de notebook executado, com duas células de

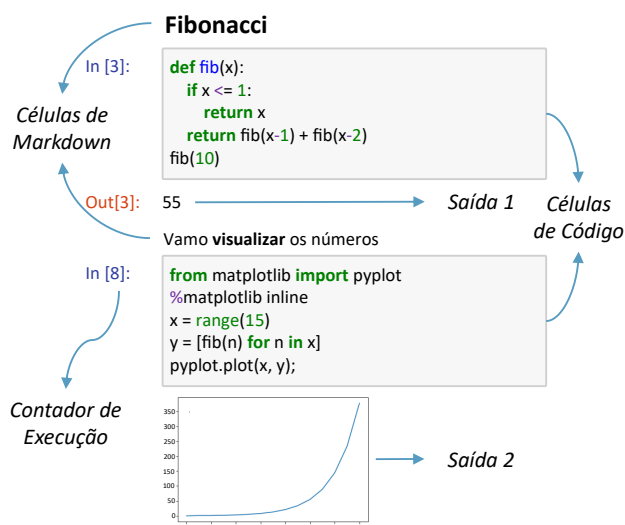


Figura 1.3. Exemplo de notebook executado com Markdown, código e resultados (adaptado de [Pimentel et al. 2019]).



Markdown e duas células de código. Na esquerda das células de código, há contadores de execução que indicam a ordem. Abaixo das células de código, Jupyter exibe os resultados. Perceba que a primeira célula de código retorna um número, identificado por **Out[3]**, e a segunda célula exibe uma imagem sem retorná-la. Nesta imagem, os contadores de execução marcam 3 e 8. Isso significa que ocorreram duas execuções antes da execução da primeira célula de código e quatro execuções entre a primeira célula de código e a segunda. Essas execuções não ficam armazenadas no notebook apesar de poderem contribuir para estado atual da execução.

### 1.3.2. Introdução ao *pandas*

A biblioteca *pandas*, cujo nome é derivado do termo inglês *Panel Data*, foi criada para a linguagem *Python* e é muito utilizada por cientistas de dados, pois fornece estruturas de dados de alto desempenho e ferramentas para análise de dados em formato tabular e de séries temporais, além de ser fácil de utilizar. Para isso, é necessário importá-la ao projeto incluindo um apelido para referenciá-la (e.g., *pd*), com o trecho de código:

```
import pandas as pd.
```

O principal recurso do *pandas* é o *DataFrame*, um objeto rápido e eficiente para manipulação de dados com indexação integrada. A Figura 1.4 apresenta um *DataFrame*, i.e., uma estrutura tabular com linhas e colunas. As linhas possuem um índice específico para acessá-las, que pode ser um nome ou um valor. Já as colunas, sendo chamadas de *Series*, são um tipo especial de dados que consiste numa lista de vários valores, onde cada valor possui um índice. Portanto, a estrutura de dados do *DataFrame* pode ser entendida como uma planilha, porém muito mais flexível.

*pandas* oferece muitas funções, incluindo para transformar (facilmente) qualquer conjunto de dados: adicionar ou remover linhas e colunas, redefinir índices, reordenar e renomear colunas. Também permite operações de álgebra relacional (e.g., projeção, junção e concatenação) e funções de limpeza (e.g., preenchimento, substituição ou inserção de valores nulos – *null*). O *pandas DataFrame* possui funções de alto desempenho para agregar, mesclar e juntar conjunto de dados em diferentes formatos: CSV, TXT, MS

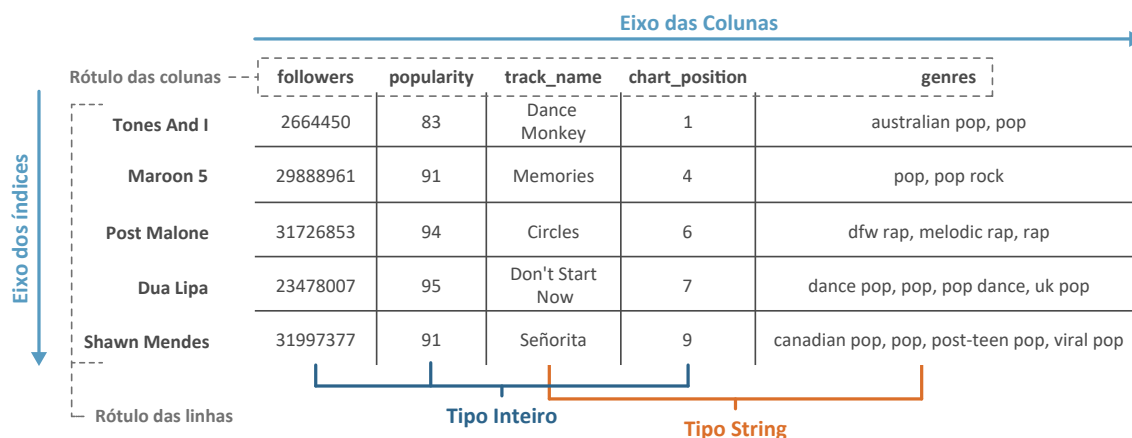


Figura 1.4. Estrutura tabular de um *DataFrame* e seus principais componentes.

Excel, SQL e HD5. Para dados incompletos ou não estruturados, oferece tratamento de dados ausentes e alinhamento inteligente de dados. Uma de suas vantagens é a excelente integração com diversas outras bibliotecas do Python para aprendizado de máquina (e.g., *scikitlearn*) e geração de visualizações (e.g., *matplotlib*).

A seguir, estão as principais ferramentas da biblioteca *pandas* alinhadas às necessidades de cientistas de dados: como utilizar as funções do *pandas* para criar e manipular *DataFrames* e *Series*; as principais funções que fornecem uma visão geral dos dados; e funções para manipular e ajustar os dados que facilitam projetos de Ciência de Dados.

### 1.3.2.1. Criar Objetos do Tipo *Series*

Um objeto do tipo *Series* é um *array* de uma dimensão e possui uma lista de valores de qualquer tipo de dados (e.g., strings, caracteres, booleanos, números e datas). No entanto, não é recomendado colocar tipos diferentes numa mesma *Series*, pois pode-se perder vantagens de desempenho. O exemplo a seguir cria uma *Series* em Python. Como resultado, é possível identificar os valores (bandas e artistas), o tipo (*object*) e os índices. Nota: por legibilidade, a saída dos comandos é mostrada ao lado (e não abaixo).

```
[1]: import pandas as pd

[2]: # Criando uma Série formada por nomes de bandas e cantores famosos
      artistas_1 = pd.Series(["Beatles", "Michael Jackson",
                             "Rihanna", "Skank"])
      artistas_1
[2]: 0         Beatles
      1    Michael Jackson
      2         Rihanna
      3           Skank
      dtype: object
```

Toda *Series* possui um índice (*index*) que rotula cada elemento e permite acessar seus valores. O exemplo anterior não tem índice específico, então o *pandas* cria um objeto do tipo *RangeIndex* de 0 até o número de elementos menos um. Por exemplo, `artistas[0]` retorna o valor “Beatles”. Pode-se criar um índice próprio que não precisa ser numérico e nem exclusivo como o próximo exemplo. Porém, esse recurso requer cuidado, pois a falta de exclusividade dos índices pode ocasionar problemas de manipulação.

```
[4]: # Serie com índices textuais: banda e artista
      artistas = pd.Series(["Beatles", "Michael Jackson", "Rihanna", "Skank"],
                           index=["banda", "artista", "artista", "banda"])
      artistas
[4]: banda         Beatles
      artista    Michael Jackson
      artista         Rihanna
      banda           Skank
      dtype: object

[5]: # Como os índices não são exclusivos, o código a seguir acessa os valores
      # de todas as 'bandas'
      artistas['banda']
[5]: banda         Beatles
      banda           Skank
      dtype: object
```

### 1.3.2.2. Criar Objetos do Tipo *DataFrames*

*DataFrames* são poderosas estruturas de dados tabulares bidimensionais com tamanho mutável e potencialmente heterogêneos, e são formados por um conjunto de *Series*. Ou

**Tabela 1.1. Principais funções de manipulação de estruturas e objetos do *pandas*.**

	Função	Descrição
Estruturas	<b>drop()</b>	Remove as linhas ou colunas de <i>Series/DataFrame</i> , especificado pelo índice ou pelo eixo. Retorna o objeto sem as linhas ou colunas especificadas.
	<b>df['Coluna'] = 'valor'</b>	Adiciona uma nova coluna em um <i>DataFrame</i> .
	<b>df.columns = ['coluna 1', ..., 'coluna n']</b>	Renomeia as n colunas de um <i>DataFrame</i> . O nome das novas n colunas deverá ser representada como uma lista.
	<b>append()</b>	Concatena duas ou mais <i>Series</i> ou <i>DataFrames</i> .
Objetos	<b>shape</b>	Retorna o número de linhas e/ou colunas de uma <i>Series</i> ou <i>DataFrame</i> .
	<b>index</b>	Informações dos índices dos objetos, tais como nomes e tipos.
	<b>columns</b>	Retorna o nome de todas as colunas de um <i>DataFrame</i> .
	<b>dtypes</b>	Retorna o nome e tipo de dados de todas as colunas de um <i>DataFrame</i> .

seja, as *Series* são colunas das tabelas *DataFrames*. Para criar um *DataFrame*, é necessário instanciar um objeto a partir da classe *DataFrame* do *pandas*. Como as *Series*, os *DataFrames* têm índices, mas também possuem colunas identificadas por nome. Se não explicitar os nomes das colunas, o *pandas* cria um *RangeIndex* de 0 até o número de colunas menos um. Pode-se criar um *DataFrame* de diversas maneiras. Os comandos a seguir mostram duas formas bastante comuns: a partir de uma lista de elementos; e a partir de um dicionário, onde a chave é o nome da coluna, e o valor, uma lista de objetos.

```
[6]: # Construindo um dataframe a partir de uma lista de elementos
df = pd.DataFrame([
    ["Coldplay", 29397183], ["Katy Perry", 17784767], ["Adele", 26296798]])
df
```

```
[6]:
```

	0	1
0	Coldplay	29397183
1	Katy Perry	17784767
2	Adele	26296798

```
[7]: # Construindo um dataframe a partir de um dicionário
data = {'Artista': ["Coldplay", "Katy Perry", "Adele"],
        'Followers': [29397183, 17784767, 26296798]}
df = pd.DataFrame(data)
df
```

```
[7]:
```

	Artista	Followers
0	Coldplay	29397183
1	Katy Perry	17784767
2	Adele	26296798

O acesso aos dados do *DataFrame* é feito de várias formas discutidas mais adiante. De modo simplificado, pode-se acessar uma coluna em *DataFrame* com o comando: **df['coluna']**. Sem índices, **df[0]** acessa os valores da primeira coluna; com índice, a mesma coluna é acessada utilizando **df['Artista']**.

### 1.3.2.3. Manipulação de *Series* e *DataFrames*

As funções para manipular *Series* e *DataFrames* são praticamente as mesmas. Porém, nem todas são aplicáveis a ambos objetos devido às limitações lógicas (e.g., funções para linhas e colunas não se aplicam a objetos *Series*). O *pandas* disponibiliza muitas funções em sua documentação.<sup>3</sup> A Tabela 1.1 destaca algumas das principais e suas aplicações. Algumas dessas são apresentados nas seções a seguir no contexto do estudo de caso.

<sup>3</sup>Link para acesso a documentação do *pandas*: <https://pandas.pydata.org/docs/>

### 1.3.3. Importação de Dados

Ciência de Dados extrai informações úteis dos dados. Porém, muitos projetos obrigam seus cientistas a reunir uma miscelânea de padrões de fontes de dados, seja em CSV, JSON, SQL, ou outro formato. Assim, é crucial saber lidar com os principais formatos de dados em Python, bem como realizar tal tarefa eficientemente; afinal, os dados são o ponto de partida de análises estatísticas, aplicação de técnicas de aprendizado de máquina, entre outras. O Python oferece bibliotecas e pacotes para os principais formatos *open data* e proprietários. Portanto, esta seção aborda a importação dos seguintes formatos de dados: CSV, TSV, XLS, JSON, SQL e XML. Embora não exaustiva, é raro um projeto de Ciência de Dados não lidar com ao menos um desses formatos.

**Arquivos Texto.** Arquivos de texto são amplamente utilizados em projetos de Ciência de Dados. Porém, existem formatos de arquivos textuais distintos, tais como CSV e TSV. Por definição, o CSV é um formato de arquivo tabular cujos valores são separados por vírgulas. A biblioteca *pandas* possui a função `read_csv()` para a leitura deste formato de arquivos. O exemplo a seguir mostra a importação de arquivos CSV em Python.

```
[2]: # A função read_csv é utilizada
# para ler arquivos texto
artistas = pd.read_csv(
    'spotify_artists.csv')
artistas
```

```
[2]:
```

	name	followers	popularity	genres
0	Coldplay	29397183	90.0	['permanent wave', 'pop']
1	Ninho	4239063	84.0	['french hip hop', 'pop urbaine']
2	KEVVO	167419	75.0	['perreo', 'reggaeton', 'reggaeton flow', 'tra...']
3	Olivia Rodrigo	1134117	89.0	['alt z', 'pop', 'post-teen pop']
4	Harry Styles	13439256	91.0	['pop', 'post-teen pop']

No entanto, os arquivos CSV demandam cuidado, pois geralmente causam conflitos devido à necessidade de cercar quebras de linha, aspas duplas e vírgulas no conteúdo de campos utilizando aspas duplas. Vírgulas literais são muito comuns em dados de texto, por exemplo, dados financeiros (R\$120,38). Nesse caso, em arquivos CSV, tais valores precisam ser escapados, normalmente utilizando aspas ("R\$120,38"). Outra opção é utilizar arquivos separados por outros delimitadores, tais como a tabulação.

Arquivos de valores separados por tabulação (TSV) também possuem formato de texto simples para armazenar dados em uma estrutura tabular (linhas e colunas). Analogamente, cada linha da tabela corresponde a uma linha do arquivo, e cada valor de campo de um registro (coluna) é separado por um caractere de tabulação (TAB). A biblioteca *pandas* oferece duas funções para a leitura de arquivos TSV: `read_table()` e `read_csv()` com `\t` de delimitador, como mostram os comandos a seguir.

```
[3]: # A função read_table é utilizada
# para ler arquivos .tsv
charts = pd.read_table(
    'spotify_charts.tsv')
charts
```

```
[3]:
```

	chart_week	position	track_name	artist	streams
0	2020-07-16	200	Como Llor	Juanfran	4534139
1	2020-07-23	1	ROCKSTAR (feat. Roddy Ricch)	DaBaby	35694103
2	2020-07-23	2	Savage Love (Laxed - Siren Beat)	Jawsh 685	33897676
3	2020-07-23	3	Blinding Lights	The Weeknd	30485774
4	2020-07-23	4	Watermelon Sugar	Harry Styles	26680752

```
[4]: # Alternativamente, pode-se utilizar o
# parâmetro delimiter para separar com \t
charts = pd.read_csv(
    'spotify_charts.csv', delimiter='\t')
charts
```

```
[4]:
```

	chart_week	position	track_name	artist	streams
0	2020-07-16	200	Como Llor	Juanfran	4534139
1	2020-07-23	1	ROCKSTAR (feat. Roddy Ricch)	DaBaby	35694103
2	2020-07-23	2	Savage Love (Laxed - Siren Beat)	Jawsh 685	33897676
3	2020-07-23	3	Blinding Lights	The Weeknd	30485774
4	2020-07-23	4	Watermelon Sugar	Harry Styles	26680752

Com TAB como separador entre os campos, um campo não pode conter um TAB. Porém, os TABs geralmente não aparecem nos itens de dados, portanto, isso raramente é uma restrição. TSV é um formato de arquivo simples com amplo suporte, muito usado para mover dados tabulares entre diferentes aplicações que oferecem suporte ao formato.

**Planilhas Excel.** A biblioteca *pandas* permite a importação de arquivos Excel (xls e xlsx) em Python. Similar aos exemplos anteriores, utiliza-se a função `read_excel('arquivo.xlsx')`. Observe que para versão anterior do Excel, pode ser necessário usar a extensão de arquivo 'xls'. Para a importação de uma planilha específica em um mesmo arquivo, é preciso utilizar o parâmetro `sheet_name`. Por exemplo, suponha que tenhamos um arquivo chamado 'dados.xlsx'. Neste arquivo, ente suas diversas abas, temos uma chamada *Streams*. Para importar apenas o conteúdo da aba *Streams* do arquivo dados.xlsx, programa-se: `df = pd.read_excel('dados.xlsx', sheet_name='Streams')`. Caso o `sheet_name` não seja especificado, importa-se a primeira aba.

**JavaScript Object Notation (JSON).** Um arquivo JSON armazena estruturas de dados simples e objetos no formato *JavaScript Object Notation*, um formato padrão de intercâmbio de dados. Esses arquivos são leves, textuais, legíveis por humanos e editáveis com editor de texto. Arquivos JSON representam dados com o conceito de chave e valor: cada valor tem uma chave que descreve seu significado. Por exemplo, a *chave:valor* `artista:'Michael Jackson'` representa o artista 'Michael Jackson'. Para importar arquivos JSON, o *pandas* tem a função `read_json()`, com funcionamento similar às anteriores.

**Extensible Markup Language (XML).** XML é uma linguagem de marcação para representar e distribuir dados semiestruturados. É semelhante ao JSON, porém os dados são organizados entre elementos (ou *tags*) que definem semântica ao valor, sempre com uma *tag* inicial e *tag* final (`<email>email@example.com</email>`). Arquivos XML são úteis para lidar com bases de dados pequenas, médias, ou em atividades que precisam de um esquema de dados flexível. Em Python, existem bibliotecas para importar e manipular XML, tais como a *DOM (Document Object Model)*, *SAX (Simple API for XML)* e *etree.ElementTree*. Esta última é mais comum, pois possui implementação simples e bom desempenho. O comando `parse()` faz a leitura do arquivo XML, e o comando `findall()` retorna os elementos de uma *tag* específica. Assim, é possível manipular os arquivos XML e construir *DataFrames* em formatos tabulares para utilizar todas as funções do *pandas*.

**Structured Query Language (SQL).** A importação de dados a partir de um banco de dados SQL requer etapas adicionais em comparação aos outros formatos, pois o *pandas* não suporta a integração com bancos de dados de forma nativa e depende de bibliotecas de terceiros para estabelecer a conexão. Assim, após a conexão com o banco de dados, é possível trabalhar diretamente com *pandas* usando a função `read_sql_query()`. Existem várias bibliotecas que fazem a integração do Python com os Sistemas Gerenciadores de Banco de Dados (SGBDs). Uma delas é a biblioteca *sqlalchemy*, responsável por criar uma *engine* de conexão com diferentes SGBDs, e.g., MySQL, Oracle, Postgres e Sqlite. Além da *engine*, é preciso importar um *driver* de conexão, o qual é específico para cada SGBD (e.g., para MySQL, usa-se o comando `!pip install pymysql`).

## 1.4. Preparação de Dados para Ciência

Em Ciência de Dados, dados de qualidade são pré-requisito para pesquisas válidas, descobertas significativas, modelos de Aprendizado de Máquina, entre outros. Porém, no mundo real, dados brutos costumam ser incompletos, ruidosos, inconsistentes e, às vezes, estão em formato inutilizável. Portanto, antes de alimentá-los a modelos (e outras etapas de pesquisa), é fundamental averiguar a integridade de dados e identificar possíveis problemas. Este processo é denominado pré-processamento de dados.

Essencialmente, preparar dados significa adequá-los para servirem de entrada nos processos da pesquisa. Existem muitas técnicas de pré-processamento que, geralmente, acontecem em etapas organizadas nas seguintes categorias: Limpeza de Dados, Integração de Dados, Transformação de Dados, e Redução de Dados. As etapas do pré-processamento não são mutuamente exclusivas e são altamente dependentes do conjunto de dados; ou seja, podem trabalhar em conjunto, mas não são obrigatórias.

Em particular, a Limpeza de dados pode remover ruído e corrigir inconsistências nos dados. A Integração de dados mescla dados de várias fontes em um armazenamento de dados coerente, como um armazém de dados. Transformações de dados, como normalização, podem melhorar a precisão e eficiência de algoritmos que envolvem medições de distância. Então, a Redução de dados pode diminuir o tamanho dos dados agregando ou eliminando recursos redundantes. Através de exemplos com dados reais, esta seção define e descreve cada uma dessas etapas técnicas.

### 1.4.1. Limpeza de Dados

A Limpeza de dados é o processo de detecção e correção de registros incorretos ou corrompidos. Refere-se à identificação e, em seguida, a substituição, modificação ou exclusão de partes incompletas, imprecisas ou irrelevantes. Em geral, a Limpeza de dados leva a uma sequência de tarefas que visam melhorar a qualidade dos dados. Algumas dessas tarefas incluem não só lidar com dados ausentes e duplicados, mas também remover dados ruidosos, inconsistentes e outliers. Esta seção apresenta cada uma dessas tarefas através de uma versão aleatoriamente modificada das tabelas do estudo de caso.

**Dados ausentes.** Representam um obstáculo para a criação da maioria dos modelos de Aprendizado de Máquina e outras análises. Portanto, é necessário identificar campos para os quais não há dados e, em seguida, compensá-los adequadamente. Dados ausentes podem ocorrer quando nenhuma informação é fornecida para um ou mais registros (ou atributos inteiros) da base de dados. Em um *pandas DataFrame*, os dados ausentes são representados como **None** ou **NaN** (*Not a Number*), embora **NaN** seja o marcador de valor ausente padrão por razões de velocidade e conveniência computacional.

Após importar pacotes necessários e carregar o conjunto de dados, inicia-se o processo de Limpeza de dados. Para facilitar a detecção, o *pandas* fornece a função **isna()** para identificar valores ausentes em um *DataFrame*. Esta função retorna uma matriz *booleana* indicando se cada elemento correspondente está faltando (**True**) ou não (**False**). O exemplo a seguir apresenta o uso desta função no *DataFrame* **df** e seleciona as três primeiras linhas por meio da função **head(3)**. Esta seleção foi feita neste capítulo por questões de legibilidade.

```
[3]: # Esta célula identifica valores ausentes no Dataframe
# e exibe as três primeiras linhas
pd.set_option('display.max_columns', 5)
df.isna().head(3)
```

```
[3]:
```

	artist_id	name	...	genres	image_url
0	False	False	...	False	False
1	False	False	...	False	False
2	False	False	...	False	False

3 rows x 6 columns

Segundo o exemplo anterior, o conjunto de dados está aparentemente completo. Porém, tal resposta não é suficiente para descartar a hipótese de que existem dados ausentes. Para uma melhor averiguação, pode-se resumir cada coluna no *DataFrame* booleano somando os valores **False=0** e **True=1**. Tal processo retorna o número total de valores ausentes. Também pode-se dividir cada valor pelo número total de linhas no conjunto de dados, resultando na porcentagem de tais ausências, conforme o exemplo a seguir.

```
[4]: # Calcula o total e a % de valores ausentes
num_ausentes = df.isna().sum()
porc_ausentes = df.isna().sum() * 100 / len(df)
# DataFrame com as informações computadas acima
df_ausentes = pd.DataFrame({
    'Coluna': df.columns,
    'Dados ausentes': num_ausentes,
    'Porcentagem': porc_ausentes
})
df_ausentes
```

```
[4]:
```

	Coluna	Dados ausentes	Porcentagem
artist_id	artist_id	0	0.00
name	name	0	0.00
followers	followers	0	0.00
popularity	popularity	62	9.92
genres	genres	40	6.40
image_url	image_url	10	1.60

O *DataFrame* resultante contém 62 popularidades, 40 listas de gêneros e dez url de imagens ausentes, resultando em 9,9%, 6,4% e 1,6% de registros de cada coluna, respectivamente. Após essa identificação, é necessário tratar esses dados. A abordagem mais simples é eliminar todos os registros que contenham valores ausentes. No *pandas*, o método **dropna()** permite analisar e descartar linhas/colunas com valores nulos. O parâmetro **axis** determina a dimensão em que a função atuará: **axis = 0** remove todas as linhas que contêm valores nulos, e **axis = 1** remove colunas, conforme o seguinte exemplo.

```
[5]: # Elimina linhas com valores ausentes
novo_df = df.dropna(axis=0)
print(f"""\
Nº de linhas do DF original: {len(df)}
Nº de linhas do DF novo: {len(novo_df)}
Nº de linhas com pelo menos 1 valor ausente: {
(len(df) - len(novo_df))}""")
```

Nº de linhas do DF original: 625  
 Nº de linhas do DF novo: 529  
 Nº de linhas com pelo menos 1 valor ausente: 96

```
[6]: # Eliminando colunas com valores ausentes
novo_df = df.dropna(axis=1)
print(f"""\
Nº de colunas do DF original: {len(df.columns)}
Nº de colunas do DF novo: {len(novo_df.columns)}
Nº de colunas com pelo menos 1 valor ausente: {
(len(df.columns) - len(novo_df.columns))}""")
```

Nº de colunas do DF original: 6  
 Nº de colunas do DF novo: 3  
 Nº de colunas com pelo menos 1 valor ausente: 3

Os exemplos anteriores removeram as linhas/colunas onde pelo menos um elemento está faltando. Ambas abordagens são particularmente vantajosas para amostras de grande volume de dados, onde os valores podem ser descartados sem distorcer significativamente a interpretação. No entanto, apesar de ser uma solução simples, ela ainda apresenta o risco de perder dados potencialmente úteis.

Uma alternativa mais confiável para lidar com dados ausentes é a imputação. Em vez de descartar tais dados, a imputação procura substituir seus valores por outros. Nessa abordagem, os valores ausentes são inferidos a partir dos dados existentes. Existem várias maneiras de imputar os dados, sendo a imputação por valor constante ou por estatísticas básicas (média, mediana ou moda) as mais simples. No exemplo a seguir, os valores de popularidade ausentes são substituídos pela média das popularidades existentes por meio da função **fillna**.

```
[7]: # Substituindo NaNs pela média de valores presentes
copia_df = df.copy()
copia_df['popularity'].fillna(copia_df['popularity'].mean(), inplace=True)
copia_df.sample(2)
```

```
[7]:
```

	artist_id	name	...	genres	image_url
376	2kqUKsTuEj1lPbm6BSn1AU	Rich Music LTD	...	['latin', 'reggaeton', 'trap latino']	https://i.scdn.co/image/7eff816abb6777c0d9efdd...
600	5OdEbQJ3yBnc3gsIASAT5	G Herbo	...	['chicago drill', 'chicago rap', 'drill', 'rap...']	https://i.scdn.co/image/8363009cbb612741bfc343...

2 rows x 6 columns

Também existem várias técnicas de imputação avançadas cuja escolha depende da utilização dos dados, por exemplo, depende de um modelo de aprendizado de máquina para inserir e avaliar com precisão os dados ausentes. A imputação múltipla e modelos preditivos podem ser mais precisos, e assim são mais comuns do que métodos mais simples. No entanto, não existe uma maneira ideal de compensar a ausência, pois cada estratégia possui um desempenho melhor ou pior dependendo do conjunto de dados.

**Dados ruidosos.** São dados que fornecem informações adicionais, mas sem sentido, chamados de ruído. Geralmente são gerados por alguma falha na coleta de dados, erros de entrada de dados, entre outros. Dados com ruído podem prejudicar resultados de análises e de modelos, como os de aprendizado de máquina, por exemplo. Tal problema pode ser solucionado a partir de diferentes abordagens, incluindo o método de Binning, Regressão e algoritmos de agrupamento de dados (*Clustering*) [Igal and Seguí 2017, Skiena 2017].

Para reduzir os efeitos de pequenos erros de observação, utiliza-se o método de Binning, que é uma técnica de suavização de dados. Os dados originais são divididos em segmentos de tamanhos iguais (*bins*) e, em seguida, são substituídos por um valor geral calculado para cada intervalo. Cada segmento é tratado separadamente, onde a substituição de valores pode ser realizada através de valores médios ou limites. No *pandas*, o método Binning usa as funções `qcut()` e `cut()`, que parecem iguais, mas são diferentes.

De acordo com a documentação do *pandas*, `qcut()` é uma função de discretização baseada em quantis: ela procura dividir os dados em *bins* usando percentis com base na distribuição da amostra. A maneira mais simples de usá-la é definir o número de quantis e deixar que o *pandas* descubra como dividir os dados. O exemplo a seguir discretiza a variável `followers` de duas maneiras diferentes: criando cinco *bins* de mesmo tamanho, e configurando três quantis rotulados como “alto”, “médio” e “baixo”.

```
[8]: # Discretizando em 5 intervalos de tamanhos iguais
df['qcut_1'] = pd.qcut(df['followers'], q=5)
# Discretizando usando três quantis
df['qcut_2'] = pd.qcut(df['followers'],
q=[0,.3,.7,1], labels=["baixo", "médio", "alto"])
df.head(3)
```

```
[8]:
```

	artist_id	name	...	qcut_1	qcut_2
0	4gzpq5DPGxSnKTe4SA8HAU	Coldplay	...	(6020134.2, 77681514.0]	alto
1	6Te49r3A6f5BilgBRxH7FH	Ninho	...	(2373906.0, 6020134.2]	alto
2	4QrBoWLM2WNIPdbFhmlaUZ	KEVVO	...	(134539.2, 790144.4]	baixo

3 rows x 8 columns

Por outro lado, pode-se utilizar a função `cut()` para segmentar e ordenar os dados em *bins*. Enquanto `qcut()` calcula o tamanho de cada *bin*, garantindo que a distribuição dos dados nos compartimentos seja igual, a função `cut()` define bordas exatas dos compartimentos. Ou seja, não há garantia sobre a distribuição de itens em cada *bin*. O exemplo a seguir corta os dados da variável `followers` em quatro *bins* de tamanhos iguais. Desta forma, se você deseja uma distribuição igual dos valores em cada compartimento, use `qcut()`. Caso contrário, se você quiser definir seus próprios intervalos numéricos de categorias, use a função `cut()`.



```
[9]: # Discretizando em 4 bins
df['cut_1'] = pd.cut(
    df['followers'],
    bins=4)
df.head(3)
```

```
[9]:
```

	artist_id	name	...	qcut_2	cut_1
0	4gzpq5DPGxSnKTe4SA8HAU	Coldplay	...	alto	(19420416.75, 38840782.5]
1	6Te49r3A6f5BilgBRxH7FH	Ninho	...	alto	(-77630.463, 19420416.75]
2	4QrBoWm2WNIPdbFhmlaUZ	KEVVO	...	baixo	(-77630.463, 19420416.75]

3 rows x 9 columns

**Outliers.** São amostras de dados notoriamente diferentes da tendência central. São geralmente criados por erros de coleta ou entrada de dados, e podem facilmente produzir valores discrepantes interferindo na qualidade de análises. A maneira mais simples de identificar outliers é observar os valores máximos e mínimos em cada variável para ver se eles estão muito fora da curva normal. O exemplo a seguir utiliza a função `describe()` para gerar estatísticas do conjunto de dados, incluindo os valores máximos e mínimos.

```
[12]: pd.set_option('display.max_columns', 10)
df.describe().round().transpose()
```

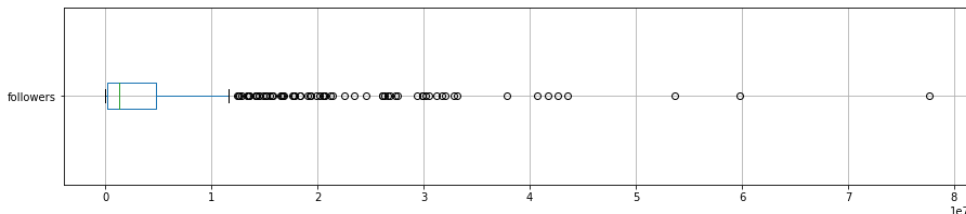
```
[12]:
```

	count	mean	std	min	25%	50%	75%	max
followers	625.0	4471046.0	8194928.0	51.0	222726.0	1256547.0	4809711.0	77681514.0
popularity	563.0	78.0	10.0	0.0	72.0	79.0	84.0	100.0

Para o recurso `followers`, o valor máximo é 77,7 milhões de seguidores, enquanto o quartil de 75% é apenas 48 milhões. Portanto, artistas com mais de 77 milhões de seguidores podem ser outliers. Essa verificação geral é melhor realizada através da representação gráfica dos dados numéricos através de seus quartis. Para isso, pode-se utilizar *boxplots*,<sup>4</sup> onde os valores discrepantes são plotados como pontos individuais. O gráfico do exemplo a seguir mostra a distribuição da variável `followers`.

```
[13]: df.boxplot(column=['followers'], figsize=(15, 3), vert=False)
```

```
[13]: <AxesSubplot:>
```



Esse exemplo ilustra que existem inúmeros pontos (outliers) entre aproximadamente 12 a 78 milhões (observe que o eixo x está em dezenas de milhões). Embora tenha sido fácil detectar tais valores discrepantes, é preciso determinar as soluções adequadas para tratá-los. Assim como no caso de dados ausentes, o tratamento de outliers depende muito do conjunto de dados e do objetivo do projeto. Soluções possíveis incluem manter, ajustar ou apenas remover os dados discrepantes.

Uma técnica comum para a remoção de outliers é o método de Z-score, que considera como outliers e remove valores a uma determinada quantidade de desvios padrões da média. A quantidade desses desvios pode variar conforme o tamanho da amostra. No exemplo a seguir, para identificar e remover os outliers da coluna `followers`, usou-se os z-scores de seus registros com a quantidade de desvios configurada para três. Para a obtenção dos z-scores, foi usado o módulo `stats` da biblioteca *SciPy*.

<sup>4</sup>Descrevemos com detalhes o que são *boxplots* na Seção 1.5

```
[14]: import numpy as np
from scipy import stats
z_scores = stats.zscore(df['followers']) # Z-scores dos valores da coluna 'followers'
abs_z_scores = np.abs(z_scores) # Obtendo os valores absolutos
novo_df = df[abs_z_scores < 3] # Filtra por desvios padrões < 3
print(f'{(len(df) - len(novo_df))} foram outliers removidos')
```

18 foram outliers removidos

**Dados duplicados.** Aparecem em muitos contextos, especialmente durante a entrada ou coleta de dados. Por exemplo, ao usar um *web scraper*, ele pode coletar a mesma página web mais de uma vez ou as mesmas informações de duas páginas diferentes. Independente da causa, a duplicação de dados pode levar a conclusões incorretas, onde algumas observações podem ser consideradas mais comuns do que realmente são. O exemplo a seguir mostra quantas linhas estão duplicadas em cada coluna do conjunto de dados.

```
[15]: # Para cada coluna,
for coluna in df.columns:
    # Seleciona linhas duplicadas
    duplicatas_df = df[df.duplicated(coluna)]

    print(f"Total de linhas duplicadas "
          f"na {coluna}: {len(duplicatas_df)}")
```

Total de linhas duplicadas na artist\_id: 0  
 Total de linhas duplicadas na name: 1  
 Total de linhas duplicadas na followers: 0  
 Total de linhas duplicadas na popularity: 569  
 Total de linhas duplicadas na genres: 136  
 Total de linhas duplicadas na image\_url: 10  
 Total de linhas duplicadas na qcut\_1: 620  
 Total de linhas duplicadas na qcut\_2: 622  
 Total de linhas duplicadas na cut\_1: 621

No exemplo, apenas duas colunas do *DataFrame* não possuem duplicatas: **artist\_id** e **followers**. Além disso, nota-se que existem duas cópias do nome de um mesmo artista. Essa duplicidade de dados é a categoria mais simples de duplicatas: são cópias exatamente iguais de um mesmo registro. Para resolver, basta identificar os valores idênticos e removê-los. O *pandas* fornece o método **drop\_duplicates()** que retorna um novo *DataFrame* com linhas duplicadas removidas, como no exemplo a seguir.

```
[16]: novo_df = df.drop_duplicates() # Remove as linhas duplicadas
duplicatas_df = novo_df[novo_df.duplicated()] # calcula o total de linhas duplicadas
print(f"Total de linhas duplicadas: {len(duplicatas_df)}")
```

Total de linhas duplicadas: 0

Com apenas uma lista de registros com duplicatas, a melhor e mais simples solução é geralmente a remoção. Porém, com dados tabulares, a melhor solução é remover os dados duplicados considerando os identificadores exclusivos. Por exemplo, **artist\_id** é a coluna de identificadores únicos, facilitando verificar se o registro duplo pode ser descartado, conforme o seguinte.

```
[17]: pd.set_option('display.max_columns', 5)
# Extraí o nome duplicado
nome_duplicado = df[df.duplicated(['name'])].name

# Linhas onde 'name' é igual ao nome duplicado
df.loc[df['name'].isin(nome_duplicado)]
```

```
[17]:
```

	artist_id	name	...	qcut_2	cut_1
88	30vzHJVtI7JW	Niack	...	baixo	(-77630.463, 19420416.75]
249	5uYe4bcAXIMP	Niack	...	baixo	(-77630.463, 19420416.75]

2 rows x 9 columns

Os resultados mostram dois artistas de nome “Niack” com identificadores diferentes; logo, não descartáveis. Existem outras formas complexas, onde mais de um registro é associado à mesma observação, porém seus valores não são completamente idênticos.

Por exemplo, nomes próprios com e sem abreviação ou omissão de algum dos sobrenomes. Essa duplicação parcial é mais difícil de identificar, e uma solução comum é utilizar funções de similaridade de strings.

A biblioteca Python *FuzzyWuzzy* usa a distância de Levenshtein para calcular as diferenças entre duas strings. Essa distância é calculada a partir da contagem de operações de inserção, remoção ou substituição necessárias para transformar uma string em outra. A seguir, as funções `ratio()` e `partial_ratio()` encontram cópias não idênticas de nomes de artistas. O código retorna duas prováveis duplicações parciais: na primeira, os dois nomes identificam duas artistas distintas, não se podendo removê-las; na segunda, “Red Velvet” denomina um grupo feminino sul-coreano, mas o nome “Red Velvet - Irene & Seulgi” foi cadastrado na plataforma Spotify para representar a primeira subunidade do grupo (composto por Irene & Seulgi).

```
[18]: from fuzzywuzzy import fuzz
      from itertools import combinations

      combinacoes = combinations(df.name, 2) # gera todas as combinações
      for nome_1, nome_2 in list(combinacoes): # para cada tupla de nomes,
          partial_ratio = fuzz.partial_ratio(nome_1, nome_2) # similaridade parcial
          ratio = fuzz.ratio(nome_1, nome_2) # similaridade simples

      # Se os nomes forem parcialmente iguais, porém não idênticos,
      if partial_ratio == 100 and ratio < 100 and ratio > 50:
          print(f'{nome_1} ({partial_ratio}) | {nome_2} ({ratio})')
```

```
The Blessed Madonna (100) | Madonna (54)
Red Velvet - IRENE & SEULGI (100) | Red Velvet (54)
```

## 1.4.2. Integração de Dados

Geralmente é necessário extrair dados a partir de várias fontes. A etapa seguinte do pré-processamento é combinar dados dessas diferentes fontes para obter uma estrutura unificada com informações mais significativas e valiosas. Por exemplo, dados básicos das músicas estão na tabela *M*, e os detalhes de tais músicas na tabela *D*. Então, uma análise mais robusta sobre as músicas requer reunir todos os dados possíveis de *M* e *D* em um único lugar, processo denominado Integração de dados.

O *pandas* permite mesclar e juntar múltiplas tabelas utilizando operações de junção. Para exemplificar, dividimos a tabela *Hits* em dois *DataFrames*, `df1` com título da música, identificador e nome de seus artistas, e `df2` com popularidade e data de lançamento das músicas. Em ambas as tabelas, a coluna `song_id` está presente, para identificar cada uma das músicas, conforme a seguir.

df1	song_id	song_name	artist_id	artist_name	df2	song_id	popularity	release_date
0	2rRjJJEo19S2J82BDsQ3F7	Falling	[7uaIm6Pw7xpIS8Dy06V6pT]	[Trevor Daniel]	0	2rRjJJEo19S2J82BDsQ3F7	77	2020-03-26
1	3BYIzNZ3i9IRQCACXSMLrT	Venetia	[4O15NlyKLIASxsJ0PrXPfz]	[Lil Uzi Vert]	1	3BYIzNZ3i9IRQCACXSMLrT	66	2020-03-06
2	1g3J9W88hTG173ySZR6E9S	Tilidin Weg	[1aS5iqEs9ci5P9KD9iZWa6]	[Bonez MC]	2	1g3J9W88hTG173ySZR6E9S	13	2020-07-30

A função `merge()` implementa vários tipos de junção: um-para-um (1:1), um-para-muitos (1:N) e muitos-para-muitos (N:N). O tipo de junção depende da organização dos conjuntos de dados de entrada.

**Junções um-para-um (1:1).** A junção um-para-um é talvez o tipo mais simples de fusão, muito semelhante à concatenação de colunas. Por exemplo, considere os dois *DataFrames*

**df1** e **df2**. O resultado da seguinte junção é um *DataFrame* que combina as duas entradas, baseado nos valores comuns da coluna **song\_id**. Observe que a ordem das entradas em cada coluna não é necessariamente mantida. Ou seja, a ordem da coluna **song\_id** difere entre as duas tabelas, e a função **merge()** as considera corretamente.

```
[5]: # Junção um-para-um
df3 = pd.merge(
    df1, df2, on='song_id')
df3.head(3)
```

```
[5]:
```

	song_id	song_name	artist_id	artist_name	popularity	release_date
0	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	['Trevor Daniel']	77	2020-03-26
1	3BYIzNZ3I9RQCACXSMLRT	Venetia	[4O15NlyKLIASxsJ0PrXPfz]	['Lil Uzi Vert']	66	2020-03-06
2	1g3J9W88hTG173ySZR6E9S	Tilidin Weg	[1aSSiqEs9ci5P9KD9tZWa6]	['Bonz MC']	13	2020-07-30

**Junções um-para-muitos (1:N)**. A junção um-para-muitos é usada quando uma das duas colunas-chave contém mais de uma entrada por registro. Nesse caso, o *DataFrame* resultante preserva tais entradas duplicadas conforme apropriado. Por exemplo, considere a junção da tabela resultante do exemplo anterior com os sucessos do Spotify na tabela *Charts*, na qual os identificadores das músicas podem aparecer mais de uma vez. O resultado do comando a seguir é um único *DataFrame* que combina as duas entradas; porém, ao contrário do exemplo anterior, os dados originais (i.e., do *DataFrame* **df3**) se repetem conforme exigido pelas entradas do *DataFrame* **df4**.

```
[7]: df4.head(3)
```

```
[7]:
```

	chart_week	position	song_id
0	2020-01-02	1	1rgnBhdG2JDF TbYkYRZAku
1	2020-01-02	2	696DnlkuDOXc MAnkITgXXK
2	2020-01-02	3	7k4I7uLgtOxP wTpFmJNTY

```
[8]: # Junção um-para-muitos de dois DataFrames
df5 = pd.merge(df3, df4, on='song_id')
df5.head(3)
```

```
[8]:
```

	song_id	song_name	artist_id	...	release_date	chart_week	position
0	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	...	2020-03-26	2020-03-26	11
1	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	...	2020-03-26	2020-04-02	13
2	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	...	2020-03-26	2020-04-09	12

3 rows x 8 columns

**Junções muitos-para-muitos (N:N)**. Na junção muitos-para-muitos, se a coluna-chave das duas tabelas incluir entradas duplicadas, o resultado é uma fusão muitos-para-muitos. Por exemplo, para criar um relacionamento muitos-para-muitos entre a tabela anterior e outra contendo o número total de streams que cada música atingiu nas semanas dos charts. A coluna-chave do *DataFrame* resultante é a combinação das colunas-chave de cada tabela. Agora, as duas tabelas terão duas colunas-chave: **song\_id** e **chart\_week**. Após a junção muitos-para-muitos, a coluna-chave do *DataFrame* resultante é expressa pela combinação dos valores de **song\_id** e **chart\_week** das tabelas **df5** e **df6**.

```
[10]: df6.head(3)
```

```
[10]:
```

	chart_week	streams	song_id
0	2020-01-02	50183626	1rgnBhdG2JDF TbYkYRZAku
1	2020-01-02	33254585	696DnlkuDOXc MAnkITgXXK
2	2020-01-02	29349573	7k4I7uLgtOxP wTpFmJNTY

```
[11]: # Junção muitos-para-muitos de dois DataFrames
df7 = pd.merge(df5, df6, on=['song_id', 'chart_week'])
df7.head(3)
```

```
[11]:
```

	song_id	song_name	artist_id	...	chart_week	position	streams
0	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	...	2020-03-26	11	21964590
1	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	...	2020-04-02	13	21724066
2	2rRJRjEo19S2J82BDsQ3F7	Falling	[7ualm6Pw7xplS8Dy06V6pT]	...	2020-04-09	12	21126685

3 rows x 9 columns

### 1.4.3. Transformação de Dados

Em geral, após a Integração de dados, colunas com diferentes tipos de dados podem ser geradas. Na Transformação de dados, o principal objetivo é transformar tais dados de diferentes formatos em um formato suportado pelo processo de pesquisa, e.g., modelos e algoritmos. A Transformação de dados possui etapas em comum com a Limpeza de

dados, incluindo técnicas de remoção de ruído e discretização de dados. Assim, a seguir, apresentamos as demais etapas envolvidas nesta fase de pré-processamento.

**Dados não padronizados.** Dados coletados de diferentes fontes podem reunir dados heterogêneos, não padronizados e em diferentes escalas. O reescalonamento de dados não é uma etapa obrigatória, mas uma boa prática, pois atributos em um padrão auxiliam o desempenho de modelos, algoritmos e afins. Existem duas técnicas de reescalonamento que transformam características para uma escala, magnitude ou intervalo. A seguir, descrevemos e exemplificamos cada uma delas utilizando o conjunto de atributos numéricos que descrevem as músicas do Spotify.

**Normalização de dados** reescalona atributos numéricos em um intervalo de 0 a 1, alterando os valores das colunas numéricas, sem distorcer as diferenças nos intervalos de valores. No exemplo a seguir, subtrai-se o valor mínimo de cada coluna e divide-se pelo intervalo, aplicando a escala *min-max* do *pandas* através dos métodos `min()` e `max()`.

```
[3]: df_norm = df.copy() # cópia do DataFrame
for coluna in df_norm.columns: # Para cada coluna,
    df_norm[coluna] = ( # Normalização min-max
        df_norm[coluna] - df_norm[coluna].min()
    ) / (df_norm[coluna].max() - df_norm[coluna].min())
df_norm.head(3)
```

```
[3]:
```

	popularity	track_number	num_artists	...	speechiness	valence	tempo
0	0.793814	0.310345	0.0	...	0.015335	0.212314	0.506887
1	0.680412	0.448276	0.0	...	0.176348	0.558386	0.606828
2	0.134021	0.000000	0.0	...	0.243727	0.143312	0.393380

3 rows x 17 columns

**Padronização de dados** reescalona a distribuição de cada atributo para média igual a zero e desvio padrão igual a um. Cada valor padronizado é calculado subtraindo a média do recurso correspondente e dividindo pelo desvio padrão. O exemplo a seguir transforma os atributos do *DataFrame*, subtraindo a média de cada coluna e dividindo pelo desvio padrão, usando os métodos `mean()` e `std()` do *pandas*.

```
[4]: df_padron = df.copy() # cópia do DataFrame
for coluna in df_padron.columns: # Para cada coluna,
    df_padron[coluna] = ( # Padronização
        df_padron[coluna] - df_padron[coluna].mean()
    ) / (df_padron[coluna].std())
df_padron.head(3)
```

```
[4]:
```

	popularity	track_number	num_artists	...	speechiness	valence	tempo
0	0.448125	1.007112	-0.644346	...	-0.802165	-1.217354	0.122480
1	-0.164943	1.806576	-0.644346	...	0.462704	0.221431	0.658761
2	-3.118819	-0.791681	-0.644346	...	0.992015	-1.504228	-0.486598

3 rows x 17 columns

**Engenharia de Características.** (*Feature Engineering*) Consiste em criar ou remover características em um conjunto de dados para melhorar o desempenho de modelos de Aprendizado de Máquina, por exemplo. Muitas técnicas de Transformação e Limpeza de dados também são *Feature Engineering*, incluindo métodos de reescalonamento, discretização e redução de dimensionalidade. Então, a seguir, discutimos duas técnicas ainda não abordadas neste capítulo.

**Dados categóricos** são variáveis que só podem assumir um número limitado e, geralmente, fixo de valores possíveis. Por exemplo, as músicas do Spotify contêm dados categóricos para informar se as músicas são explícitas. Ou seja, cada unidade de observação (i.e., música) é atribuída a um determinado grupo ou categoria nominal. Muitos algoritmos e modelos comuns em Ciência de Dados (e.g., aprendizado de máquina) têm dificuldades em processar dados categóricos, necessitando alguma forma de conversão para valores numéricos. Uma das abordagens mais simples e comuns é converter variáveis categóricas em *dummies* ou indicadores, utilizando a função `get_dummies()` do *pandas*, conforme o exemplo a seguir.

```
[7]: # Convertendo em dummies
df_dummies = pd.get_dummies(df)
df_dummies.head()

[8]: df_dummies.dtypes

[7]:
```

	explicit_False	explicit_True	song_type_Collaboration	song_type_Solo
0	1	0	0	1
1	0	1	0	1
2	1	0	0	1
3	1	0	0	1
4	0	1	1	0

```
[8]: explicit_False      uint8
explicit_True          uint8
song_type_Collaboration  uint8
song_type_Solo         uint8
dtype: object
```

**Variáveis de data** são um tipo especial de dados categóricos. Embora uma data forneça apenas um momento específico no calendário, quando pré-processada corretamente, ela pode enriquecer muito o conjunto de dados. Os exemplos a seguir convertem datas em formatos amigáveis, extraindo recursos e criando novas variáveis que podem ser usadas na análise de um modelo. Especificamente, extraímos recursos da data de lançamento das músicas do Spotify, utilizando métodos do *pandas*.

```
[10]: # Convertendo a data de lançamento em 'datetime'
data['release_date'] = pd.to_datetime(
    data['release_date'])
data.dtypes

[11]: data['day'] = data['release_date'].dt.day # dia
data['month'] = data['release_date'].dt.month # mês
data['year'] = data['release_date'].dt.year # ano
data.head(3)
```

```
[10]: song_id      object
song_name      object
release_date    datetime64[ns]
dtype: object

[11]:
```

	song_id	song_name	release_date	day	month	year
0	2rRjJJe019S2J82BDsQ3F7	Falling	2020-03-26	26	3	2020
1	3BYLzNZ39lRQCACXSMLrT	Venetia	2020-03-06	6	3	2020
2	1g3J9W88hTG173ySZR6E9S	Tilidin Weg	2020-07-30	30	7	2020

**Dados desbalanceados.** São uma ocorrência comum em domínios reais, especialmente em modelos de classificação. Dados estão desbalanceados quando existe uma desproporção de observações de cada classe. Tal desbalanceamento não gera um erro imediato ao construir e executar um modelo. Porém, os resultados podem ser ilusórios, dado que a maioria das técnicas de Aprendizado de Máquina, por exemplo, funcionam melhor quando o número de amostras em cada classe é equilibrado. O exemplo a seguir usa características acústicas das músicas do Spotify para prever se uma música é uma colaboração ou não. Aqui, modelamos o problema como uma classificação binária: cada música recebe o valor 1 se for uma colaboração, ou 0 caso contrário.

```
[13]: data.head(3)
```

```
[13]:
```

	song_type	duration_ms	key	...	speechiness	valence	tempo
0	Solo	159381	10	...	0.0364	0.236	127.087
1	Solo	188800	9	...	0.1750	0.562	142.933
2	Solo	180950	10	...	0.2330	0.171	109.090

3 rows x 14 columns

```
[14]: data['song_type'] = [int(x == 'Collaboration') for x in data.song_type] # Mapeia 'song_type' em 0 e 1
data['song_type'].value_counts() # qtd de cada classe

[14]: 0      752
1      532
Name: song_type, dtype: int64
```

A função `value_counts()` computa os valores únicos de cada classe. Como resultado, apenas cerca de 41,4% das observações (i.e., 532 músicas) são colaborações. Portanto, ao prever a classe 0, músicas solo, obtém-se precisão de 58,6%. A seguir, testamos esse cenário criando um modelo linear de regressão logística, usando o módulo `sklearn.linear_model` e a classe `LogisticRegression`.

```
[15]: import numpy as np
      from sklearn.linear_model import LogisticRegression
      from sklearn.metrics import accuracy_score
      y = data.song_type # treino (X)
      X = data.drop('song_type', axis=1) # teste (y)
      logr = LogisticRegression().fit(X, y) # treinando o modelo de regressão logística
      pred_y_0 = logr.predict(X)
      print(accuracy_score(pred_y_0, y)) # acurácia do modelo
      print(np.unique(pred_y_0)) # classes previstas pelo modelo

0.5856697819314641
[0]
```

O resultado anterior indica que o modelo apresentou uma precisão geral de 58,6%; ou seja, como consequência dos dados desbalanceados, o modelo prevê apenas uma classe. Em outras palavras, ele ignora a classe minoritária (i.e., colaborações) em favor da classe majoritária (i.e., músicas solo). A seguir, aplicamos a técnica de *Downsampling*, que remove aleatoriamente observações da classe majoritária para evitar que seu sinal domine o algoritmo de aprendizagem: (1) separa-se as observações de cada classe em diferentes *DataFrames*; (2) realiza-se nova amostragem da classe majoritária sem substituição, definindo o número de amostras para corresponder ao da classe minoritária; e (3) concatena-se o *DataFrame* da classe minoritária com o *DataFrame* resultante.

```
[16]: from sklearn.utils import resample
      df_majority = data[data.song_type == 0] # classe majoritária
      df_minority = data[data.song_type == 1] # classe minoritária
      df_majority_downsampled = resample(
          df_majority, replace=False, # amostra sem substituição
          n_samples=532, # para corresponder à classe minoritária
          random_state=123) # garantindo reprodutibilidade
      df_downsampled = pd.concat([df_majority_downsampled, df_minority])
      df_downsampled.song_type.value_counts()

[16]: 1    532
      0    532
      Name: song_type, dtype: int64
```

Apesar de o novo *DataFrame* ter menos observações do que o original, a proporção das duas classes está balanceada. Pode-se, então, treinar novamente o modelo de regressão logística. Conforme o exemplo a seguir, após o balanceamento, o modelo prevê duas classes com menor precisão, mas com métrica de avaliação mais significativa.

```
[17]: y = df_downsampled.song_type # treino (X)
      X = df_downsampled.drop('song_type', axis=1) # teste (y)
      logr = LogisticRegression().fit(X, y) # Treinando o modelo de regressão logística
      pred_y_1 = logr.predict(X)
      print(np.unique(pred_y_1)) # acurácia do modelo
      print(accuracy_score(y, pred_y_1)) #classes previstas pelo modelo

[0 1]
0.49154135338345867
```

#### 1.4.4. Redução de Dados

Gerenciar e processar dados requer tempo, esforço e recursos, especialmente com grandes volumes de dados de várias fontes e em diferentes formatos. Para enfrentar tais desafios, técnicas de Redução de dados são aplicadas, auxiliando na análise de dados com alta dimensionalidade. Embora essenciais, essas técnicas geralmente são complexas, pois exigem amplo conhecimento para a escolha adequada de qual técnica utilizar. Assim, por simplicidade, essa seção apresenta apenas uma técnica, citando o texto [Iguar and Seguí 2017] para as demais.

A Análise de Componentes Principais (PCA) é um dos métodos mais simples e, de longe, o mais comum para a redução da dimensionalidade. PCA é um algoritmo não supervisionado<sup>5</sup> que cria combinações lineares dos atributos originais, classificadas em ordem de sua variância explicada. O próximo exemplo utiliza a biblioteca *scikit-learn* para importar o módulo `sklearn.decomposition` e a classe `PCA` para extrair os dois componentes principais (`n_components = 2`) do nosso conjunto de dados.

```
[5]: from sklearn.preprocessing import StandardScaler
      from sklearn.decomposition import PCA
      # Padronizando os dados de treino
      X = StandardScaler().fit_transform(X)
      # Calculando os dois componentes principais
      pca_resultado = PCA(n_components=2)
      df_pcs = pd.DataFrame(pca_resultado.fit_transform(X), columns=['PC1', 'PC2'])
      df_pcs.tail()
```

	PC1	PC2
1279	-1.850735	-1.341044
1280	5.250560	1.087556
1281	0.449023	-1.070109
1282	-0.780147	0.100485
1283	-2.275789	0.858764

O *DataFrame* resultante apresenta os valores dos dois componentes principais para todas as 1283 amostras. O conjunto de dados foi padronizado, utilizando a classe `StandardScaler`; do contrário, os atributos em maior escala dominariam os novos componentes principais. Após a extração dos componentes principais, o método fornece a quantidade de informações ou variação que cada componente principal mantém após projetar os dados em um subespaço de dimensão inferior. O primeiro componente principal detém 17,3% das informações, enquanto o segundo apenas 9% das informações. Ou seja, ao reduzir a dimensionalidade do conjunto de dados para duas dimensões, 73,7% das informações originais foram perdidas, conforme mostrado a seguir.

```
[6]: print('Variação explicada por componentes principais: {}'.format(
      pca_resultado.explained_variance_ratio_))
```

Variação explicada por componentes principais: [0.17349178 0.08857436]

## 1.5. Ciência de Dados Básica

Após o pré-processamento dos dados descrito na seção anterior, a sequência natural é realizar a Análise Exploratória dos Dados (*Exploratory Data Analysis* – EDA). Esta etapa consiste em entender melhor um conjunto de dados através de estatísticas descritivas e de gráficos visuais, sendo comum em Ciência de Dados para levantar hipóteses e responder questões sobre os dados em si. Apesar de relativamente simples, essa análise é importante e pode ser demorada, pois é conectada com a etapa de pré-processamento, podendo exigir reexecução de passos para assegurar melhor qualidade de dados.

Uma das principais vantagens de utilizar o Jupyter em projetos de Ciência de Dados é justamente a sua estrutura em células. Tal vantagem é evidenciada nas etapas de análise exploratória e visualização de dados (Seção 1.6), pois a fácil visualização em formato tabular ou gráfico permite a cientistas uma rápida análise descritiva de dados. Assim, a depender dos resultados, é possível corrigir, descartar ou gerenciar os dados de forma apropriada. Além disso, não existe uma única sequência de passos para descrever e lidar com dados. Cabe a cientistas de dados escolher as melhores técnicas para a tarefa de análise, a partir de um entendimento prévio e do arcabouço prático disponível.

<sup>5</sup> Algoritmos não-supervisionados visam extrair padrões de dados não-rotulados (e.g., definir grupos de instâncias a partir de características comuns). Para mais detalhes, ver [Igal and Seguí 2017, Skiena 2017].



Mesmo sem sequência fixa para análise exploratória, esta seção apresenta os passos comumente utilizados. Os principais tópicos abordados incluem a exploração e descrição inicial de um *DataFrame*, estatísticas básicas para variáveis numéricas e categóricas, além da análise de distribuições e de correlação de valores. Aqui, é utilizada a tabela *Tracks* do conjunto de dados da Seção 1.2, que contém informações relevantes sobre as canções que entraram na parada global de sucesso do Spotify no ano de 2020.

### 1.5.1. Exploração Inicial

A etapa de exploração inicial é simples, mas importante para cientistas se familiarizarem com o conteúdo de seus dados. A seção anterior focou no pré-processamento de dados para deixá-los prontos para as primeiras análises, conforme descrito a seguir.

**Amostras de Dados.** Uma das primeiras ações sobre um novo conjunto de dados é verificar uma amostra de suas instâncias, para entender o conteúdo das colunas e o tipo de informação em cada uma delas. O *pandas* possui funções que permitem visualizar tais amostras. Utilizá-las em um ambiente Jupyter facilita ainda mais a compreensão, pois permite uma visualização simples e amigável do *DataFrame* ao final de cada célula. Um exemplo é a função `head()` para visualizar as primeiras linhas de um *DataFrame*.

```
[3]: df.head() # Mostrando os primeiros registros do dataframe (5, por padrão)
```

```
[3]:
```

	song_id	song_name	artist_id	...	speechiness	valence	tempo
0	2rRjJrJEo19S2J82BDsQ3F7	Falling	['7ualm6Pw7xplS8Dy06V6pT']	...	0.0364	0.236	127.087
1	3BYIzNZ3i9iRQCACXSMLrT	Venetia	['4O15NlyKLIASxsJ0PrXPfz']	...	0.1750	0.562	142.933
2	1g3J9W88hTG173ySZR6E9S	Tiidin Weg	['1aS5tqEs9ci5P9KD9tZWa6']	...	0.2330	0.171	109.090
3	75pQqzwwGjUOSSy5CpmAjy	Pero Ya No	['4q3ewBCX7sLwd24euv69X']	...	0.1180	0.742	147.982
4	7kDUspsoYlLkWNZR7qwhZI	my ex's best friend (with blackbear)	['6TIYQ3jFPwQSRmorSezPxx', '2cFrymmkijnjDg9SS9...']	...	0.0434	0.298	124.939

5 rows x 24 columns

Por padrão, `head()` exibe as cinco primeiras linhas do *DataFrame*, mas também permite usar o parâmetro `n` para personalizar a saída. O próximo exemplo mostra somente os dois primeiros registros. Esta função limita apenas as linhas, e então todas as colunas do *DataFrame* são mostradas no resultado.

```
[4]: df.head(n=2) # Exibindo somente os primeiros dois registros do dataframe
```

```
[4]:
```

	song_id	song_name	artist_id	...	speechiness	valence	tempo
0	2rRjJrJEo19S2J82BDsQ3F7	Falling	['7ualm6Pw7xplS8Dy06V6pT']	...	0.0364	0.236	127.087
1	3BYIzNZ3i9iRQCACXSMLrT	Venetia	['4O15NlyKLIASxsJ0PrXPfz']	...	0.1750	0.562	142.933

2 rows x 24 columns

De forma similar, o *pandas* também oferece a função `tail()`, que exibe as últimas linhas do *DataFrame*. Esta função também pode ter a saída personalizada com o parâmetro `n`, e, portanto, é uma importante ferramenta na inspeção manual dos registros.

```
[5]: df.tail(n=1) # Mostrando o último registro do dataframe
```

```
[5]:
```

	song_id	song_name	artist_id	...	speechiness	valence	tempo
1283	2bgoUk2A3jkbCJ7KpQuTi	Mi Niña	['3E6xrwgnvFYCrCs0ePERDz', '7iK8PXO48WeuP03g8Y...']	...	0.166	0.791	99.999

1 rows x 24 columns

Entretanto, tais funções só permitem analisar as extremidades do *DataFrame*, e uma inspeção mais aprofundada sobre os outros registros pode ser necessária para um melhor entendimento dos dados. Dessa forma, o *pandas* também oferece a função `sample()`, que retorna de fato uma amostra aleatória dos registros do *DataFrame*. Esta função

possui mais parâmetros disponíveis do que as anteriores, o que oferece um maior poder de análise para cientistas. O próximo exemplo usa os parâmetros `n`, para a quantidade de linhas amostradas, e `random_state`, que é uma semente de um gerador de números aleatórios que mantém o resultado caso a célula seja executada várias vezes.

```
[6]: df.sample(n=3, random_state=7) # Mostrando uma amostra aleatória de n=3 linhas do DataFrame
```

```
[6]:
```

	song_id	song_name	artist_id	...	speechiness	valence	tempo
	12450PQsrLxPbOBBwmmXCnGvcF	Diamonds (with Normani)	['181bsRPaVXVLUKXrxwZiHK', '2cWZOzeOm4WmBJRnD...	...	0.0873	0.488	94.012
	992 1bRpSCFv6P2OUhciByeRYR	Jangueo	['2DspEsT7UXGKd2VaaedgG4', '11YLRsSzA3YVuQQtHX...	...	0.0536	0.723	103.994
	1045 4r9jkMErArtWGH2rL2FZI0	A Tu Merced	['4q3ewBCX7sLwd24euuV69X']	...	0.0568	0.887	92.023

3 rows x 24 columns

**Dimensões e tipos de variáveis.** Além de amostras, conhecer informações como as dimensões e os tipos das colunas de um *DataFrame* é igualmente útil. Para saber as dimensões de um *DataFrame* (ou qualquer objeto *pandas*, como uma *Series*), basta acessar o atributo `shape`. Para um *DataFrame*, este atributo contém uma tupla com dois valores, informando o número de linhas e colunas, respectivamente, como a seguir.

```
[7]: print('Dimensões do DataFrame completo: ', df.shape)
print('Dimensões da coluna song_id, que é uma Series: ', df['song_id'].shape)
```

```
Dimensões do DataFrame completo: (1284, 24)
Dimensões da coluna song_id, que é uma Series: (1284,)
```

O *pandas* possui outras funções com informações gerais sobre um *DataFrame*. No próximo exemplo, a função `info()` retorna, para cada coluna, a quantidade de valores não-nulos, a existência desses valores, bem como o tipo de dado armazenado naquela coluna (*dtype*). O tipo de dados é essencialmente a construção interna que a linguagem usa para entender como armazenar e manipular os dados. A Tabela 1.2 apresenta a lista de todos os tipos de dados suportados pelo *pandas*.

```
[8]: df.info() # Verificando tipos das variáveis
```

#	Column	Non-Null Count	Dtype
0	song_id	1284 non-null	object
1	song_name	1284 non-null	object
2	artist_id	1284 non-null	object
3	artist_name	1284 non-null	object
4	popularity	1284 non-null	int64
5	explicit	1284 non-null	bool
6	song_type	1284 non-null	object
7	track_number	1284 non-null	int64
8	num_artists	1284 non-null	int64
9	num_available_markets	1284 non-null	int64
10	release_date	1284 non-null	object
11	duration_ms	1284 non-null	int64

dtypes: bool(1), float64(9), int64(8), object(6)  
memory usage: 232.1+ KB

Conhecer os tipos de dados é crucial em projetos de Ciência de Dados. Se o *pandas* acusa um tipo string onde se espera um valor numérico, tal coluna pode conter ruído. Assim, é necessário voltar à etapa de pré-processamento e realizar a remoção de ruídos antes de prosseguir para análises mais complexas (ver Seção 1.4).

**Tabela 1.2. Lista dos tipos de dados (dtypes) suportados pelo *pandas* e sua descrição.**

Tipo	Descrição
<b>object</b>	<i>String</i> ou uma mistura de valores numéricos e não-numéricos
<b>int64</b>	Valores inteiros
<b>float64</b>	Valores com ponto flutuante
<b>bool</b>	Valores booleanos ( <b>True/False</b> )
<b>datetime64</b>	Valores com data e hora
<b>timedelta[ns]</b>	Diferença entre valores <b>datetime64</b>
<b>category</b>	Lista finita de categorias ( <i>string</i> )

### 1.5.2. Estatísticas Básicas para Dados Numéricos

Em geral, variáveis de conjuntos de dados podem ser classificadas como *categóricas* ou *numéricas/quantitativas*. Dados categóricos são aqueles obtidos a partir de observações nominais (i.e., classes), enquanto as variáveis numéricas são derivadas de medições quantitativas em uma escala numérica. Por exemplo, a cor de uma régua é um atributo categórico, e seu valor sempre será obtido de uma lista de categorias bem definidas (i.e., cores). Já o seu comprimento é um atributo numérico, medido com valores inteiros. Para cada tipo, existem um conjunto de estatísticas descritivas que permitem sumarizar essas informações de uma forma mais legível e próxima para as pessoas analistas.

Para variáveis numéricas, a análise exploratória pode considerar métodos de agregação sobre a população (i.e., conjunto de todas as instâncias do conjunto de dados). Exemplos incluem média, mediana, desvio padrão, soma, valores máximos e mínimos. Todas estão disponíveis de forma simples e prática pelo *pandas*, e cientistas não precisam gastar tempo ou linhas de código para obtê-las. O próximo exemplo mostra a média<sup>6</sup> de todas as variáveis numéricas de um *DataFrame* com um único comando.

```
[9]: df.mean() # Exibindo a média das variáveis numéricas
```

popularity	68.959502	danceability	0.697962
explicit	0.445483	energy	0.627272
track_number	4.961059	instrumentalness	0.011885
num_artists	1.598910	liveness	0.180263
num_available_markets	157.330997	loudness	-6.415536
duration_ms	196804.523364	speechiness	0.124298
key	5.331776	valence	0.511828
mode	0.564642	tempo	123.467972
time_signature	3.966511	dtype: float64	
acousticness	0.243906		

Nesse exemplo, todas as variáveis exibidas são do tipo **int64** ou **float64**, com a exceção de **explicit** (informa se a letra de uma música contém termos explícitos), cujo tipo é **bool**. Neste caso, o próprio *pandas* converte esses valores booleanos para inteiros (i.e., 0 para **False** e 1 para **True**), permitindo então o cálculo da média. Assim como a média, também existem funções no *pandas* para outros métodos de agregação para variáveis numéricas. A Tabela 1.3 apresenta as principais delas acompanhadas de sua descrição.

As funções de agregação do *pandas* também podem ser aplicadas a colunas individuais de um *DataFrame*, isto é, objetos do tipo *Series*. A seguir, são exibidas algumas

<sup>6</sup>Aqui média é com o comando **mean**, mas representa o que em outros softwares é calculado com **average**. Não confundir com a mediana, cujo comando em Python é **median**.

Tabela 1.3. Principais funções de agregação disponibilizadas pelo *pandas*.

Função	Descrição	Função	Descrição	Função	Descrição
<b>count</b>	Número de observações	<b>max</b>	Valor máximo	<b>sem</b>	Erro padrão da média
<b>sum</b>	Soma de valores	<b>mode</b>	Moda	<b>skew</b>	Assimetria ( <i>Skewness</i> )
<b>mean</b>	Média dos valores	<b>abs</b>	Valor absoluto	<b>kurt</b>	Curtose ( <i>Kurtosis</i> )
<b>mad</b>	Desvio absoluto médio	<b>prod</b>	Produto de valores	<b>quantile</b>	Quantis (valor em %)
<b>median</b>	Valor mediano	<b>std</b>	Desvio padrão	<b>cumsum</b>	Soma cumulativa
<b>min</b>	Valor mínimo	<b>var</b>	Variância	<b>cumprod</b>	Produto cumulativo

características da duração das músicas, medidas em milissegundos na coluna `duration_ms`.

```
[10]: print('Registros não-nulos:', df['duration_ms'].count())
      print('Valor máximo:', df['duration_ms'].max())
      print('Valor mínimo:', df['duration_ms'].min())
      print('Média:', df['duration_ms'].mean())
      print('Mediana:', df['duration_ms'].median())
      print('Desvio padrão:', df['duration_ms'].std())
      print('Variância:', df['duration_ms'].var())
```

Registros não-nulos: 1284  
 Valor máximo: 484146  
 Valor mínimo: 30133  
 Média: 196804.52336448597  
 Mediana: 193683.0  
 Desvio padrão: 44185.523479052776  
 Variância: 1952360485.1179242

**Percentis.** O *pandas* também oferece a funcionalidade de calcular percentis, que são pontos estabelecidos em uma função de distribuição de probabilidade de uma variável aleatória. A função `quantile()` permite o cálculo de um ou mais percentis a partir do parâmetro ‘q’, que aceita valores entre 0 e 1 (o 50º percentil, com  $q = 0,5$  é a mediana). A seguir, o 10º percentil da variável `duration_ms` é calculado, ou seja, o valor abaixo do qual se encontram 10% das instâncias dessa variável (ordenadas da menor para a maior). O comando seguinte mostra como efetuar o cálculo de percentis para todo o *DataFrame*.

```
[11]: # Cálculo do 10º percentil
      # (q=0.1) de duration_ms
      df['duration_ms'].quantile(
          q=0.1
      )
```

[11]: 150759.9

```
[12]: # 10º e 90º percentis para variáveis numéricas
      df.quantile(q=[0.1, 0.9])
```

```
[12]:
```

	popularity	explicit	track_number	...	speechiness	valence	tempo
0.1	54.0	0.0	1.0	...	0.0346	0.2076	86.3606
0.9	85.0	1.0	13.0	...	0.2900	0.8227	167.9285

2 rows x 18 columns

Por fim, o *pandas* oferece uma função que combina os principais agregadores utilizados nesta seção. A função `describe()` permite verificar, para as colunas numéricas do *DataFrame*, a quantidade de valores não-nulos (**count**), a média, desvio padrão, valores mínimos, máximo, bem como os percentis 25º, 50º (mediana) e 75º.

```
[13]: df.describe() # Principais informações descritivas para o DataFrame
```

```
[13]:
```

	popularity	track_number	num_artists	...	speechiness	valence	tempo
count	1284.000000	1284.000000	1284.000000	...	1284.000000	1284.000000	1284.000000
mean	68.959502	4.961059	1.598910	...	0.124298	0.511828	123.467972
std	17.942532	5.003355	0.929485	...	0.109577	0.226580	29.547921
min	0.000000	1.000000	1.000000	...	0.023200	0.036000	46.718000
25%	65.000000	1.000000	1.000000	...	0.045800	0.341000	99.028500
50%	73.000000	3.000000	1.000000	...	0.078550	0.508500	122.066000
75%	80.000000	8.000000	2.000000	...	0.173250	0.680250	144.843500
max	97.000000	30.000000	9.000000	...	0.884000	0.978000	205.272000

8 rows x 17 columns

### 1.5.3. Estatísticas Básicas para Dados Categóricos

Para dados categóricos, uma das principais estratégias descritivas é a contagem de instâncias pertencentes a cada categoria. Assim, cientistas de dados podem entender melhor a

frequência de ocorrência de cada classe em um determinado conjunto de dados. O *pandas* oferece a função `value_counts`, que retorna as classes existentes ordenadas pela quantidade de elementos. Essa função tem um parâmetro *booleano* `normalize` que, caso verdadeiro, retorna a frequência (porcentagem) de ocorrência de cada classe. Seguindo as análises sobre o *DataFrame* com os dados de músicas, a coluna `song_type` informa o tipo de música com relação à quantidade de artistas (solo ou colaboração). O exemplo a seguir aplica `value_counts` nessa coluna com o parâmetro `normalize`.

```
[14]: # Contagem de valores para o tipo de música
df['song_type'].value_counts()

[15]: # Frequência dos tipos de música
df['song_type'].value_counts(normalize=True)
```

<pre>[14]: Solo          752 Collaboration    532 Name: song_type, dtype: int64</pre>	<pre>[15]: Solo          0.58567 Collaboration    0.41433 Name: song_type, dtype: float64</pre>
---	---

Assim, descobre-se que mais da metade das músicas no conjunto de dados são canções solo (752 ou 58,6%), enquanto as colaborações representam aproximadamente 41,4% (532 canções). Esse resultado pode ser exibido de uma forma mais legível apenas adicionando uma função à contagem de valores. De forma gráfica, utilizando o comando `.plot.barh()`, transformamos a contagem/frequência em um gráfico de barras horizontal simples, conforme exemplo a seguir. Aqui, o parâmetro `figsize` foi usado para legibilidade.

```
[16]: # Frequência em um gráfico de barras
(
df['song_type'].value_counts(normalize=True)
.plot.barh(figsize=(7, 1.25))
)
```

[16]: <AxesSubplot:>

song_type	frequency
Solo	0.58567
Collaboration	0.41433

#### 1.5.4. Distribuições

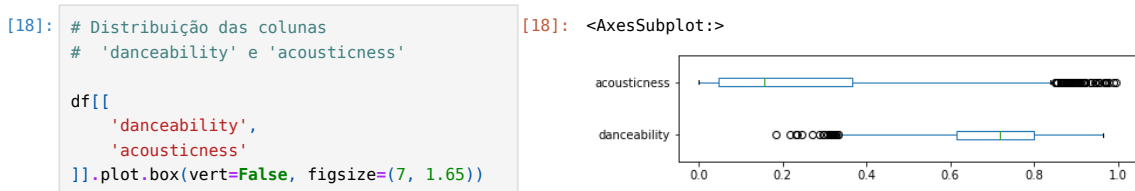
Estatísticas básicas para variáveis numéricas como média, mediana e variância são um bom ponto de partida para compreender conjuntos de dados. No entanto, elas não são suficientes para descrever totalmente esses dados, pois dados completamente diferentes podem ter a mesma média, por exemplo. Dessa forma, é necessário aprofundar a análise olhando para a distribuição, que de forma geral apresenta a frequência com que os valores aparecem no conjunto de dados. Nesta seção, são apresentadas duas das principais formas de se visualizar distribuições de valores: *boxplots* e histogramas.

**Boxplots.** Também conhecidos como *diagramas de caixa*, são uma forma padronizada e visual de sumarizar a distribuição de uma variável através de seus quartis: primeiro (Q1), segundo (Q2, mediana) e terceiro (Q3). Além deles, um *boxplot* também exibe outras informações importantes, como o intervalo interquartil (IQR) e *outliers*. As principais definições necessárias para entender um *boxplot* são as seguintes:

- *Mediana* (Q2, 50° percentil): o valor “do meio” do conjunto de dados;
- *Primeiro quartil* (Q1, 25° percentil): valor abaixo do qual se encontram os primeiros 25% dos valores do conjunto de dados (ordenados do menor para o maior);
- *Terceiro quartil* (Q3, 75° percentil): valor abaixo do qual se encontram os primeiros 75% dos valores do conjunto de dados (ordenados do menor para o maior); e
- *Intervalo interquartil* (IQR): intervalo entre o 25° e o 75° percentis.

Então, um *boxplot* para uma variável consiste em uma caixa cujas extremidades vão do primeiro ao terceiro quartil (Q1 a Q3), com uma linha indicando a mediana (Q2). Além disso, este tipo de gráfico contém linhas (“bigodes”) que saem das extremidades da caixa e possuem o comprimento de  $1,5 * IQR$ . Todos os valores acima ou abaixo dos “bigodes” são classificados como *outliers*, e, portanto, são visualizados como círculos.

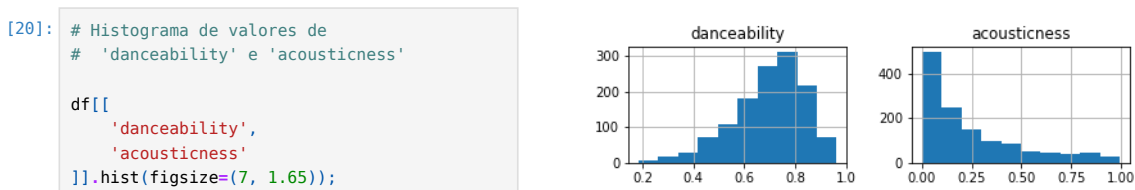
O comando `.plot.box()` exibe um *boxplot* para uma ou mais colunas de um *DataFrame* (ou uma *Series*). O exemplo a seguir mostra *boxplots* para as colunas **danceability** e **acousticness**, features do Spotify para a probabilidade de uma música ser dançante e acústica, respectivamente. Assim como `figsize`, o parâmetro `vert` foi usado para legibilidade.



Com esse exemplo, percebe-se que os valores de **acousticness** estão em sua maioria abaixo de 0,5, e poucos outliers com valores altos (círculos fora dos quartis). Além disso, a maioria dos valores de **danceability** se encontra em um patamar acima de 0,5, ou seja, a maioria das músicas do nosso conjunto de dados possui um alto valor para essa feature e, portanto, são bastante dançantes. Também são poucas as músicas com valores baixos para essa métrica (outliers). Como nosso conjunto de dados compreende músicas que entraram nas paradas de sucesso do Spotify em 2020, pode-se afirmar que a maioria dos hits são dançantes e não-acústicos (geralmente gravadas em estúdio).

**Histogramas.** Em linhas gerais, um histograma é uma representação da distribuição de dados numéricos, mostrando a frequência dos valores que ocorrem em um conjunto de dados. O intervalo em que os valores ocorrem é dividido em partes iguais (*bins*) cujo tamanho pode ser especificado pela pessoa que analisa os dados, e a quantidade de valores que estão em cada uma dessas partes é representada por barras. Diferentemente dos gráficos de barras para dados categóricos, histogramas não possuem espaços entre as barras por representarem valores em um espectro contínuo.

No *pandas*, um histograma de valores é facilmente construído adicionando o comando `.hist()` em um *DataFrame* ou *Series*. Assim como para os *boxplots*, os histogramas a seguir mostram a distribuição de valores das colunas **danceability** e **acousticness** do conjunto de músicas das paradas globais do Spotify. Observe que as mesmas conclusões dos *boxplots* podem ser obtidas a partir dos histogramas, por serem duas formas diferentes de visualizar a mesma informação, i.e., a distribuição dos valores para um atributo.



### 1.5.5. Correlações

Correlação é uma métrica que informa o grau em que duas ou mais variáveis estão relacionadas entre si. A correlação entre variáveis pode ser medida por um coeficiente, cujo valor varia entre -1 e 1. Uma correlação é dita totalmente positiva quando o coeficiente é igual a 1, e totalmente negativa quando igual a -1. Coeficientes iguais a zero informam que não existe correlação explícita entre as variáveis consideradas. Existem várias formas de se calcular coeficientes de correlação, dentre as quais pode-se eleger duas principais:

- *Pearson* ( $r$ ): mede a correlação linear (de primeira ordem) entre variáveis;
- *Spearman* ( $\rho$ ): mede correlação monotônica entre variáveis, i.e., utiliza a ordem dos dados ao invés dos valores em si.

Para calcular correlações, o *pandas* dispõe da função `corr()`, que pode ser facilmente aplicada tanto para *Series* quanto para *DataFrame*. O exemplo a seguir exhibe o valor do coeficiente de Pearson para as colunas `energy` e `loudness`, que medem a intensidade de uma música e a altura medida em decibéis (dB), respectivamente.

```
[21]: # Correlação de Pearson entre as colunas energy e loudness
df['energy'].corr(other=df['loudness'], method='pearson')
```

```
[21]: 0.737109548990312
```

A partir de tal resultado, pode-se dizer que a correlação linear entre essas duas variáveis é forte e positiva, pois o valor de 0,737 é muito próximo de 1. Ou seja, à medida que o valor de `energy` aumenta, o valor de `loudness` também cresce, e vice-versa. É importante notar que correlação nem sempre implica causalidade, e então não se pode afirmar que uma coisa acontece devido à outra, mas que ambas ocorrem juntas.

Além disso, cientistas de dados frequentemente calculam correlações entre mais de duas variáveis para ter uma visão ampla de algum fenômeno. Para isso, **matrizes de correlação** são bastante úteis, pois permitem uma fácil e rápida visualização em única estrutura de dados. Em uma matriz de correlação  $M$ , as linhas e colunas são as variáveis para as quais se deseja observar a correlação, e os valores  $m_{ij}$  representam a correlação entre as variáveis  $i$  e  $j$ . A diagonal principal dessas matrizes é sempre igual a 1, pois a correlação entre uma variável e ela mesma é total. O próximo exemplo mostra a matriz de correlação entre algumas *features* acústicas das músicas, i.e., obtidas do áudio da mesma.

```
[22]: # Construção da matriz de correlação para features acústicas
acoustic_features = ['acousticness', 'danceability', 'energy', 'instrumentalness', 'loudness', 'valence']
df[acoustic_features].corr(method='spearman') # Aqui, vamos utilizar a correlação de Spearman
```

```
[22]:
```

	acousticness	danceability	energy	instrumentalness	loudness	valence
acousticness	1.000000	-0.210552	-0.427403	-0.007332	-0.313412	-0.034162
danceability	-0.210552	1.000000	0.051333	-0.017828	0.149479	0.312837
energy	-0.427403	0.051333	1.000000	0.009851	0.703220	0.343683
instrumentalness	-0.007332	-0.017828	0.009851	1.000000	-0.130229	-0.133157
loudness	-0.313412	0.149479	0.703220	-0.130229	1.000000	0.330416
valence	-0.034162	0.312837	0.343683	-0.133157	0.330416	1.000000

Nessa matriz, utiliza-se a correlação de Spearman entre as colunas do *DataFrame*. O tipo de correlação pode ser definido através do parâmetro `method`. Além dos coeficientes de Pearson e Spearman, a função `corr()` do *pandas* também permite calcular a correlação de Kendall ( $\tau$ ), que também é baseada na ordem das variáveis.

De modo geral, a análise exploratória dos dados possui grande interseção com a etapa de visualização de dados, pois muitas ferramentas de análise produzem gráficos como resultado (histogramas, gráficos de caixa (*boxplots*), dentre outros). No entanto, o objetivo dessa seção foi apresentar as principais funções do *pandas* e outras bibliotecas que auxiliam na exploração dos dados, ou seja, o foco aqui reside na análise de tais dados. Mais informações sobre os diferentes tipos de visualizações e como personalizá-las são abordadas na próxima seção.

## 1.6. Visualização de Dados Científicos

Na Ciência de Dados, a Visualização de dados é uma fase importante que envolve a representação gráfica de informações. Através de elementos visuais, ela permite a atribuição de sentido e a comunicação de tendências e padrões nos dados. É um meio de comunicação de resultados de pesquisa, bem como uma plataforma de análise e exploração de dados. Portanto, saber como criar visualizações de dados ajuda cientistas a resolver problemas e analisar os objetos de um estudo de forma detalhada e significativa.

Existem vários métodos e abordagens para a criação de recursos visuais com base na natureza e na complexidade dos dados. Em particular, o Jupyter Notebook é uma ótima ferramenta para a exploração de dados e a criação de visualizações. Ao mesmo tempo, associado ao suporte simplificado do Jupyter, a linguagem Python oferece uma variedade de bibliotecas e módulos de visualização de dados. Matplotlib e Seaborn são duas das bibliotecas de visualização mais populares do Python. Ambas funcionam muito bem para a tarefa de visualização, sendo mais poderosas quando comparadas com os recursos adicionais de plotagem incorporados do *pandas*, apresentadas na seção anterior.

Diante da variedade de opções e métodos, esta seção fornece uma visão geral de como visualizar conjuntos de dados no Jupyter Notebook com a ajuda das bibliotecas Matplotlib e Seaborn para demonstrar os recursos de plotagem do Python. Para melhor compreensão e organização, descrevemos um conjunto básico de recursos visuais agrupados não pelo tipo de dados visualizados, mas pelo tipo de mensagem ou informação que transmitem. Assim, esta seção é dividida em quatro categorias de visualização: Quantidade, Distribuição, Correlação e Evolução.

### 1.6.1. Quantidade

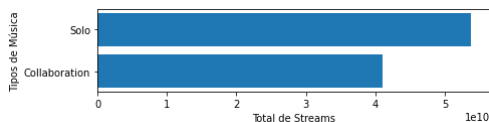
Ao visualizar quantidades, o interesse está em analisar a magnitude de algum conjunto de números. Por exemplo, pode-se querer visualizar o número total de *streams* de diferentes tipos de música, ou o número total de artistas para cada tipo de gênero musical. Em todos esses casos, existe um conjunto de categorias (por exemplo, tipos de música e gêneros musicais) e um valor quantitativo para cada categoria (número total de *streams*). A visualização mais utilizada para esse cenário é o gráfico de barras.

**Gráfico de barras simples.** Em um gráfico de barras, cada entidade da variável categórica é representada como uma barra, e o tamanho da barra representa seu valor numérico. Para exemplificar o conceito de gráfico de barras, considere o total de *streams* no Spotify para as músicas solo e colaborativas mais populares em 2020. Esse tipo de dado é comumente visualizado com barras horizontais ou verticais. O exemplo a seguir ilustra como construir um gráfico de barras simples usando Matplotlib (à esquerda) e Seaborn (à

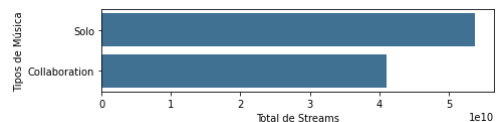


direita). Em ambos os gráficos, para cada tipo de música, existe uma barra que começa em zero e se estende até a quantidade de *streams*.

```
[5]: fig, ax = plt.subplots()
ax.barh(y=df['Tipos de Música'],
        width=df['Total de Streams'])
ax.set_xlabel('Total de Streams')
ax.set_ylabel('Tipos de Música')
fig.set_size_inches(7, 1.5)
```

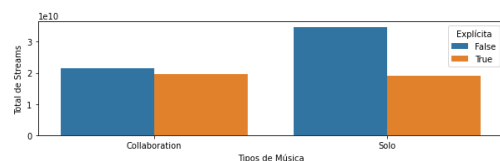


```
[6]: sns.barplot(data=df,
                 x="Total de Streams",
                 y="Tipos de Música",
                 order=['Solo', 'Collaboration'],
                 color='#3274A1')
plt.gcf().set_size_inches(7, 1.5)
```



**Gráfico de barras agrupadas.** O exemplo anterior mostra como uma quantidade varia em relação a uma variável categórica. Porém, frequentemente, é necessário analisar duas variáveis categóricas ao mesmo tempo. Por exemplo, agora pretende-se saber o total de *streams* de músicas explícitas ou não, além do tipo da música. Tal informação é melhor visualizada com um gráfico de barras agrupadas. Esse tipo de gráfico contém um grupo de barras em cada posição ao longo do eixo *x*, determinado por uma variável categórica, e mais barras dentro de cada grupo conforme a outra variável categórica. As legendas do eixo e do gráfico distinguem os grupos, como o exemplo a seguir.

```
[9]: sns.barplot(
    x="Tipos de Música", y="Total de Streams",
    hue="Explícita",
    data=df)
plt.gcf().set_size_inches(10, 2.5)
```

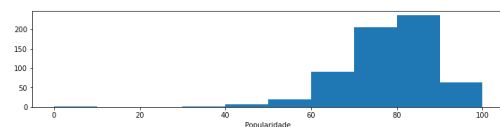


## 1.6.2. Distribuição

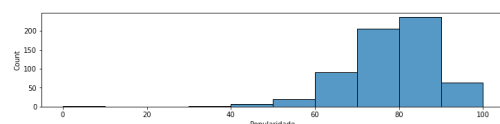
Visualização de distribuições facilita entender como uma determinada variável é distribuída em um conjunto de dados. Por exemplo, para visualizar a distribuição da popularidade ou número de seguidores de artistas de sucesso, pode-se utilizar histogramas. Além de histogramas, existem visualizações alternativas como gráficos de densidade e *boxplots*.

**Histograma.** É uma representação gráfica da distribuição de uma variável numérica, no qual a variável é dividida em barras, e o número de observações por barra é representado pela sua altura. Considere a distribuição da popularidade de artistas das músicas de 2020. Para cada artista, o próximo exemplo mostra a distribuição da variável **popularity** dividida por intervalos de tamanho igual a dez, usando Matplotlib e Seaborn.

```
[4]: fig, ax = plt.subplots()
ax.hist(df["Popularidade"], bins=10)
ax.set_xlabel('Popularidade');
plt.gcf().set_size_inches(12, 2.5)
```

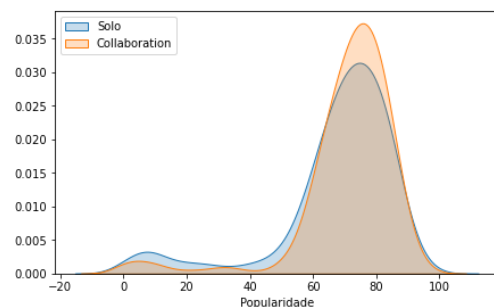


```
[5]: sns.histplot(df["Popularidade"],
                 kde=False,
                 bins=10);
plt.gcf().set_size_inches(12, 2.5)
```



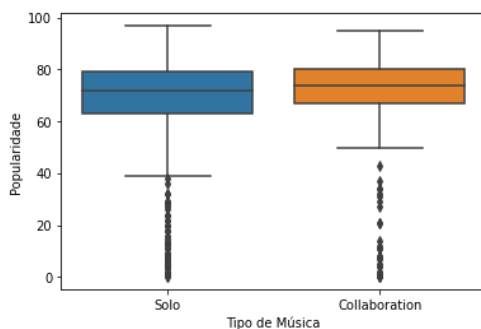
**Gráfico de densidade.** Geralmente, histogramas são mais utilizados para visualizar uma única distribuição. Porém, ao visualizar mais de uma distribuição simultaneamente, gráficos de densidade são mais recomendados. Em um gráfico de densidade, a distribuição é visualizada através de uma curva contínua. Essa curva precisa ser estimada a partir dos dados, e o método mais comum para tal é chamado de estimativa de densidade de kernel. O exemplo a seguir utiliza o kernel gaussiano através do método `seaborn.kdeplot()` para estimar a distribuição da popularidade conforme o tipo de música. As cores das curvas indicam se a distribuição é de músicas solo ou colaborações.

```
[7]: # Plotando duas distribuições em uma mesma figura
fig = sns.kdeplot(df.loc[
    df['Tipo de Música'] == 'Solo', 'Popularidade'
], shade=True, label='Solo')
fig = sns.kdeplot(df.loc[
    df['Tipo de Música'] == 'Collaboration',
    'Popularidade'
], shade=True, label='Collaboration')
plt.xlabel("Popularidade")
plt.ylabel("")
plt.legend(loc='upper left')
plt.gcf().set_size_inches([7, 4.25])
```

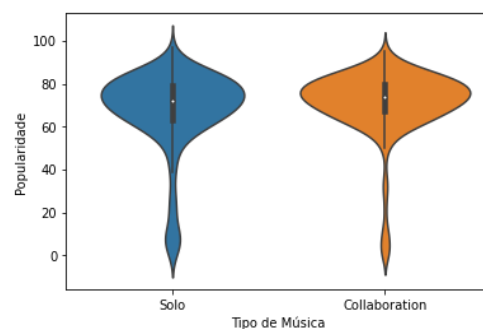


**Boxplots e Gráficos Violino.** Assim como os gráficos de densidade, *boxplots* são uma ótima maneira de visualizar distribuições. Em um *boxplot*, os dados são divididos em quartis, fornecendo um bom resumo da distribuição de variáveis numéricas. Esse tipo de visualização tem a vantagem de ocupar pouco espaço, o que é útil ao comparar distribuições entre muitos grupos ou conjuntos de dados. Uma desvantagem dos *boxplots* é que ao resumir as distribuições, informações podem ser perdidas. O *boxplot* do exemplo a seguir (à esquerda), aparentemente, permite concluir que as colaborações são em média mais populares do que as músicas solo. No entanto, o gráfico não ilustra a distribuição subjacente de pontos em cada categoria ou o número de observações. Uma alternativa é utilizar gráficos violino (à direita), que descrevem a distribuição permitindo uma compreensão mais profunda sobre a mesma. Os gráficos a seguir utilizam os métodos `seaborn.boxplot()` e `seaborn.violinplot()` para melhor visualização das distribuições do exemplo anterior.

```
[8]: # Criando um boxplot básico
sns.boxplot(x=df["Tipo de Música"],
            y=df["Popularidade"]);
```



```
[9]: # Criando um gráfico violino
sns.violinplot(x=df["Tipo de Música"],
              y=df["Popularidade"]);
```

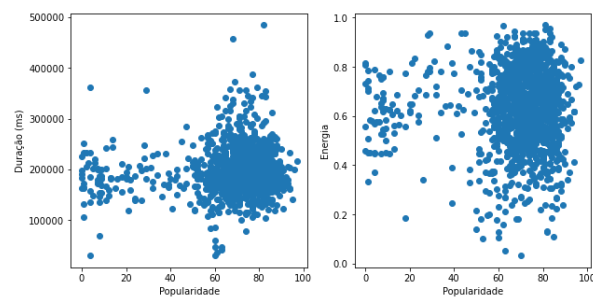


### 1.6.3. Correlação

Correlação permite analisar como duas variáveis se relacionam entre si, por exemplo, entre popularidade e duração da música. Por exemplo, podemos querer visualizar a relação entre a popularidade e alguma feature acústica de música de sucesso. Para representar graficamente a relação de duas dessas variáveis, geralmente se utiliza um gráfico de dispersão. Ele pode ser útil também para conjuntos de dados com alta dimensionalidade, como exemplificado na Seção 1.5. Esta seção explica melhor o gráfico de dispersão básico e inclui algumas variações, como os gráficos de bolhas e correlogramas. Todos os exemplos desta seção consideram o conjunto de dados *Hits*.

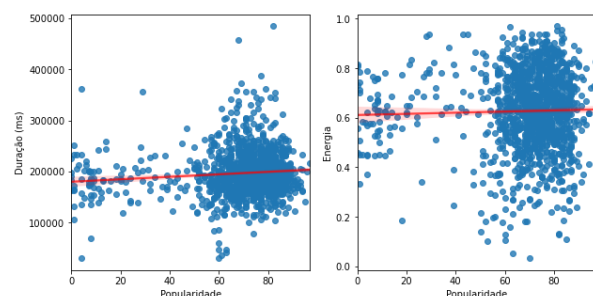
**Gráfico de dispersão.** Permite estudar a relação entre duas variáveis. Para cada ponto de dados, o valor de sua primeira variável é representado no eixo  $x$  e a segunda no eixo  $y$ . Os dois gráficos a seguir exemplificam a dispersão sobre a popularidade em relação à duração e à energia das músicas, respectivamente, utilizando o método `matplotlib.pyplot.scatter()`. Note que `subplots()` permite colocar gráficos lado-a-lado na mesma saída.

```
[3]: fig, ax = plt.subplots(1, 2, figsize=(10, 5))
# popularidade vs. duração
ax[0].scatter(
    x = df_hits['popularity'],
    y = df_hits['duration_ms'])
ax[0].set_xlabel("Popularidade")
ax[0].set_ylabel("Duração (ms)")
# popularidade vs. energia
ax[1].scatter(
    x = df_hits['popularity'],
    y = df_hits['energy'])
ax[1].set_xlabel("Popularidade")
ax[1].set_ylabel("Energia");
```



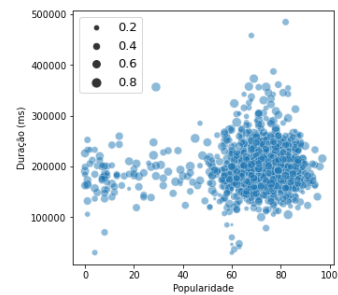
Geralmente, linhas ou curvas são ajustadas dentro do gráfico de dispersão para auxiliar a análise. O método `seaborn.regplot()` permite criar os mesmos gráficos de dispersão, mas, agora, com uma reta vermelha representando o ajuste linear.

```
[4]: # Gráfico de dispersão com o ajuste linear
fig, ax = plt.subplots(ncols=2, figsize=(10,5))
line_kws = {"color": "r", "alpha": 0.7}
sns.regplot(x='popularity', y='duration_ms',
            data=df_hits, ax=ax[0],
            line_kws=line_kws)
ax[0].set(xlabel='Popularidade',
          ylabel='Duração (ms)')
sns.regplot(x='popularity', y='energy',
            data=df_hits, ax=ax[1],
            line_kws=line_kws)
ax[1].set(xlabel='Popularidade',
          ylabel='Energia');
```



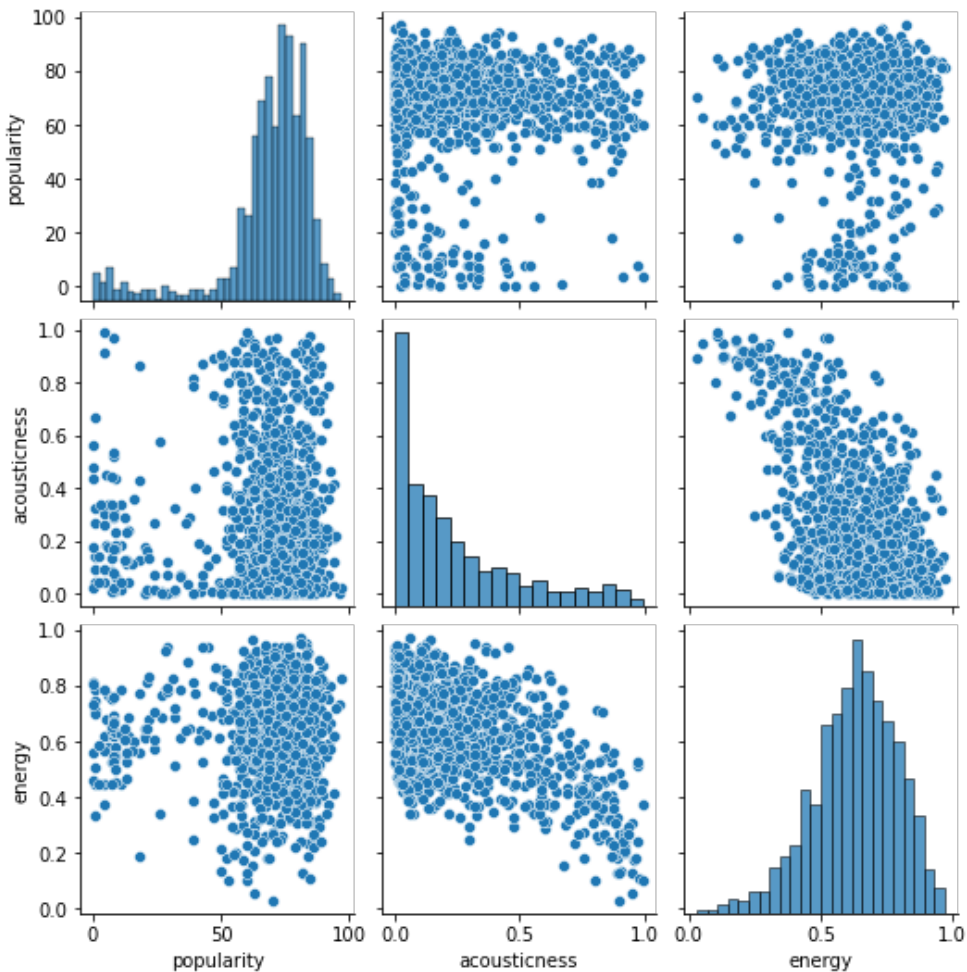
**Gráfico de bolhas.** É um gráfico de dispersão com três dimensões definidas por variáveis numéricas como entrada: duas estão nos eixos  $x$  e  $y$ , e a terceira no tamanho dos pontos. O código a seguir usa a função `seaborn.scatterplot()` para analisar a popularidade das músicas em relação a sua duração e nível de energia. O parâmetro `size` controla o tamanho do círculo de acordo com uma variável numérica (dado `energy`).

```
[5]: fig, ax = plt.subplots(figsize=(5, 5))
sns.scatterplot(
    data=df_hits,
    x="popularity",
    y="duration_ms",
    size="energy",
    alpha=0.5,
    sizes=(5, 100))
plt.legend(loc='upper left', fontsize=13)
ax.set_xlabel('Popularidade')
ax.set_ylabel('Duração (ms)');
```



**Correlogramas.** Ou matriz de correlação, permite analisar a relação entre pares de variáveis numéricas de um conjunto de dados. Correlogramas são muito úteis na Análise Exploratória, pois permitem visualizar as relações de todo o conjunto de dados de uma só vez. A relação entre cada par de variáveis é geralmente mostrada com um gráfico de dispersão, enquanto a diagonal representa a distribuição de cada variável, usando um histograma ou um gráfico de densidade. No próximo exemplo, visualizamos três variáveis utilizando a função `seaborn.pairplot()`: **popularity**, **acousticness** e **energy**.

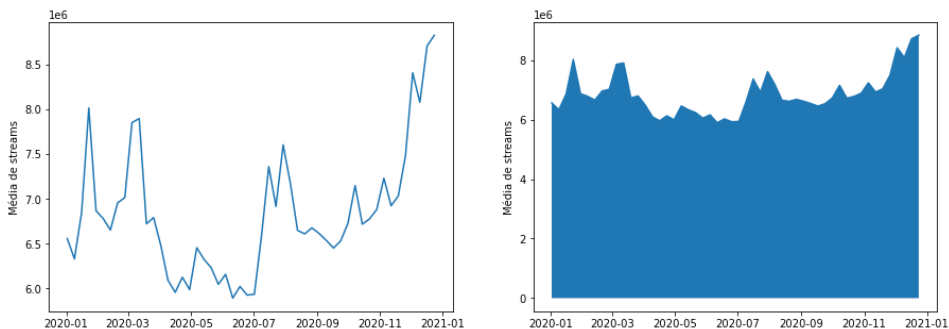
```
[6]: df = df_hits[['popularity', 'acousticness', 'energy']]
sns.pairplot(df); # criando um correlograma
```



## 1.6.4. Evolução

A visualização de uma evolução permite analisar uma tendência nos dados ao longo de intervalos de tempo – uma série temporal. Por exemplo, pode-se visualizar a evolução de músicas solo vs. colaborações ou a média de *streams* das músicas mais populares em 2020. Nesses casos, gráficos de linha e de área são frequentemente utilizados, sendo semelhantes a um gráfico de dispersão, porém os pontos de medição são ordenados e unidos com segmentos de linha reta. O exemplo a seguir mostra um gráfico de linha e outro de área para visualizar a evolução da média de *streams* no ano de 2020. O código utiliza as funções `matplotlib.pyplot.plot()` e `fill_between()`, respectivamente.

```
[5]: fig, ax = plt.subplots(1, 2, figsize=(15, 5))
ax[0].plot('chart_week', 'streams', '-', data=df) # gráfico de dispersão
ax[0].set_ylabel("Média de streams")
ax[1].plot('chart_week', 'streams', '-', data=df) # gráfico de área
ax[1].fill_between('chart_week', 'streams', data=df)
ax[1].set_ylabel("Média de streams");
```

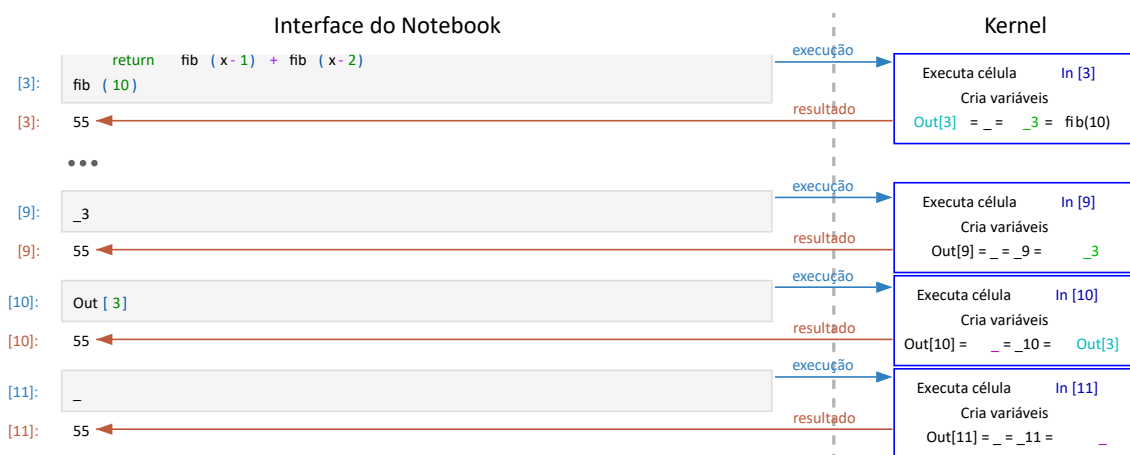


## 1.7. Jupyter Avançado

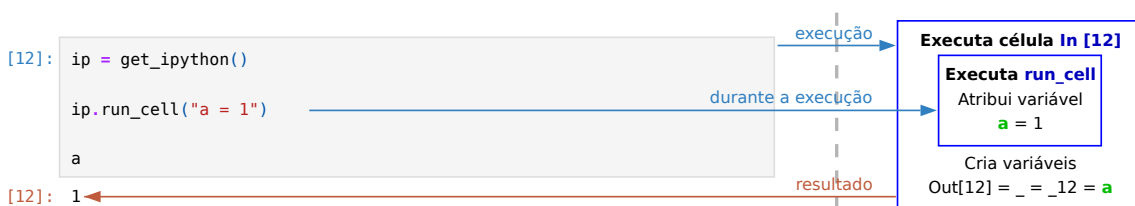
Além do já apresentado, o Jupyter possui características avançadas que permitem tanto um uso mais aprofundado de sua interatividade quanto de suas visualizações para análise de dados. Essas características estão presentes em diversas bibliotecas existentes e também podem ser usadas para estender o Jupyter. Esta seção tem objetivo de se aprofundar em características avançadas do Jupyter que permitam a criação de extensões.

### 1.7.1. IPython

A execução de códigos no Jupyter acontece através de um *kernel*. Ao executar um código na interface web, o Jupyter envia uma mensagem para o *kernel*, que trata da execução, mantém o ambiente de execução, e devolve o resultado em algum formato suportado. Existem *kernels* de diversas linguagens para Jupyter e até mesmo linguagens com múltiplas implementações de *kernel*. Para Python, o *kernel* mais usado é o IPython, que fornece um superconjunto da linguagem Python com operações orientadas ao Jupyter e visualizações de diferentes formatos. Após a execução de cada célula, o IPython popula variáveis para permitir que resultados de células anteriores sejam acessados e visualizados. O exemplo a seguir continua o exemplo da Figura 1.3, mostrando a execução de células no *kernel* e a população de variáveis. Perceba que é possível acessar o resultado da célula 3 tanto pela variável `_3` (célula [9]) quanto pelo acesso `Out[3]` (célula [10]). Além disso, se a célula tiver acabado de ser executada, é possível acessar seu resultado pela variável `_`, como ocorre na última célula da figura.



O *kernel* também pode ser diretamente acessado através da função `get_ipython` para a realização de funções mais avançadas. O próximo exemplo apresenta uma célula que executa essa função para obter o objeto do *kernel* e executa a função `run_cell` para executar uma célula diretamente no *kernel* sem uma célula associada no notebook. Esta célula define a variável `a` com valor 1. Por fim, a célula original termina sua execução acessando o valor de `a` recentemente definido.



Além de controlar a execução do código Python em notebooks e de criar variáveis após a execução de células, o IPython também estende a linguagem para adicionar expressões *bang*, “mágicas” de linha e de célula e consultas à documentação, como exemplificado no próximo notebook. Este notebook utiliza todos esses recursos para: listar arquivos no diretório de *datasets* (*bang* na célula [2]); verificar o histórico de execução de células (mágica de linha na célula [3] – perceba que o notebook começa pela célula [2]); mostrar a quantidade de linhas em cada arquivo do *dataset* (atribuição de *bang* na célula [4]); criar links em HTML para os arquivos do *dataset* (mágica de célula em [5]); e consultar a documentação da função `len` (interrogação na célula [6]). Apesar de ser uma extensão à sintaxe padrão do Python, todas essas operações são transformadas em código Python pelo IPython imediatamente antes da execução.

Uma *expressão bang* é uma expressão que executa um comando no sistema e retorna a saída padrão do comando como resultado. Ela pode ser usada com uma exclamação precedendo o resto do comando na linha. Por exemplo, a expressão `!ls` usada na célula [2] do exemplo anterior retorna todos os arquivos do diretório especificado em um sistema Unix. Note que `ls` é um comando que não existe normalmente no Windows. Portanto, a execução dessa célula em um sistema operacional Windows resultaria em erro. Para executar essa operação, o IPython transforma `!ls` em `get_ipython().system('ls')`.

```
[2]: !ls ../dataset/

spotify_artists_info_complete.tsv  spotify_hits_dataset_complete.tsv
spotify_charts_complete.tsv

[3]: %history -n

1: !ls
2: !ls ../dataset/
3: %history -n

[4]: files = !ls ../dataset/ # Atribui lista de arquivos a variável files
for name in files: # Percorre arquivos, pegando nome e número de linhas
    print(name, '--', len(open("../dataset/" + name).readlines()))

spotify_artists_info_complete.tsv -- 626
spotify_charts_complete.tsv -- 10401
spotify_hits_dataset_complete.tsv -- 1285

[5]: %%html
<a href="../dataset/spotify_artists_info_complete.tsv">artists</a>
<a href="../dataset/spotify_charts_complete.tsv">charts</a>
<a href="../dataset/spotify_hits_dataset_complete.tsv">hits</a>

artists charts hits

[6]: len?

Signature: len(obj, /)
Docstring: Return the number of items in a container.
Type:      builtin_function_or_method
```

Uma *mágica* do IPython tem o objetivo de modificar a forma de executar operações. Existem dois tipos de mágicas: mágica de linha e de célula. Uma mágica de linha altera o restante da linha e é usada de forma semelhante a expressões *bang*, com o símbolo `%` precedendo o nome da mágica. Por exemplo, a mágica `%history -n` usada na célula [3] do exemplo anterior exibe o histórico de células executadas com a numeração de células. Perceba que o resultado também apresenta o histórico da célula [1], que já não existe mais no notebook. Para executar essa mágica de linha, o IPython transforma `%history -n` em `get_ipython().run_line_magic('history', '-n')`.

Expressões *bang* e mágicas de linha também podem ser atribuídas a variáveis do Python, como ocorre na segunda linha da célula [4] do exemplo anterior. Para essa atribuição, é necessário retornar um valor ao invés de simplesmente imprimir na tela. Portanto, o IPython transforma `files = !ls` em `files = get_ipython().getoutput('!ls')`. O objeto retornado por `getoutput` funciona como uma lista de linhas, mas possui métodos especiais de visualização e transformação para o uso no notebook.

Uma mágica de célula altera a célula inteira e deve ser usada no topo da célula com `%%` precedendo o nome da mágica. A mágica `%%html` permite escrever código HTML que é exibido diretamente no navegador, mesmo que a linguagem do notebook seja Python. Na célula [5] do exemplo anterior, essa mágica é usada para criar links para os arquivos do *dataset*, que são normalmente carregados pela aplicação do notebook. Para a executar esta célula, o IPython a transforma em `get_ipython().run_cell_magic('html', '', '<a>...</a>\n')`.

O IPython oferece diversas mágicas de linha e de célula e também permite definir novas mágicas como apresentado na Seção 1.7.2. Para obter uma lista de mágicas disponíveis, é possível usar a mágica `%lsmagic`. A Tabela 1.4 apresenta as 10 mágicas de linha e as 10 mágicas de célula mais usadas em notebooks no GitHub. Para construir

**Tabela 1.4. Mágicas mais usadas em um conjunto de 198.374 notebooks.**

Mágica	Notebooks	Descrição
<code>%matplotlib</code>	156.353	Configura matplotlib para funcionar interativamente
<code>%load_ext</code>	14.216	Carrega uma extensão IPython pelo nome do módulo
<code>%autoreload</code>	12.920	Recarrega módulos automaticamente
<code>%pylab</code>	8.183	Carrega numpy e matplotlib
<code>%time</code>	5.172	Mede a duração da execução de uma expressão
<code>%config</code>	4.805	Configura o IPython
<code>%pinfo</code>	3.253	Acessa a documentação de um objeto
<code>%run</code>	2.474	Executa um arquivo Python dentro do IPython
<code>%timeit</code>	2.318	Tira a mediana de várias execuções de <code>%time</code>
<code>%reload_ext</code>	1.635	Recarrega uma extensão IPython
<code>%%time</code>	13.129	O mesmo que <code>%time</code> , mas para a célula
<code>%%html</code>	2.944	Exibe célula como HTML
<code>%%writefile</code>	2.937	Escreve o conteúdo de uma célula em um arquivo
<code>%%bash</code>	2.850	Executa a célula com bash em um subprocesso
<code>%%timeit</code>	2.628	O mesmo que <code>%timeit</code> , mas para a célula
<code>%%javascript</code>	2.447	Executa a célula em Javascript no navegador
<code>%%sql</code>	1.822	Realiza consulta SQL
<code>%%R</code>	1.285	Executa a célula com R em um subprocesso
<code>%%capture</code>	1.131	Captura stdout, stderr e IPython display da execução
<code>%%cython</code>	1.117	Compila e importa todas as funções definidas em cython

essa tabela, foram usados dados de um estudo que coletou 1.450.071 de notebooks do GitHub até abril de 2018 [Pimentel et al. 2021]. Apesar de ter coletado essa quantidade inicial de notebooks, após a seleção de apenas notebooks de repositórios que continuavam existindo em julho de 2020, o descarte de duplicatas, e o descarte de notebooks com erros de sintaxe ou escritos outras linguagens, a quantidade total de notebooks em Python usados para a análise foi 886.668. Desses, 173.256 notebooks utilizam mágicas de linha e 33.541 notebooks utilizam mágicas de célula. Perceba na tabela que a mágica de linha mais usada é a `%matplotlib`, que serve para configurar a biblioteca *matplotlib* para funcionar interativamente no notebook. Já a mágica de célula mais usada é a `%%time`, que serve para medir o tempo de execução da célula.

Por fim, o IPython estende a sintaxe do Python para permitir o acesso a documentações e definições de funções e classes através dos símbolos `?` e `??`, respectivamente. Por exemplo, o comando `len?` da célula [6] do exemplo anterior apresenta a função *len* com sua assinatura, documentação e tipo. Esses símbolos podem ser usados com qualquer função, classe, módulo, ou variável do Python e com qualquer mágica do IPython.

Além de executar células e de estender a sintaxe para melhorar a interatividade, o *kernel* também tem a função de fornecer métodos para visualizações ricas no Jupyter. As seções anteriores apresentaram diversos gráficos e tabelas em notebooks como visualizações ricas. Dentre os formatos personalizados, é possível exibir imagens em PNG, JPEG, SVG e PDF, equações em  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ , texto formatado com Markdown, Javascript executável no navegador e tabelas em HTML. No IPython, objetos são exibidos em duas situações: quando a função `display` é invocada, ou quando o objeto é o resultado da célula (i.e., úl-



tima expressão). O formato da visualização rica depende dos métodos implementados para isso em cada objeto, como apresentado na próxima seção.

## 1.7.2. Estendendo o IPython

Esta seção mostra como é possível estender o IPython tanto para definir novas mágicas quanto para definir novas visualizações para cada tipo de objeto.

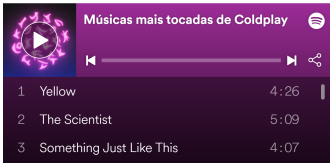
**Mágicas.** Para construir novas mágicas, é possível definir uma classe de mágicas em que cada método é uma mágica de linha ou de célula. Esses métodos recebem os argumentos da mágica como parâmetro `line` e o código da célula como parâmetro `cell`, e retornam o resultado da operação. O código a seguir mostra o exemplo de uma mágica `%%artist` que exibe um tocador do Spotify para um registro de artista (i.e., a banda Coldplay).

```
[1]: import pandas as pd
    dfa = pd.read_csv("../dataset/spotify_artists_info_complete.tsv", sep="\t")
    dfa.loc[0]
```

```
[1]: artist_id          4gzpq5DPGxSnKTe4SA8HAU
    name                Coldplay
    followers           29397183
    popularity          90
    genres              ['permanent wave', 'pop']
    image_url           https://i.scdn.co/image/4ffd6710617d289699cc0d...
    Name: 0, dtype: object
```

```
[2]: from IPython.core.magic import Magics, magics_class, line_magic
    EMBED_URL = ('<iframe src="https://open.spotify.com/embed/{type_}/{id}" width="{w}" height="{h}"
    ' frameborder="0" allowtransparency="true" allow="encrypted-media"></iframe>')
    @magics_class # Define classe de mágicas
    class SpotifyMagics(Magics):
        @line_magic # Define mágica de linha
        def artist(self, line): # Executa mágica de célula no kernel (self.shell) para exibir HTML
            return self.shell.run_cell_magic("html", '', EMBED_URL.format(id=line, type_="artist", w=360, h=180))
    get_ipython().register_magics(SpotifyMagics) # Cadastra mágica
```

```
[3]: %%artist 4gzpq5DPGxSnKTe4SA8HAU
```



Considerando as células do código anterior, a primeira importa a biblioteca *pandas*, o arquivo de artistas e a informação do artista na posição 0. Nessa célula o dado mais relevante é o identificador do artista (i.e., `artist_id`). Em seguida, a segunda célula define uma mágica de linha com o nome `%%artist` que executa a mágica de célula `%%html` para exibir um `IFrame` com o tocador do Spotify, usando o identificador do artista passado como parâmetro para a mágica (i.e., argumento `line` do método `artist`). Para definir essa mágica de linha, a célula decorou o método `artist` com `line_magic` (i.e., precede o comando), decorou a classe `SpotifyMagics` com `magics_class`, e registrou a(s) mágica(s) dessa classe no kernel com a função `register_magics`. Por fim, a terceira célula do notebook utilizou a mágica definida passando o id do artista obtido na primeira célula e exibiu um tocador. Com esse tocador, é possível ouvir as músicas dos artistas que foram analisados anteriormente.

Essa mágica está restrita a um único notebook. Para reusá-la em mais de um notebook, cria-se uma extensão para o IPython em um módulo externo: um arquivo Python com o código da mágica e com uma função `load_ipython_extension` encarregada de registrá-la. O próximo exemplo define um módulo IPython no arquivo `spotify.py` para a mágica `%artist` e uma nova mágica `%track`, que atua como mágica de linha e como mágica de célula para exibir tocadores de música. Para isso, essa mágica foi decorada com `line_cell_magic`.

```
from IPython.core.magic import Magics, magics_class, line_magic, line_cell_magic
from IPython.core.magic_arguments import parse_argstring, magic_arguments, argument
from IPython.display import HTML
EMBED_URL = ('<iframe src="https://open.spotify.com/embed/{type_}/{id}" width="{w}" height="{h}"'
            ' frameborder="0" allowtransparency="true" allow="encrypted-media"></iframe>')

# Define classe de mágicas
@magics_class
class SpotifyMagics(Magics):
    # Mágica artist definida anteriormente
    @line_magic
    def artist(self, line):
        return HTML(EMBED_URL.format(id=line, type_="artist", w=360, h=180))

    # Nova mágica track
    @magic_arguments() # Define argumentos para mágica
    @argument("song_ids", nargs="*", help="Lista de músicas")
    @argument("-w", "--width", type=int, default=300, help="Largura")
    @argument("-h", "--height", type=int, default=80, help="Altura")
    @line_cell_magic # Define mágica de linha e de célula
    def track(self, line, cell=""):
        # Carrega argumentos
        args = parse_argstring(self.track, line)
        # Usa lista de ids do parâmetro ou linhas da célula
        ids = args.song_ids or cell.split("\n")
        # Retorna tocadores separados por <br>
        return HTML("<br>".join(
            EMBED_URL.format(id=song_id, type_="track", w=args.width, h=args.height)
            for song_id in ids if song_id # Um tocador para cada música
        ))

# Carrega extensão
def load_ipython_extension(kernel):
    kernel.register_magics(SpotifyMagics) # Cadastra mágicas
```

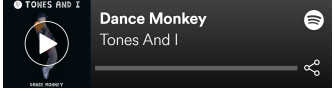
Outra diferença da mágica `%track` para a mágica `%artist` está na definição de argumentos. A mágica `%track` aceita uma lista de músicas e o tamanho do tocador como argumentos. Esses argumentos são configurados pelos decoradores `magic_arguments` e `argument`. Além disso, eles são usados pela função `parse_argstring`. Por fim, ao invés de usar a mágica de célula `%%html` para exibir o IFrame do tocador, a função `track` usa diretamente um objeto do tipo `HTML`, que define a própria forma de visualização HTML. Apesar desta alteração não ser estritamente necessária, ela foi realizada em ambas as funções para permitir o uso destas mágicas na extensão de visualizações ricas, como a seguir.

Para carregar a extensão, usa-se a mágica `%load_ext spotify`, como no próximo exemplo. A primeira célula carrega a extensão e os dados, e a segunda usa a mágica de linha `%track` para exibir o tocador de uma música. Essa música é definida por um código Python que é executado por conta das chaves nos argumentos. Caso os argumentos fossem passados sem as chaves, seriam interpretados como texto. A última célula possui a mágica de célula `%%track`, que cria tocadores para cada linha da célula.

```
[1]: %load_ext spotify
import pandas as pd
dfc = pd.read_csv("../dataset/spotify_charts_
                "complete.tsv", sep="\t")
dfc.loc[0, "song_id"]

[1]: '1rgnBhdG2JDFtBYkYRZAku'
```

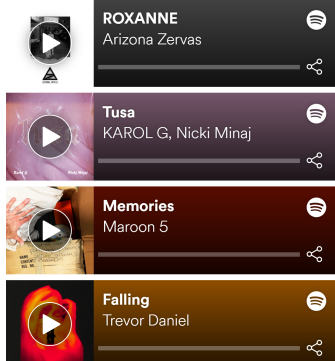
```
[2]: %track {dfc.loc[0, "song_id"]} -h 80
```



```
[3]: dfc.loc[1:4, "song_id"]
```

```
[3]: 1    696Dn1kuD0XcMAnK1TgXXK
     2    7k4t7uLgt0xPwTpFmtJNTY
     3    2b8f0ow8UzyDFAE27Yh0ZM
     4    4TnjEaw0eW0eKTKIEvJyCa
     Name: song_id, dtype: object
```

```
[4]: %%track -h 80
696Dn1kuD0XcMAnK1TgXXK
7k4t7uLgt0xPwTpFmtJNTY
2b8f0ow8UzyDFAE27Yh0ZM
4TnjEaw0eW0eKTKIEvJyCa
```



**Visualizações ricas.** Como mencionado, o Jupyter suporta a visualização de textos, erros, imagens, HTML com Javascript e Markdown. Porém, o Python em si só possui a representação textual de seus objetos. Para aproveitar as visualizações ricas do Jupyter, pode-se usar a função `display` do IPython, passando o valor a ser exibido e o tipo da exibição (i.e., SVG, PNG, JPEG, HTML, Javascript, LaTeX). Também pode-se definir métodos especiais na forma `__repr__<tipo>` em classes para retornarem cada tipo de visualização. Por exemplo, a definição do método `__repr__png__` permite retornar o conteúdo binário de uma imagem PNG para exibição no Jupyter. Ainda, é possível definir um único método `__repr__mimebundle__` para retornar todas as visualizações suportadas pelo objeto.

O próximo exemplo define esses métodos em duas classes, `Artist` e `Track`. Essas classes recebem como parâmetro de inicialização uma linha dos *Dataframes* instanciadas anteriormente (i.e., `dfa` e `dfc`). A classe `Artist` define o método `__repr__` para visualização padrão no Python, o método `__repr__html__` para visualização do tocador em HTML, e o método `__repr__markdown__` para visualização de texto formatado em Markdown. Já a classe `Track` define um único método `__repr__mimebundle__`, que retorna visualizações nesses mesmos formatos.

Por padrão, o Jupyter exibe apenas um dos formatos disponíveis seguindo a ordem HTML, Markdown, LaTeX, SVG, imagem, JavaScript, texto. Note que as células [4] e [8] desse exemplo exibiram a visualização HTML dos objetos `artist` e `track` ao retornarem esses objetos. Contudo, é possível forçar a visualização de outros formatos usando a função `display`. As células [3] e [6] permitem visualizar esses objetos como Markdown.

Note que a função `__repr__mimebundle__` de `Track` define um formato adicional, `trk.spotify+json`, que a célula [7] tenta visualizar. Esse formato adicional não é suportado pelo Jupyter por padrão, portanto não é exibido. Entretanto, é possível criar extensões para a interface do Jupyter – tanto do Notebook quanto do Lab – para suportar novos formatos.<sup>7</sup> A criação de extensões para a interface requer a definição de arquivos externos relativamente grandes e está fora do escopo deste capítulo.

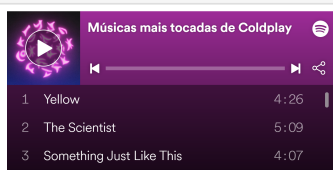
<sup>7</sup><https://github.com/jupyterlab/mimerender-cookiecutter-ts>

```
[2]: class Artist:
      def __init__(self, row):
          self.row = row
      def __repr__(self):
          return str(self.row['name'])
      def _repr_html_(self):
          html = %artist {self.row.artist_id}
          return html.data
      def _repr_markdown_(self):
          return f"# {self.row['name']}"
artist = Artist(dfa.loc[0])
```

```
[3]: display(artist, include=["text/markdown"])
```

## Coldplay

```
[4]: artist
```



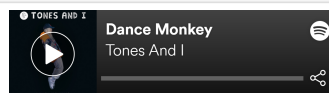
```
[5]: class Track:
      def __init__(self, row):
          self.row = row
      def _repr_mimebundle_(self, **kwargs):
          row = self.row
          html = %track {row.song_id}
          return {
              'text/markdown': f"{row.artist} - "
                               f"{row.track_name}",
              'text/html': html.data,
              'trk.spotify+json': row.to_json(),
              'text/plain': row.track_name
          }
track = Track(dfc.loc[0])
```

```
[6]: display(track, include=["text/markdown"])
```

Tones And I - Dance Monkey

```
[7]: display(track, include=["trk.spotify+json"])
```

```
[8]: track
```



### 1.7.3. Widgets

Finalmente, uma outra forma de interagir com o Jupyter é a partir de *widgets* interativos. *Widgets* utilizam as visualizações ricas do IPython e do Jupyter para interatividade e podem ser usados para fazer formulários, dashboards e até mesmo variar rapidamente parâmetros de funções em análises.

O próximo exemplo mostra um formulário para consultar diversos percentis de uma coluna de um *DataFrame* de forma interativa. Tal formulário é criado em cima da função `percentil`, que recebe a coluna e o percentil como parâmetros e imprime qual é o valor determinado para o percentil. Para transformar essa função em um formulário composto de *widgets*, usa-se o decorador `@interact` da biblioteca `ipywidgets`. Esse decorador tenta criar um formulário com um *widget* para cada argumento da função decorada, identificando o tipo mais provável dos atributos padrões.

```
[2]: from ipywidgets import interact, interact_manual, IntSlider
      @interact(q=IntSlider(value=50, min=0, max=100))
      def percentil(column="duration_ms", q=10):
          print(f"Percentil {q} de {column} é {df[column].quantile(q=q/100)}")
```

```
column duration_ms
q 50
Percentil 50 de duration_ms é 193683.0
```

Note que foi criado um *widget* de texto para o argumento `column` que está com um valor texto como padrão. Além disso, o decorador `@interact` aceita parâmetros próprios para definir formatos diferentes para determinados argumentos. O código do exemplo define o parâmetro `q` como controlado por um *widget* de controle deslizante, `IntSlider`, com valor inicial 50, mínimo 0 e máximo 100.

Ao executar a célula, o formulário é exibido com diversos *widgets* e quem estiver usando o notebook pode interagir com eles, deslizando o controle do argumento `q` ou

escrevendo um nome de coluna diferente para o argumento `column`. Assim que qualquer valor é alterado, a função `percentil` é executada com os valores atualizados, e o resultado da execução imediatamente atualizado. Isso permite maior interatividade na análise.

Em algumas situações, a função computada pelo formulário é demorada e não deseja-se executá-la automaticamente a cada pequena alteração em valores. Para utilizar *widgets* nesse caso, é possível substituir o decorador `interact` por `interact_manual`. Esse decorador cria um *widget* de botão no final do formulário para executar a função decorada apenas quando todos os parâmetros forem ajustados.

Nem sempre uma única função decorada é o suficiente para construir a interatividade desejada em formulários e *dashboards*. O próximo exemplo apresenta um formulário em que a pessoa que está interagindo com o notebook deve primeiro escolher uma música do conjunto de dados e depois uma artista que esteja associada à música. Com esse objetivo, a célula [2] define dois campos de seleção com busca (`Combobox cs` para música e `ca` para artista), dois campos de exibição para tocadores (`Output out_song` para música e `out_artist` para artista), e uma caixa vertical para a exibição do formulário como um único *widget* (`VBox`). Este *widget* não é exibido de imediato.

```
[2]: from ipywidgets import Combobox, Output, VBox
# Cria itens para comboboxes
songs = df.song_name.to_list()
artists = []
# Cria form com 2 campos e 2 saídas
cs = Combobox(description="Música", options=songs)
out_song = Output()
ca = Combobox(description="Artista", options=[""])
out_artist = Output()
widget = VBox([cs, out_song, ca, out_artist])
```

```
[4]: # Chama função ao alterar o artista
@ca.observe
def change_artist(w):
    # Limpa saída de artista
    out_artist.clear_output()
    # Seleciona artista
    v = ca.value
    if not v or v not in ca.options:
        return
    index = ca.options.index(v)
    # Exibe artista
    with out_artist:
        html = %artist {artists[index]}
        display(html)
```

```
[3]: # Chama função ao alterar a musica
@cs.observe
def change_song(w):
    global artists
    # Limpa saídas e seleção ca
    out_song.clear_output()
    out_artist.clear_output()
    ca.value = ""
    artists = []
    # Seleciona linha da música
    if cs.value not in songs:
        return
    index = songs.index(cs.value)
    row = df.loc[index]
    # Exibe música
    with out_song:
        html = %track {row.song_id} -w 360
        display(html)
    # Atualiza opções de artista
    artists = eval(row.artist_id)
    ca.options = tuple(eval(row.artist_name)+[""])
```

```
[5]: widget
```

A célula [3] define uma função para observar mudanças de valores no campo `cs` (i.e., música). Toda vez que uma música é selecionada, a função `change_song` é chamada. Essa função apaga o conteúdo dos campos de exibição `out_song` e `out_artist` utilizando o método `clear_output` e verifica se o nome da música existe no *DataFrame* usado. Se existir, a função exibe o tocador da música utilizando a mágica `%track` no gerenciador de contexto `with out_song`, e atualiza o *Combobox* `ca` com a lista de artistas da música.

De forma semelhante, a célula [4] define uma função para observar mudanças de valores no campo `ca` (i.e., artista). Quando o campo de artista é alterado, a função `change_artist` é chamada e apaga o campo de exibição de artista e verifica a existência de artista no campo de seleção. Se existir, a função usa a mágica `%artist` para exibir o tocador de artista no local apropriado através do gerenciador de contexto `with out_artist`. Por fim, a célula [5] retorna o *widget* resultante e permite a interação com os campos definidos.

## 1.8. Ciência Aberta com Jupyter

Por ser usado em experimentos e análises de dados, existe a necessidade de que notebooks do Jupyter sejam reprodutíveis e estejam disponíveis em plataformas abertas para consulta e reprodução de experimentos [Perkel 2021a]. Esta seção trata tanto do compartilhamento quanto do desenvolvimento de notebooks reprodutíveis.

### 1.8.1. Compartilhamento de Notebooks

Arquivos de notebooks possuem código e resultados de execução de notebook. Por conterem código, é natural que se usem plataformas de compartilhamento e versionamento de código, como GitHub. Essas plataformas utilizam o git ou outro sistema de versionamento para gerenciar diferentes versões do código e chegam a implementar a visualização de notebooks em suas interfaces web.

Apesar de permitirem o armazenamento de arquivos de notebooks, ferramentas como git podem não ser a escolha ideal para experimentos científicos por diversos motivos. Primeiro, elas tentam realizar operações de *diff* em arquivos de texto para reduzir a sobrecarga do armazenamento de versões e permitir o desenvolvimento paralelo. Como notebooks são armazenados em arquivos de texto (JSON) que também possuem resultados binários (imagens) dentro, a realização do *diff* pode falhar e induzir ao erro no uso habitual de notebooks. Ferramentas como *nbdime*<sup>8</sup> podem reduzir o problema com operações de *diff* e *merge*, mas exigem que cientistas as usem ativamente por fora do uso padrão do git. Segundo, experimentos científicos costumam ter grandes conjuntos de dados de entrada que precisam ser compartilhados para garantir a reprodutibilidade. Por ser projetado para fazer versionamento de código, o git não lida bem com um volume grande de dados e plataformas como GitHub restringem o tamanho de arquivos e de repositórios, inviabilizando o uso. Por fim, não existe uma garantia de que páginas do GitHub se manterão sem mudanças ao longo do tempo.

Dessa forma, ao citar que um repositório possui dados do experimento, existe a possibilidade do repositório versionado de código evoluir e deixar de representar o experimento citado em um artigo. Esse problema pode ser amenizado com a citação de uma versão específica, mas requer um trabalho a mais de cientistas marcarem a versão.

Por conta dessas limitações, repositórios de acesso aberto como figshare<sup>9</sup> e Zenodo<sup>10</sup> foram criados especificamente para permitir o compartilhamento de dados científicos, notebooks e outros artefatos digitais. Apesar de permitirem o versionamento dos dados, essas plataformas não foram projetadas para o desenvolvimento contínuo de ver-

---

<sup>8</sup><https://nbdime.readthedocs.io/en/latest/>

<sup>9</sup><https://figshare.com/>

<sup>10</sup><https://zenodo.org/>

sões. Ao invés disso, elas promovem o compartilhamento de versões finais dos experimentos, com a indicação de licenças apropriadas e documentação de como reproduzir. Após a confirmação do envio, essas plataformas geram números de DOI que podem ser citados em artigos com a garantia de que não ocorrerão mudanças.

### 1.8.2. Reprodução e Boas Práticas

A análise exploratória em notebooks ajuda a entender melhor os dados e fazer mudanças parciais sem ter que reexecutar toda a análise. Por outro lado, esse modo de trabalhar em notebooks gera células fora de ordem e estados ocultos de execução que podem comprometer a reprodutibilidade de experimentos. Além disso, a falta de inclusão dos dados de entrada em repositórios (ver seção anterior) e falta de rastreamento de bibliotecas usadas no momento do desenvolvimento também podem comprometer a reprodutibilidade.

Esta parte do capítulo tem como base um estudo de reprodutibilidade que coletou mais de um milhão de notebooks do GitHub e observou taxas baixas de reprodução ao tentar executar notebooks com código Python em diferentes ambientes e diferentes ordens [Pimentel et al. 2021]. A taxa de execuções que foram capazes de executar todas as células variou de 22,57% a 26,09%. Já a taxa de execuções que produziram os resultados próximos aos armazenados nos notebooks variou de 4,9% a 15,04%, após normalizações de resultados. Esse estudo evidenciou que não basta usar notebooks para fazer Ciência de Dados com reprodutibilidade. É necessário ter um trabalho organizado e seguir boas práticas como as propostas a seguir [Pimentel et al. 2019].

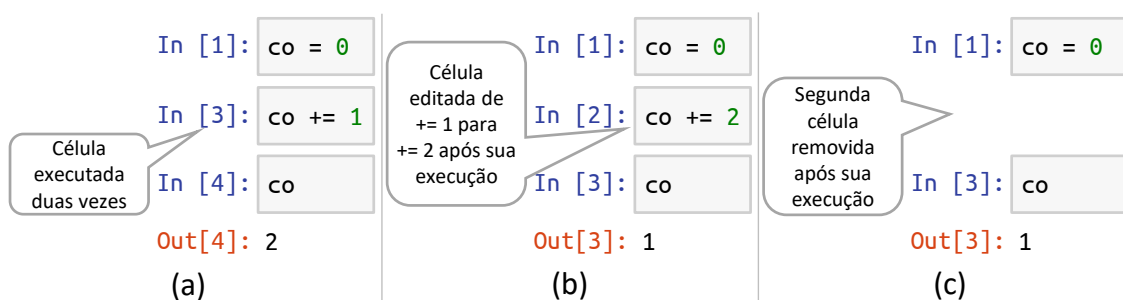
1. **Usar títulos descritivos com um conjunto restrito de caracteres (A-Z a-z 0-9 . \_ -) e células Markdown com títulos mais detalhados no corpo.** No Jupyter, o nome de um arquivo de notebook representa seu título. Esse título muitas vezes é importante para determinar não só o assunto do notebook como também a ordem de execução dos notebooks em projetos com muitos arquivos. Portanto, recomenda-se o uso de títulos descritivos para que isso seja possível. Contudo, alguns sistemas operacionais não suportam todo tipo de caractere em seus sistemas de arquivo. Por exemplo, o uso do caractere de dois pontos pode causar problemas no Windows e no MacOS. Dessa forma, para garantir a reprodutibilidade em diversos sistemas, recomenda-se apenas o uso de um conjunto restrito de caracteres no título, como o conjunto definido no guia de nomes de arquivos completamente portáteis do POSIX [Lewine 1991]. O título completo do notebook pode ser escrito dentro do documento através de células Markdown.
2. **Prestar atenção ao final do notebook e verificar se células do final foram executadas e se não cabe nenhuma célula de Markdown para concluir a análise.** Durante a análise de notebooks do GitHub [Pimentel et al. 2021], observou-se que o início do notebook costuma receber mais atenção, incluindo células de Markdown e células com resultados de execuções. Isso ocorre por essa parte estar sempre sendo revisitada para a inclusão de novos *imports* e reexecução de partes do notebook. Já o final de notebooks costuma ter mais células de código sem execução, menos células de Markdown, e poucos notebooks selecionados para a análise qualitativa possuíam células de Markdown com conclusões dos resultados das análises, apesar de um número considerável ter definido o que se desejava observar no início. Portanto, recomenda-se dar mais atenção ao final do notebook, para que a qualidade seja mantida do início ao fim.

3. **Abstrair funções, classes e módulos a partir do código e testá-los.** Por mais genérico que seja, o código de uma célula só costuma ser acessível no próprio notebook. Dessa forma, para reusar um determinado código, muitas vezes é necessário copiar e colar sua definição de um arquivo para outro, prejudicando o reuso e levando códigos a evoluírem de maneira independente sem compartilharem melhorias (i.e., é possível que uma versão do código receba uma correção de defeito e a outra receba uma funcionalidade nova). Além disso, a execução de testes automatizados em notebooks pode ser problemática por quebrar a narrativa que está sendo contada pela programação literária. Assim, recomenda-se a abstração de códigos em funções, classes e módulos que possam ser importados por diversos notebooks e testados externamente. Um exemplo de abstração foi apresentado na Seção 1.7.2 deste capítulo, ao apresentar a extração de uma classe de mágicas para um módulo externo. Essa abstração permitiu o reuso das mesmas mágicas em diferentes notebooks, sem a necessidade de copiar e colar definições de um notebook para outro. Por questão de escopo, não foram definidos testes automatizados para o módulo extraído, mas o uso de um módulo externo permite que isso seja feito de forma transparente sem quebrar a narrativa do notebook.
4. **Declarar dependências em arquivos apropriados e fixar a versão de módulos.** O estudo de reprodutibilidade de notebooks observou que muitos notebooks falhavam em executar até fim por falta da instalação de módulos [Pimentel et al. 2021]. Além disso, o estudo identificou que arquivos *requirements.txt* eram menos suscetíveis a falhas do que os outros formatos avaliados, *setup.py* e *Pipfile*. Dessa forma, recomenda-se definir dependências explicitamente em arquivos *requirements.txt* com suas respectivas versões fixas. Arquivos *setup.py* e *Pipfile* também podem ser usados se as pessoas que estão desenvolvendo o projeto considerarem mais apropriados ou com o aumento da maturidade de arquivos *Pipfile* – que ainda eram bem recentes quando o estudo de reprodutibilidade foi feito. Por exemplo, para este capítulo, foi usada a versão 1.1.3 do *pandas*. Para favorecer a reprodutibilidade, um arquivo *requirements.txt* poderia ter a linha `pandas==1.1.3`. Além desta declaração, também seria necessário declarar versões do *IPython* (7.19.0), *numpy* (1.19.2), *scipy* (1.5.2), *scikit-learn* (0.24.2), *fuzzywuzzy* (0.18.0), *matplotlib* (3.3.2), *seaborn* (0.11.0), *ipywidgets* (7.5.1), entre outras.
5. **Usar um ambiente limpo para testar dependências e verificar se todas estão declaradas corretamente.** Muitos projetos presumem um conjunto de dependências no sistema e não definem versões explícitas para elas em arquivos de dependências. O estudo inicial de reprodutibilidade [Pimentel et al. 2019] observou que instalar dependências em ambientes limpos falhou mais do que usar ambientes com anaconda, um gerenciador de pacotes que vem com uma distribuição Python e diversas bibliotecas científicas instaladas. De forma semelhante, a extensão do estudo [Pimentel et al. 2021] observou que usar ambientes inchados de dependências falhavam menos por *ImportError* e mais por atualizações em módulos. Dessa forma, recomenda-se configurar um ambiente limpo para testar as dependências dos notebooks antes de suas publicações, para verificar se todas as dependências estão declaradas corretamente.
6. **Colocar importações de módulos no início do notebook.** Essa recomendação é semelhante à recomendação da PEP 8 de se colocar *imports* no topo de arquivos [van Rossum et al. 2001]. Em notebooks, além dessa recomendação ajudar a organizá-los, pode ajudar na verificação de *imports* discutida anteriormente. Se todos os coman-



dos de *import* estiverem juntos, a verificação do ambiente pode ser feita de forma simplificada executando apenas o início do notebook. Note, entretanto, que isso não garante a reprodutibilidade do notebook, visto que versões diferentes de módulos podem permitir a execução do comando de *import*, mas gerar resultados diferentes.

7. **Usar caminhos relativos para acessar dados no repositório.** O estudo de reprodutibilidade [Pimentel et al. 2021] observou que o acesso a arquivos também foi uma causa comum de erros. Em muitas situações, isso aconteceu por arquivos de dados não estarem disponíveis no repositório; e em outras, ocorreu por notebooks tentarem acessar arquivos usando caminhos absolutos no sistema de arquivos. Usar caminhos relativos e disponibilizar dados de análise podem minimizar esse problema.
8. **Re-executar notebooks do início ao fim antes de enviá-los para algum repositório.** Como observado na análise [Pimentel et al. 2021], muitos notebooks possuem células fora de ordem e saltos na execução. Essas características podem induzir a problemas por estados ocultos de execução e reduzir a reprodutibilidade de notebooks. A Figura 1.5 apresenta três situações que ocorrem durante o desenvolvimento de notebooks que podem gerar problemas de estados ocultos. A re-execução de notebooks do início ao fim ajuda a restaurar contadores de execução e a verificar a existência de estados ocultos. Dessa forma, recomenda-se que a prática seja adotada antes do envio de notebooks a repositórios.



**Figura 1.5. Três tipos de estados ocultos: (a) Re-execução; (b) célula editada; (c) célula removida (adaptado de [Pimentel et al. 2021])**

Seguir algumas dessas boas práticas manualmente pode exigir um esforço cognitivo de estar sempre organizando o notebook. Algumas ferramentas foram propostas para diminuir esse esforço e melhorar a qualidade e reprodutibilidade de notebooks: Julynter [Pimentel et al. 2021], NBSafety [Macke et al. 2020], ReproZip [Chirigati et al. 2016], nbgather [Head et al. 2019] e Osiris [Wang et al. 2020].

Julynter [Pimentel et al. 2021] é uma extensão para Jupyter Lab feita com o objetivo de oferecer *linting* para notebooks. Essa ferramenta utiliza informações já coletadas durante a execução pelo kernel do IPython e as combina com análises estáticas de código-fonte para exibir recomendações em tempo de desenvolvimento para quem estiver escrevendo o notebook. As recomendações são diretamente baseadas nas recomendações descritas anteriormente, e sete das oito boas práticas estão cobertas pelas recomendações do Julynter. Apesar de suportar diversas recomendações, muitas não são completas o suficiente para todo tipo de problema. Por exemplo, o Julynter consegue detectar dependências

expressas por *imports* no Python, mas não consegue detectar o uso de programas externos. Além disso, o Julynter só pode fazer recomendações durante o desenvolvimento. Dessa forma, não é muito útil resolver a ordem de execução após fechar a sessão para auxiliar a reprodução ou para ajudar a execução de notebooks desenvolvidos por outras pessoas em outros momentos.

NBSafety [Macke et al. 2020] funciona de forma semelhante ao Julynter, coletando execuções de células durante o desenvolvimento, analisando o código estaticamente e apresentando recomendações para resolver problemas de estados ocultos e ordem de execução de células. Diferente do Julynter que possui recomendações variadas para diversas situações, NBSafety direciona as recomendações apenas a essas questões da execução e apresenta mais detalhes sobre elas. Além disso, NBSafety funciona tanto no Jupyter Notebook quanto no Jupyter Lab, enquanto que Julynter só funciona no Jupyter Lab.

Para tratar a questão de dependências, o ReprZip [Chirigati et al. 2016] captura dependências automaticamente durante a execução. O ReprZip é capaz de capturar não só bibliotecas escritas em Python, mas também programas externos e dados usados pelos notebooks. Após a execução de um notebook, o ReprZip cria pacotes com essas dependências que podem ser usados para reprodução em outros ambientes.

Em relação à ordem de execução e à presença de estados ocultos, tanto nbgather [Head et al. 2019] quanto o kernel do noWorkflow [Pimentel 2021] são capazes de coletar todas as execuções de células que ocorrem durante o desenvolvimento para a reconstrução de um notebook organizado posteriormente. Essas ferramentas também possuem recursos para limpar a execução, removendo células do histórico que não contribuem para o resultado final. A nbgather realiza essa operação de forma estática, observando nomes que aparecem em cada célula executada. Já o noWorkflow faz a coleta de forma dinâmica, identificando dependências reais que existem entre variáveis.

Por fim, muitas vezes essas ferramentas não foram usadas durante o desenvolvimento e é necessário reproduzir notebooks seguindo a ordem correta. Para resolver esses problemas, Osiris [Wang et al. 2020] busca reordenar as células de um notebook para uma ordem reprodutível. Para isso, Osiris analisa a ocorrência de definição e uso de nomes em cada célula do notebook e reordena a execução de forma a minimizar a presença de células fora de ordem. Osiris atinge uma taxa de reprodução de 82,23% em notebooks que não possuem problemas de dependência.

## 1.9. Considerações Finais

A necessidade de transformar dados em informações úteis de forma fácil e rápida tornou a Ciência de Dados um dos tópicos mais relevantes não só em Computação mas também nas mais diversas Ciências. O principal objetivo dessa área de pesquisa interdisciplinar é extrair conhecimento não-trivial a partir de dados brutos. Através de técnicas e algoritmos de mineração de dados e aprendizado de máquina, é possível detectar padrões e obter *insights* valiosos sobre informações múltiplas, de forma com que o valor agregado gerado permita responder perguntas e solucionar problemas. Nesse amplo e relevante contexto, este capítulo apresentou a ferramenta Jupyter com reprodutibilidade para a realização de projetos em Ciência de Dados. As seções foram organizadas por ordem de complexidade, iniciando nos conceitos básicos, detalhando o tratamento de dados necessário para deixá-

los prontos para realizar ciência, a qual se baseia em diferentes métodos de análise e visualização, e finalizando com conceitos avançados de Jupyter e reprodutibilidade.

De fato, notebooks Jupyter são tão flexíveis que permitem construir processamentos muito mais complexos em cima da base abordada neste capítulo. Por exemplo, o artigo [Fangohr et al. 2021] introduz uma edição especial de cinco artigos convidados a partir de apresentações na JupyterCon<sup>11</sup> 2020. Além do primeiro artigo com os criadores do Jupyter, os demais abordam pesquisas em astrofísica e geociências, bem como aplicações relevantes de simulação matemática e visualização de estruturas moleculares.

Com sua grande amplitude de utilização, vários aspectos sobre Jupyter e Ciência de Dados não foram abordados neste capítulo. Um aspecto fundamental para qualquer ciência é a atualização de dados que interfere em cálculos, análises e resultados. Atualmente, existem trabalhos em andamento para tornar notebooks mais reativos e interativos, assim como planilhas de cálculo [Perkel 2021a]. Outra frente de trabalho busca transformar notebooks em aplicativos Web que sejam leves, interativos, de código aberto e reprodutíveis [Clarke et al. 2021]. Os chamados *Appyters* constituem workflows reutilizáveis baseados na Web e incluem pipelines de aprendizado de máquina, entre outras funções destinadas às áreas de biomedicina e bioinformática. Outra questão relevante sempre presente em pesquisa orientada a dados inclui ética no compartilhamento e processamento de dados. Como exemplo, o trabalho [Peisert 2021] discute como ambientes de execução confiável (do inglês *trusted execution environments*) permitem dados sensíveis serem analisados sem ter de confiar na administração do sistema ou provedora de computadores.

Jupyter também tem muito a contribuir para a educação em seus diversos níveis. Por exemplo, o artigo [Willis et al. 2020] mostra como notebooks Jupyter podem auxiliar a desenvolver habilidades de comunicação através da escrita científica. De maneira mais ampla, o artigo [Manzoor et al. 2020] aponta que corrigir e avaliar (dar notas) para trabalhos escolares entregues em notebooks Jupyter são tarefas complexas de serem realizadas manualmente. Então, apresenta a implementação de um autocorretor para tais trabalhos, com as vantagens de feedback imediato para estudantes e diminuição considerável da carga de trabalho docente.

Os benefícios do Jupyter para educação são muito claros. Porém, como a sua utilização para fins educacionais ainda está nos anos iniciais, existem muitos pontos a serem melhorados, como discutido em [Johnson 2020]. Entre as principais críticas apontadas em tal estudo estão comportamento imprevisível, dificuldade de replicação, permissibilidade de práticas fracas de programação, e abertura para possíveis problemas de segurança. Este capítulo está em sincronia com as duas primeiras críticas ao discutir conceitos de reprodutibilidade e boas práticas na seção anterior.

**Disponibilidade de dados e código.** O código-fonte do capítulo inteiro, bem como o conjunto de dados utilizado estão disponíveis no repositório <https://github.com/opgabriel/jai2021-jupyter>.

**Agradecimentos.** Agradecemos a Juliana Freire, Leonardo Murta e Vanessa Braganholo por colaborações em trabalhos anteriores de onde obtivemos dados sobre o uso de Note-

---

<sup>11</sup><https://jupytercon.com>

books e boas práticas para reprodutibilidade e qualidade. Este trabalho foi parcialmente financiado por Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq, Fundação de Amparo à Pesquisa do Estado de Minas Gerais – FAPEMIG e Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – CAPES.

## Referências

- [Chirigati et al. 2016] Chirigati, F., Rampin, R., Shasha, D., and Freire, J. (2016). Rezip: Computational reproducibility with ease. In *SIGMOD*, pages 2085–2088.
- [Clarke et al. 2021] Clarke, D. J. et al. (2021). Appyters: Turning jupyter notebooks into data-driven web apps. *Patterns*, 2(3):100213.
- [Fangohr et al. 2021] Fangohr, H., Kluyver, T., and DiPierro, M. (2021). Jupyter in computational science. *Computing in Science Engineering*, 23(2):5–6.
- [Head et al. 2019] Head, A., Hohman, F., Barik, T., Drucker, S. M., and DeLine, R. (2019). Managing messes in computational notebooks. In *CHI*, pages 1–12.
- [Iguar and Seguí 2017] Iguar, L. and Seguí, S. (2017). *Introduction to Data Science - A Python Approach to Concepts, Techniques and Applications*. Undergraduate Topics in Computer Science. Springer.
- [Johnson 2020] Johnson, J. W. (2020). Benefits and pitfalls of jupyter notebooks in the classroom. In Khazanchi, D., Siy, H. P., Grispos, G., and Setor, T. K., editors, *SIGITE*, pages 32–37.
- [Kluyver et al. 2016] Kluyver, T. et al. (2016). Jupyter notebooks - a publishing format for reproducible computational workflows. In *ELPUB*, pages 87–90.
- [Lewine 1991] Lewine, D. (1991). *POSIX programmers guide*. "O'Reilly Media, Inc."
- [Macke et al. 2020] Macke, S., Gong, H., Lee, D. J.-L., Head, A., Xin, D., and Parameswaran, A. (2020). Fine-grained lineage for safer notebook interactions. *arXiv preprint arXiv:2012.06981*.
- [Manzoor et al. 2020] Manzoor, H., Naik, A., Shaffer, C. A., North, C., and Edwards, S. H. (2020). Auto-grading jupyter notebooks. In *SIGCSE*, pages 1139–1144.
- [Oliveira et al. 2020] Oliveira, G. P., Silva, M. O., Seufitelli, D. B., Lacerda, A., and Moro, M. M. (2020). Detecting collaboration profiles in success-based music genre networks. In *ISMIR*, pages 726–732.
- [Peisert 2021] Peisert, S. (2021). Trustworthy scientific computing. *Communications of the ACM*, 64(5):18–21.
- [Perkel 2018] Perkel, J. M. (2018). Why Jupyter is data scientists' computational notebook of choice. *Nature*, 563:145–146.
- [Perkel 2021a] Perkel, J. M. (2021a). Reactive, reproducible, collaborative: computational notebooks evolve. *Nature*, 593:156–157.

- [Perkel 2021b] Perkel, J. M. (2021b). Ten computer codes that transformed science. *Nature*, 589:344–348.
- [Pimentel 2021] Pimentel, J. F. (2021). *Provenance from Script*. PhD thesis, Universidade Federal Fluminense.
- [Pimentel et al. 2019] Pimentel, J. F., Murta, L., Braganholo, V., and Freire, J. (2019). A large-scale study about quality and reproducibility of jupyter notebooks. In *MSR*, pages 507–517.
- [Pimentel et al. 2021] Pimentel, J. F., Murta, L., Braganholo, V., and Freire, J. (2021). Understanding and improving the quality and reproducibility of jupyter notebooks. *Empirical Software Engineering*, 26(4):65.
- [Shen 2014] Shen, H. (2014). Interactive notebooks: Sharing the code. *Nature*, 515(7525):151–152.
- [Skiena 2017] Skiena, S. (2017). *The Data Science Design Manual*. Texts in Computer Science. Springer.
- [van Rossum et al. 2001] van Rossum, G., Warsaw, B., and Coghlan, N. (2001). Pep 8: style guide for python code. <https://www.python.org/dev/peps/pep-0008/>. Accessed: 2019-10-01.
- [Wang et al. 2020] Wang, J., Tzu-Yang, K., Li, L., and Zeller, A. (2020). Assessing and Restoring Reproducibility of Jupyter Notebooks. In *ASE*, pages 138–149.
- [Willis et al. 2020] Willis, A., Charlton, P., and Hirst, T. (2020). Developing students’ written communication skills with jupyter notebooks. In *SIGCSE*, pages 1089–1095.



## Capítulo

# 2

## Métodos Experimentais em Interação Humano Computador

Carlos H. Morimoto e Antonio Diaz-Tula

### *Abstract*

*In this course you will learn the basics to conduct scientific research in Human Computer Interaction (HCI), i.e., to plan, conduct, and analyze the results of an experiment with users interacting with some computational device. These experiments are fundamental for the innovation and development of interactive products. The course blends theory and practice, beginning with a brief introduction to HCI and the interaction design process to motivate the need to conduct user experiments. To conduct an experiment, you will be required to define your methodology and follow a rigorous statistical analysis to validate your results. At the end you will conduct a simple experiment to apply these concepts and evaluate a model that helps predict user performance in pointing tasks.*

### *Resumo*

*Nesse curso você vai aprender alguns fundamentos para conduzir pesquisa científica na área de Interação Humano Computador (IHC), ou seja, como planejar, conduzir e analisar os resultados de um experimento com usuários de algum sistema computacional interativo. Esses experimentos são fundamentais para a inovação e desenvolvimento de produtos interativos. O curso alia teoria e prática, começando com uma breve introdução da área para motivar a necessidade da realização de experimentos com usuários. A condução de experimentos requer a definição da metodologia e uma rigorosa análise estatística para comprovação dos resultados. Ao final, você vai conduzir um experimento para aplicar esses conceitos e avaliar um modelo que ajuda a prever o desempenho das pessoas em tarefas de apontamento.*

### **2.1. Introdução**

Os computadores de uso genérico, até a década de 70, eram máquinas grandes, caras, a que poucas pessoas tinham acesso. Eram tipicamente usadas por instituições de grande

e médio porte, tanto públicas quanto privadas, para processar seus dados. As poucas pessoas que tinham contato e acesso a essas máquinas costumavam ser altamente treinadas e com muitos anos de experiência e, assim, não havia grande preocupação com a "interação" com computadores. Apenas a partir da década de 80, com o surgimento dos microcomputadores pessoais de mesa, que o problema de interação começou a ganhar evidência.

A área de Interação Humano Computador (IHC) floresceu dessa necessidade de desenvolver melhores computadores. Não apenas máquinas mais velozes, poderosas e baratas. Mas melhores no sentido de serem mais úteis, mais fáceis de aprender e eficazes para as pessoas que fazem uso do computador em pequenos estabelecimentos comerciais e em seus lares. Essas pessoas usuárias de microcomputadores, principalmente nas décadas de 80 e 90, não tinham conhecimento nem experiência prévia com o uso de computadores. As conquistas realizadas na tentativa de vencer tal desafio, de tornar uma máquina tão complexa mais adequada às necessidades das pessoas que tentavam usá-la, é uma das razões que torna a área de IHC tão fascinante pois a sua própria evolução está diretamente relacionada à dramática alteração nas práticas de desenvolvimento de sistemas computacionais interativos.

Uma dessas práticas é o envolvimento das próprias pessoas usuárias no processo de desenvolvimento, como no método de **Design Centrado no Usuário** (*User-Centered Design*) [Sharp et al. 2019]. A Figura 2.1 ilustra esse processo iterativo. O processo tem início com um entendimento dos **requisitos** do sistema para satisfazer as necessidades das pessoas usuárias. A etapa de **desenho** busca desenvolver soluções que pode requerer um maior entendimento sobre as pessoas e algumas ideias podem ser rapidamente avaliadas. As melhores ideias devem ser prototipadas para uma avaliação mais rigorosa. Como o resultado da avaliação pode revelar problemas, dependendo da origem e severidade desses problemas, todas as etapas anteriores podem ser revistas. O processo se encerra quando a avaliação revelar que o projeto satisfaz os requisitos.

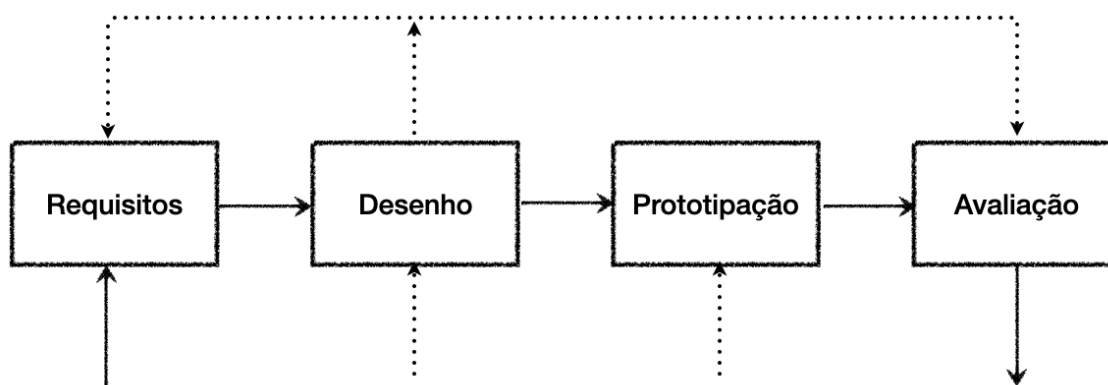


Figura 2.1. Processo de Design Centrado no Usuário.

O uso dessa prática é fundamental para evitar um foco excessivo na "engenharia" do sistema. Um projeto focado apenas na engenharia também resulta em produtos funcionais. No entanto, o processo de Design Centrado no Usuário estende o foco para a forma e/ou maneira de usar. Por fim, o envolvimento de pessoas usuárias garante que



a forma de usar é adequada e realmente satisfatória aos usuários (ao invés de adequado aos engenheiros). Historicamente, muitos projetos desenvolvidos com foco na engenharia fracassam por não satisfazerem os usuários finais e descobrirem muito tardiamente os motivos dessas insatisfações. Essas novas práticas facilitam as descobertas desses problemas com antecedência e portanto aumentam as chances de sucesso de um projeto.

Muitas variações desse processo foram introduzidas ao longo dos anos e que, tipicamente, variam o nível de participação que as pessoas usuárias tem sobre a definição do projeto e durante as etapas do projeto. Por exemplo, o método de *Design Thinking* usado pela Escola de projeto da Universidade de Stanford (<https://dschool.stanford.edu/resources>) busca conhecer seu público alvo profundamente, com um alto grau de empatia, para desenvolver soluções inovadoras que satisfaçam alguma necessidade específica. Observe que esse processo de desenho é genérico e não requer que o produto seja computacional.

Esse processo se baseia em testar as ideias rapidamente, envolvendo usuários. A prototipação permite que as pessoas possam experimentar e criticar ideias fundamentais do projeto antes de serem concretizadas na forma de produto. Esse processo iterativo de projeto, com avaliações frequentes, leva a criação de produtos realmente úteis e satisfatórios.

Boa parte das avaliações, principalmente nas fases iniciais de um projeto, podem utilizar **métodos de desconto** (*discount methods*), como o método de avaliação heurística de usabilidade proposta por Nielsen [Nielsen 1994]. Essas avaliações são ágeis, simples de executar e de baixo custo, mas por terem propósitos específicos, o escopo dos resultados dessas avaliações tende a ser limitado ao produto sendo desenvolvido.

A pesquisa (científica) em IHC possui objetivos mais fundamentais, que não estão voltados a um produto ou projeto específico e, por isso, seus resultados podem ser generalizados e utilizados para melhorar sistemas existentes e também podem contribuir no desenvolvimento de novos sistemas interativos inovadores. Um exemplo específico desse processo é o iPhone lançado em 2007 que, segundo o Prof. Selker [Selker 2008], "por meio do iPhone, a Apple conseguiu reunir com sucesso décadas de pesquisa em um só produto", como as pesquisas realizadas sobre gestos usando múltiplos toques realizadas desde a década de 80 [Buxton et al. 1985].

A pesquisa em IHC tipicamente depende de comprovação empírica que confirma (ou refuta) uma ideia. Para isso as ideias precisam ser colocadas na forma de hipóteses cuja validade deve ser testada em experimentos com usuários. Na próxima seção vamos discutir os tipos, as razões e as maneiras como esses experimentos podem ser realizados. Antes porém, queremos descrever melhor nossa motivação ao propor esse curso para ser oferecido no JAI 2021.

### **2.1.1. Motivação e objetivos**

Esse curso surgiu do interesse de alguns alunos e alunas durante aulas da disciplina MAC0446 ("Princípios de Interação Humano Computador") oferecida pelo Departamento de Ciência da Computação (DCC) do Instituto de Matemática e Estatística (IME) da Universidade de São Paulo (USP) de aprender mais sobre o desenho de experimentos com

usuários e sobre a análise dos seus resultados.

Esse curso, portanto é voltado para alunas e alunos de graduação e talvez no início de uma pós-graduação, que já conheçam um pouco da área de IHC e que tenham conhecimentos básicos de computação e estatística. No entanto, embora nossa motivação inicial tenha sido IHC, gostaríamos que esse curso no JAI possa acolher pessoas de outras áreas, fora de IHC e até fora da computação, que tenham interesse de realizar estudos com usuários e/ou aprender um pouco sobre o método científico de pesquisa.

### **2.1.2. Organização do curso**

Por ser um curso de poucas horas, nenhum assunto será coberto com a devida profundidade. Nosso principal objetivo é introduzir e discutir alguns conceitos a partir de um exemplo real baseado em um artigo publicado pelos autores do curso e fomentar a curiosidade científica dos participantes para que inventem e realizem seus próprios experimentos, apoiados pela grande variedade de ferramentas estatísticas e computacionais que existem hoje à disposição.

A Seção 2.2 desse texto resume vários capítulos do livro *Human Computer Interaction: An Empirical Research Perspective*, do Prof. Scott Mackenzie [Mackenzie 2012]. Caso você tenha interesse, recomendamos a leitura desse livro para continuar o seu aprendizado sobre métodos experimentais em IHC. Como sugestão de outros livros mais recentes, que cobrem as técnicas de análise estatística com um pouco mais de profundidade, citamos também:

- *Research Methods in Human Computer Interaction*, de Lazar, Feng e Hocheiser [Lazar et al. 2017]; e
- *Modern Statistical Methods for HCI*, por Robertson e Kaptein (editores) [Robertson and Kaptein 2018].

Além do conhecimento estatístico, a exploração dos dados requer o uso de boas ferramentas computacionais. Nós escolhemos usar ferramentas baseadas na linguagem Python devido à generalidade da linguagem e sintaxe simples, que facilita o aprendizado e entendimento dos trechos de código utilizados nesse curso.

Ao final do curso, vamos sugerir um experimento que você pode fazer em sua própria casa para aplicar esses conceitos e ferramentas. Trata-se de um típico experimento para avaliar o desempenho humano em uma tarefa simples, para levantar uma curva de desempenho conhecida como lei de Fitts [Fitts 1954].

## **2.2. Pesquisa em IHC**

Quando dizemos que uma pesquisa é empírica significa que seus resultados provêm de um experimento envolvendo pessoas usuárias realizando uma determinada tarefa. Mas nem todas as pesquisas envolvem experimento. Muito se pode descobrir sobre as pessoas usuárias por meio de uma pesquisa de opinião por exemplo, onde as pessoas são consultadas diretamente (por meio de uma entrevista ou respondendo a um questionário) sobre como utilizam um determinado sistema ou qual sistema elas utilizam com mais

frequência. Além disso, algumas pesquisas desse tipo podem ser feitas por meio de consultas "indiretas" baseadas no número de cliques ou visitas de uma página na Internet. Esse método de pesquisa é conhecido por **método observacional** e é muito comum em ciências sociais. Seus resultados tendem a ser mais **qualitativos** que **quantitativos** e, por isso, são mais utilizados para descobrir a razão (o "por quê") ou a maneira que (o "como") as pessoas realizam a interação.

Nessa seção vamos discutir como realizar pesquisas segundo o **método experimental**, também conhecido como **método científico**, onde o conhecimento é adquirido por meio de **experimentos controlados**. No caso de IHC, os experimentos tipicamente envolvem pessoas e por isso é costumeiramente chamado de **estudo com usuários** (*user study*). O método científico procura responder uma questão de pesquisa cuja resposta depende da comprovação de uma hipótese. Essa hipótese é testada em um experimento e o resultado é analisado estatisticamente para determinar a confiança que podemos ter no resultado.

Uma importante característica do método científico é que o experimento deve ser reproduzível, ou seja, ao realizar o mesmo experimento várias vezes, devemos chegar a mesma conclusão. Por isso, as condições do experimento tendem a ser controladas, o que pode reduzir de certa forma a relevância dos resultados em IHC pois as pessoas realizam as tarefas em condições artificiais (controladas).

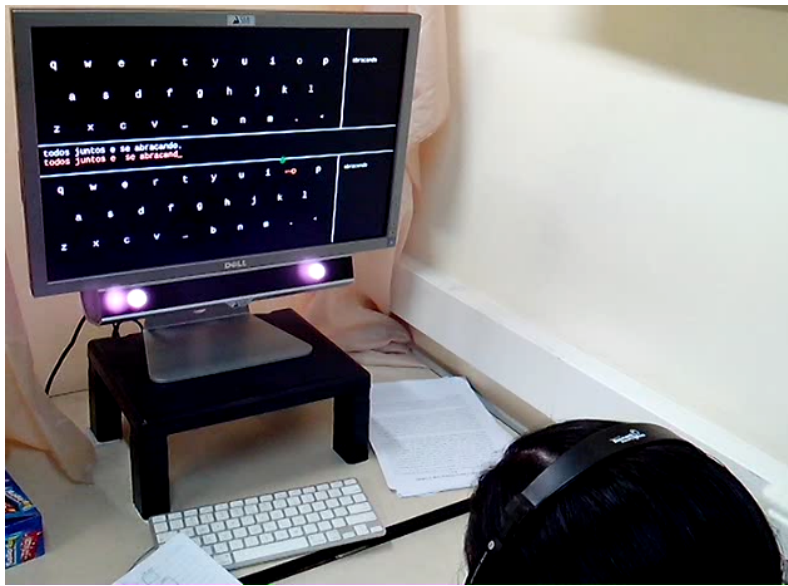
Um experimento controlado requer ao menos duas variáveis: uma variável a ser **manipulada** e outra a ser medida para indicar a **resposta**. Em IHC, a variável manipulada em geral corresponde a um parâmetro modificável na interface, como o tamanho de uma tecla virtual em um editor de texto. Já a resposta pode ser uma medida qualitativa como a opinião dos participantes sobre o conforto oferecido pelo teclado ou uma medida quantitativa como a velocidade ou taxa de erros de digitação.

### **2.2.1. AugKey: exemplo de um estudo com usuários**

Para contextualizar nossos exemplos e facilitar o entendimento dos conceitos que vamos apresentar a respeito de estudos com usuários vamos utilizar um experimento real publicado pelos autores utilizando um teclado virtual com predição (denominado AugKey [Diaz-Tula and Morimoto 2016]). O objetivo do AugKey é melhorar a experiência de pessoas com deficiências motoras na tarefa de entrada de texto utilizando apenas o olhar.

Para realizar a entrada de texto ("digitação") usando apenas os movimentos dos olhos utilizamos um dispositivo rastreador de olhar como ilustrado na Figura 2.2. O rastreador de olhar [Morimoto and Mimica 2005] é tipicamente composto por uma câmera (colocada sob o monitor) que rastreia os movimentos dos olhos e os mapeia para a tela do próprio computador, como mostrado na figura. Esse dispositivo permite, por exemplo, controlar o cursor na tela pelo olhar, ao invés de usar o mouse ou outro dispositivo de apontamento manual. O custo desses dispositivos tem caído muito e atualmente há modelos comerciais de rastreadores de olhar que podem ser acoplados diretamente à tela de monitores de mesa e notebooks, usados por exemplo em alguns jogos eletrônicos.

Um uso tradicional de rastreadores de olhar é na área de acessibilidade. Algu-

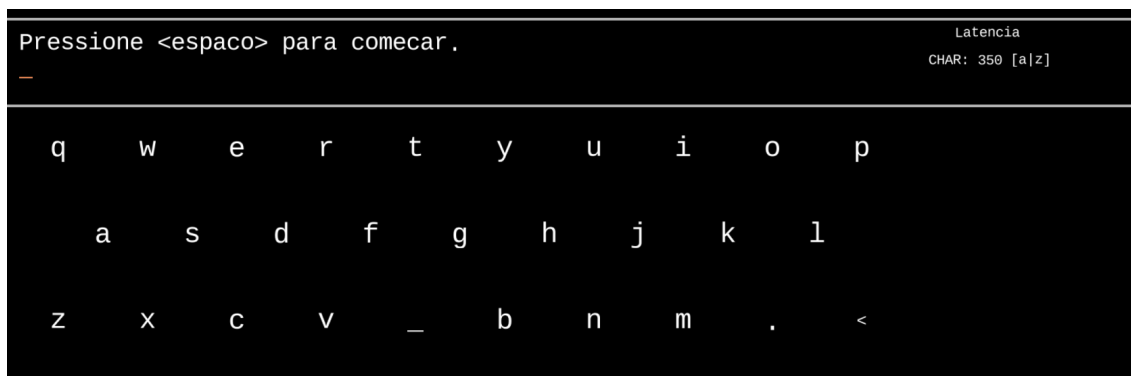


**Figura 2.2.** Entrada de texto pelo olhar usando um rastreador remoto.

mas pessoas com deficiências motoras severas, como por exemplo pessoas afetadas por Esclerose Lateral Amiotrófica ou Síndrome do Encarceramento, não possuem ou perdem controle muscular dos membros do corpo e, portanto, não conseguem utilizar dispositivos convencionais como teclado e mouse. Muitas dessas pessoas se beneficiam de aplicativos controlados pelo olhar para interagir com o computador e, em casos mais severos quando a pessoa não tem controle da fala, essas interfaces se tornam também um meio importante para elas se comunicarem. A entrada de texto pelo olhar, além de facilitar a comunicação dessas pessoas, permite a sua integração digital com a sociedade.

Tipicamente, a digitação pelo olhar é realizada utilizando um teclado virtual exibido no monitor, como ilustrado na Figura 2.3. O primeiro desafio é como "clique" em uma tecla. Algumas pessoas conseguem usar um botão acionado mecanicamente usando a boca ou o pé, ou ainda piscando os olhos ou balançando a cabeça. Uma solução frequentemente utilizada e baseada apenas no olhar é usar tempo de latência (*dwell-time*) para acionar a tecla sendo focada. Assim, quando a usuária deseja digitar uma tecla, ela deve olhar para a tecla e permanecer olhando por um determinado tempo, sem desviar o olhar da tecla, até ela ser acionada. Uma ampulheta (ícone que indica a passagem do tempo) é exibida para que a pessoa saiba quanto tempo ela precisa esperar e/ou até quando ela pode desistir de olhar para evitar uma seleção indesejada. É comum utilizar tempos de latência entre 500 e 1000 ms.

Digitar por meio do olhar é um processo lento e pode causar fadiga visual após um período mais prolongado. Para reduzir a fadiga visual e aumentar a velocidade de digitação, podemos utilizar a técnica de predição de palavras encontrada em vários editores de texto, em particular, aqueles utilizados em dispositivos móveis. A predição de palavras permite que o sistema ofereça algumas alternativas logo após a digitação de alguns caracteres. A palavra candidata mais provável pode aparecer como opção de *auto-completar*, e mais palavras candidatas podem aparecer em outra região da interface. Para utilizar uma dessas outras palavras, basta a pessoa clicar na opção desejada na lista de



**Figura 2.3. Teclado virtual para entrada de texto pelo olhar. Os movimentos oculares são rastreados usando uma câmera de vídeos apontada para os olhos.**

palavras candidatas [Majaranta 2009, Wobbrock et al. 2008, Urbina and Huckauf 2010].

Embora o uso de predição de palavras tenha o potencial de aumentar a velocidade de digitação pelo fato de economizar a digitação dos caracteres que são completados, a necessidade de alternar o foco entre o teclado sendo acionado pelo olhar e a lista de palavras candidatas exibidas em outra região da tela limita bastante o ganho obtido na prática. Além disso, a pessoa usuária precisa olhar repetidamente para a área de texto para detectar erros na digitação e confirmar o que realmente foi digitado.

Com o objetivo de aliviar o problema de desviar o foco repetidamente para a lista de palavras, propomos em [Diaz-Tula and Morimoto 2016] um novo modelo de interação chamado de *AugKey* (*Augmented Keys*). Nossa ideia foi aumentar a quantidade de informação exibida na tecla sob o foco do olhar para que essas informações possam ser consumidas durante a latência, sem que o olhar precise se mover. Assim o olhar não precisaria mais ser deslocado da tecla focada para procurar informação sobre o texto que acabou de ser digitado, pois essa informação já se encontra na área da tecla focada. A Figura 2.4 mostra um teclado virtual com *AugKey*. Observe que, com *AugKey*, dentro da região da tecla e ao redor do caractere recebendo o foco do olhar (a tecla "t" na figura) são mostradas informações adicionais: os últimos caracteres digitados e as palavras candidatas que serão exibidas na lista de palavras candidatas após a seleção do caractere focado (fim do tempo de latência).

As perguntas de pesquisa que devemos investigar devem comprovar se as alterações propostas pelo *AugKey* tem algum efeito sobre os teclados tradicionais e, caso tenha algum efeito, como podemos medir se esse efeito é positivo ou negativo. Por exemplo, desejamos saber qual método permite atingir uma maior velocidade de digitação, ou qual deles é mais confortável, ou mais fácil de aprender. Ao projetar um experimento devemos definir as variáveis que devem ser manipuladas para que outras possam ser medidas, como veremos a seguir.

### **2.2.2. Variáveis e Dados**

A sofisticação das conclusões do experimento estão diretamente relacionadas às medidas e, por isso, para se obter conclusões mais elaboradas é necessário utilizar medidas quantitativas sempre que possível. No entanto, há dados de natureza qualitativa que são

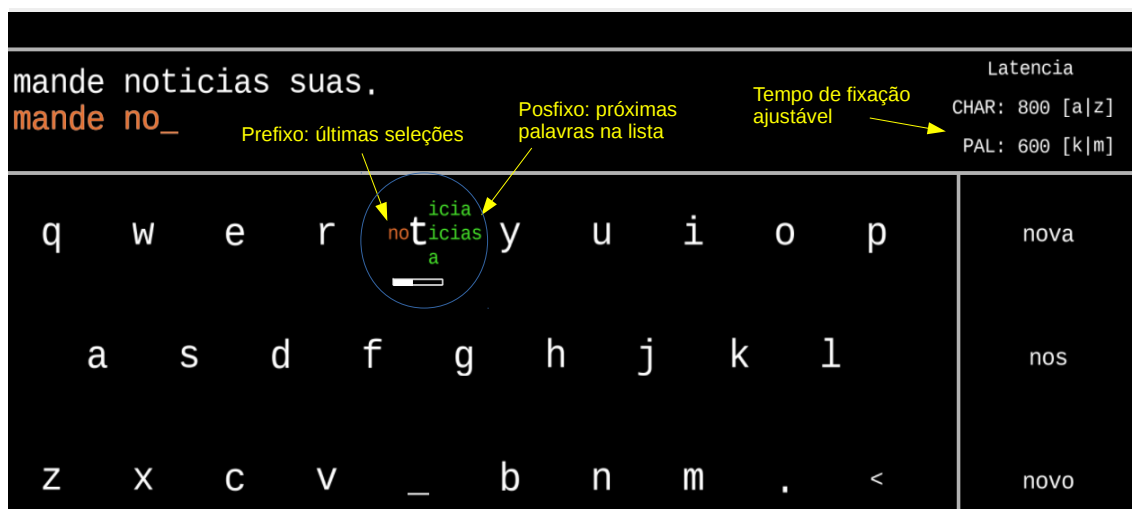


Figura 2.4. Teclado virtual controlado por tempo de latência com uma lista de predição de palavras baseado em AugKey. A lista de palavras é mostrada do lado direito da interface. A tecla recebendo o foco do olhar exhibe os últimos caracteres digitados, as próximas palavras a serem apresentadas na lista de predições, e a apulheta sob o caractere.

essenciais também para avançar o conhecimento. Cada tipo de dado exige um tratamento diferente para ser analisado.

Além do tipo, devemos considerar também a escala de medição utilizada. É comum utilizar as seguintes quatro escalas: nominal, ordinal, intervalar e proporcional. Escalas nominais e ordinais são tipicamente usadas para medir dados **qualitativos** enquanto escalas intervalares e proporcionais são usadas para medir dados **quantitativos**. Vamos a seguir descrever propriedades dessas escalas para entender seus usos e limitações.

### 2.2.2.1. Escala Nominal

Um dado em escala nominal (dado nominal) é basicamente atribuído a uma categoria como, por exemplo, sexo (masculino ou feminino), nacionalidade (brasileira, cubana, japonesa etc.), ou tipo de animal preferido de estimação (cachorro ou gato). Essa escala é usada para agrupar os dados em classes ou categorias distintas.

Dados em escala nominal limitam manipulações matemáticas e estatísticas. Por exemplo, será que podemos calcular a média dentre os animais de estimação para obter um bicho metade cachorro e metade gato?

Dados nominais são tipicamente usados para exibir a frequência ou número de ocorrências em cada categoria, como por exemplo, dizer coisas como: 72% dos moradores do bairro Periferia tem animais de estimação, desses 67% são cães, 21% gatos e 12% possuem outros animais como roedores, tartarugas e pássaros.

No experimento do AugKey a variável "teclado" é de tipo nominal e possui dois valores possíveis como "Sem predição" de palavras e "AugKey".

### 2.2.2.2. Escala Ordinal

Um dado em escala ordinal (dado ordinal) permite que ele seja ordenado segundo alguma propriedade. Por exemplo, podemos entrevistar algumas pessoas e pedir para que elas ordenem três cores, ou perfumes, ou marcas de refrigerante, segundo sua preferência. Assim, certa pessoa pode ter como fruta preferida "melancia", seguido de "manga" e "mamão". Ou ainda, podemos perguntar a uma pessoa que ordene propriedades que ela considera ao comprar algum produto como preço, beleza, durabilidade e conforto. Como resultado de uma pesquisa usando essas propriedades considerando, por exemplo, um sofá, podemos descobrir que a beleza é o ponto mais importante para nossos "clientes" (pessoas que participaram da pesquisa e potenciais usuárias de sofás), seguido de preço, durabilidade e, por último, conforto.

A principal limitação do uso de dados ordinais é que geralmente a diferença entre pontos sucessivos na escala não é a mesma, ou seja, a diferença entre o preço e a durabilidade pode ser grande para os clientes, enquanto a diferença entre durabilidade e conforto pode ser pequena.

Observe que ainda não faz sentido calcular a média desses dados mas eles fornecem uma informação mais rica que dados nominais pois permitem comparações do tipo maior e menor, como *pessoas consideram preço mais relevante que conforto*, nesse exemplo.

Voltando ao AugKey, há várias propriedades qualitativas que poderiam ser ordenadas, como por exemplo o conforto oferecido por cada interface ao digitar e a preferência de cada pessoa por um dos métodos. Além disso, mesmo propriedades quantitativas, como velocidade de digitação e número de erros, podem receber ordenações segundo a percepção das usuárias. É possível, por exemplo, que a velocidade de digitação de um método seja percebida como menor que outro método embora, ao ser medida quantitativamente, se revele maior.

### 2.2.2.3. Escala Intervalar

Um dado em escala intervalar (dado intervalar) apresenta distâncias iguais entre valores adjacentes. No entanto, a escala ainda limita certas comparações pela falta de uma referência absoluta (um zero). Um exemplo típico de dado intervalar (e que pode confundir você) é a temperatura medida em graus Celsius.

Dados intervalares, como temperatura, permitem o cálculo de medidas estatísticas como a média e a variância da temperatura em um determinado dia do ano. No entanto, não é válido considerar o resultado de proporções desses dados. Por exemplo, não deveríamos dizer que  $30^{\circ}\text{C}$  é seis vezes mais quente que  $5^{\circ}\text{C}$  (ou ainda menos três vezes mais quente que  $-10^{\circ}\text{C}$ ?).

Em IHC é comum também considerar os dados de questionários com respostas na forma de escalas de Likert como dados intervalares. Embora exista evidência de que as pessoas percebam os itens nos extremos da escala como mais afastados que os itens no centro [Kaptein et al. 2010], para que a média dos valores faça sentido é necessário con-

siderar que os intervalos possuam a mesma distância. Uma solução é instruir as pessoas participantes a considerar que as categorias possuem mesma distância.

Observe que dados intervalares são mais ricos que dados ordinais e que é possível transformar esses dados em ordinais e até nominais, por exemplo, classificando temperaturas nas categorias "quente" e "frio", ou ainda "quente", "morno" e "frio" etc. No caso de teclados, a velocidade de digitação pode ser percebida como teclado "rápido" ou "lento".

#### **2.2.2.4. Escala Proporcional**

Um dado em escala proporcional (dado proporcional) permite o cálculo de proporções, ou razões entre valores (por isso é também chamada de escala de razão (*ratio scale*)) pois possuem um zero absoluto. Tais medidas permitem o cálculo de uma série de medidas que contribuem para conclusões mais elaboradas sobre os resultados do experimento. Esses dados podem ser somados e subtraídos, multiplicados e divididos, para calcular médias, desvios e variâncias estatísticas.

Em IHC, a medida mais comum utilizada em escala proporcional é o tempo, na forma de intervalo de tempo utilizado para completar uma tarefa. Até mesmo a idade e anos de experiência (tempo) são dados proporcionais bastante utilizados. Podemos assim dizer que certa pessoa possui duas vezes mais experiência que outra.

Em geral, qualquer outra medida física além de tempo, como a força que uma pessoa deve exercer sobre um controle ou é percebida de um dispositivo háptico, ou a distância percorrida pelo cursor no monitor, são também medidas proporcionais.

No exemplo que estamos estudando sobre comparação de teclados virtuais, a velocidade de digitação medida em número de caracteres (ou palavras) por minuto representa uma variável em escala proporcional.

#### **2.2.3. Metodologia**

Enquanto o método científico define o processo, a metodologia descrita nessa seção se refere ao "estudo" desse processo que define como o experimento deve ser realizado para responder à pergunta da pesquisa. No caso de IHC, isso envolve a escolha dos participantes, os materiais (hardware e software), as tarefas (e como elas devem ser realizadas), as instruções para receber e treinar os participantes antes do experimento e coletar informações individuais durante e após seu término, as variáveis a serem manipuladas e medidas, a forma de coleta e análise dos dados etc.

Definir uma metodologia adequada é fundamental para que possamos confiar nos resultados e replicá-los. Uma metodologia deficiente não permite conclusões (pois impossibilita sua replicação) ou aumenta a incerteza sobre os resultados (os coloca em dúvida).

Por envolver pessoas, é muito importante também submeter o experimento a um comitê de ética e só iniciar o experimento após a sua aprovação. No Brasil, é necessário submeter seu projeto pela Plataforma Brasil [<http://plataformabrasil.saude.gov.br>], que encaminha o projeto a um comitê de ética próximo ao local onde o projeto será conduzido.



A Plataforma Brasil é uma base nacional e unificada de registros de pesquisas envolvendo seres humanos mantida pelo Ministério da Saúde. Ela permite que as pesquisas sejam acompanhadas em seus diferentes estágios - desde sua submissão até a aprovação final pelo comitê de ética que acompanha o projeto. O sistema permite a apresentação de documentos também em meio digital, propiciando à sociedade o acesso aos dados públicos de todas as pesquisas aprovadas.

O comprometimento ético requer também que os participantes sejam esclarecidos sobre:

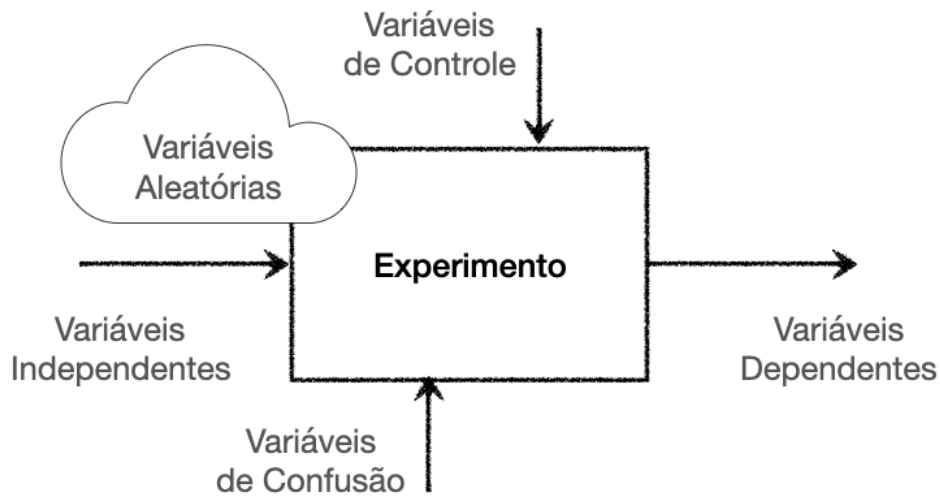
- a natureza e os propósitos da pesquisa;
- a metodologia utilizada (procedimentos, questionários etc);
- os riscos e benefícios da participação;
- o direito de não participar, interromper a participação a qualquer momento e de não precisar responder as perguntas que não desejar, sem incorrer em nenhum prejuízo ou penalidade;
- o direito ao anonimato e confidencialidade dos dados coletados.

Assim, ao participar de um experimento, a pessoa recebe instruções iniciais descrevendo o experimento e o protocolo experimental. A seguir deve preencher um formulário de consentimento esclarecido confirmando que está ciente dos objetivos do experimento e está de acordo com a forma que o experimento será conduzido e como os dados coletados serão usados. Esse formulário é assinado pelo experimentador e pelo participante, que também recebe uma cópia para que possa entrar em contato com os experimentadores após o experimento.

### 2.2.3.1. Desenho experimental

O desenho do experimento define as ações necessárias para responder a pergunta de pesquisa (testar a hipótese). Um passo importante é definir as variáveis do experimento. As variáveis a serem manipuladas são chamadas de variáveis **independentes** enquanto as variáveis a serem medidas são chamadas de **dependentes**. Em IHC, qualquer aspecto mensurável do comportamento humano pode ser uma variável dependente. Há vários tipos de variáveis que podem afetar os resultados, como ilustrado na Figura 2.5 e que são descritas nessa seção.

Voltando ao exemplo de digitação pelo olhar, a velocidade de digitação (medida em número de palavras por minuto [Mackenzie et al. 1999]) constitui a variável dependente, pois depende do que a pessoa participante faz e também do tipo de teclado virtual utilizado para digitação: sem predição de palavras ou AugKey. Da mesma forma, até porque as pessoas participantes não tem como influenciar no tipo de teclado, a variável tipo de teclado (sem predição ou AugKey) corresponde à variável independente desse experimento.



**Figura 2.5. Tipos de variáveis a serem consideradas no desenho de um experimento.**

As variáveis independentes são também chamadas de **fatores** pois é por meio da manipulação desses fatores (mudança de teclado) que medimos os resultados (velocidade de digitação). Experimentos concebidos dessa forma são chamados de **experimentos fatorados** (*factorial experiments*).

Assim como no caso do teclado, a variável independente é tipicamente nominal, com duas (ou mais) categorias. Em nosso exemplo temos dois teclados, o sem previsão e o AugKey, mas poderíamos ter outros tipos de teclado também. O número de categorias define os **níveis** (*levels*) da variável independente e são frequentemente chamadas de **condições de teste** (*test conditions*) pois o experimento precisa ser realizado sob cada condição.

Entretanto, as variáveis independentes não precisam estar relacionadas a alguma característica da interface, mas podem estar relacionadas também a características humanas (como sexo, altura, peso, nível de educação, condição econômica, etc) e do ambiente (como nível de barulho, iluminação etc.).

Embora pareça razoável conceber e conduzir um experimento com apenas uma variável independente, é comum conduzir experimentos com múltiplas variáveis imaginando economizar tempo e outros recursos gastos na condução do experimento. No entanto, quanto mais variáveis, mais complexo se torna o experimento e efeitos inesperados entre as variáveis podem comprometer todo o experimento.

Um desenho baseado em apenas um fator possui um **efeito principal** (*main effect*) e nenhum **efeito de interação** (*interaction effect*) entre variáveis. Um desenho com duas variáveis independentes possui dois efeitos principais e um efeito de interação, ou seja, 3 efeitos que devem ser estudados. O efeito de interação é também chamado de **interação por duas vias** (*two-way interaction*) pois é efeito da interação de duas variáveis que não seria sentido na ausência de uma delas. Isso é comum por exemplo em medicamentos que

isoladamente tem efeitos benéficos mas que não devem ser tomados em conjunto pois a interação entre os medicamentos pode ser danosa.

No exemplo de digitação pelo olhar, uma outra variável independente que podemos considerar é o número de sessões que uma pessoa participante deve realizar utilizando cada teclado. Em cada sessão, poderíamos pedir que a pessoa digite um certo número de frases para que possamos avaliar a velocidade de digitação. É comum que após algumas sessões as pessoas aprendam a usar as interfaces e por isso melhorem seu desempenho. Neste caso, podemos avaliar dois efeitos principais (tipo de teclado e número de sessão) e um efeito de interação entre ambas as variáveis independentes.

Imagine agora um experimento com 3 fatores (ou variáveis independentes). No total precisamos considerar 7 efeitos (3 principais, 3 efeitos de duas vias e 1 efeito de 3 vias)! Com quatro fatores, o número de efeitos cresce para 15 (4 principais, 6 de duas vias, 4 de três vias e 1 de quatro vias) e para 5 o número de efeitos a considerar é 31 (5 principais e 26 possíveis efeitos de interação). O elevado número de fatores dificulta a interpretação dos resultados e por isso experimentos com 3 ou mais fatores devem ser evitados.

Como há vários fatores que podem influenciar os resultados mas nem sempre queremos investigar os seus efeitos, esse fatores podem ser controlados e deixados de lado durante o experimento. Esses fatores são chamados de **variáveis de controle**, que podem ser, por exemplo, iluminação do ambiente, configuração do monitor (tamanho, brilho, cor, distância etc.), altura da cadeira, entre outros. A descrição das variáveis de controle e dos valores utilizados permite a reprodução dos experimentos nas mesmas condições.

Entretanto, nem sempre é possível controlar todos os fatores e alguns podem variar aleatoriamente e, por isso, são chamados de **variáveis aleatórias** (*random variables*). Essas variáveis em geral estão associadas a características dos participantes, que inclui sua biometria (altura, peso, força etc.), condição social, nível de estudo etc.

Um perigo que pode ocorrer em um experimento é a existência de **variáveis de confusão** (*confounding variables*), ou fatores de confusão, que correspondem a condições que são alteradas junto com as variáveis independentes. Voltando ao exemplo de avaliação de teclados virtuais, imagine que o tamanho dos teclados usados no experimento são diferentes: a lista de palavras ocupa a tela toda, sendo que as suas teclas são grandes, enquanto o AugKey usa 50% do espaço na tela, portanto suas teclas são menores. Assim, ao variar o tamanho das teclas, a precisão no apontamento por meio do olhar pode ser maior na lista de palavras, provocando menos erros de digitação comparado com o AugKey, que é mais suscetível a erros na estimação do olhar. Esses fatores de confusão nem sempre são tão evidentes quanto nesse exemplo, portanto os pesquisadores precisam ficar atentos a esses fatores para eliminá-los ou, quando não for possível, considerá-los de alguma forma para corrigir os seus efeitos.

### 2.2.3.2. Escolha da tarefa e participantes

Vamos voltar ao exemplo de desenho de um experimento para a avaliar se o AugKey tem algum efeito sobre a velocidade de digitação em teclados virtuais comparado com um

teclado virtual sem predição de palavras. Digamos que foi decidido realizar o experimento com uma variável independente apenas com dois níveis: sem predição e AugKey.

Que tarefa as pessoas participantes do experimento devem realizar para medir a velocidade de digitação? A escolha da tarefa deve satisfazer dois objetivos:

1. **representar** uma atividade real, comum, natural para cada pessoa realizar usando o teclado; e
2. **discriminar** as condições sendo avaliadas.

Em geral, como em nosso exemplo, a tarefa a ser realizada é evidente, ou seja, como se trata de um teclado virtual, as pessoas participantes devem digitar textos usando cada teclado. Idealmente, os textos deveriam ser os mesmos, mas digitar um texto longo pode causar fadiga e influenciar também nos resultados. Copiar um texto também é inconveniente pois cada participante pode precisar olhar o texto original várias vezes e se perder durante a digitação.

Uma forma de resolver esses problemas é criar uma base de dados com várias frases curtas e neutras, que as pessoas possam memorizar facilmente como "café com leite, pão e manteiga". Quando se sentirem prontas, as pessoas iniciam a digitação da frase e, ao terminar, uma nova frase pode ser oferecida. A medida de tempo pode ser automaticamente iniciada com a primeira letra digitada e terminada com o ponto final da frase. Além do tempo, o estímulo (frase) e a resposta (texto digitado) podem ser gravados para posterior processamento e análise.

Dessa forma, espera-se que o único efeito que influencie na velocidade de digitação seja a habilidade dos participantes de usar cada teclado. Mas alguns participantes já podem saber digitar antes do experimento usando um dos teclados mas não o outro. Como eliminar esse efeito?

Podemos selecionar participantes sem nenhuma experiência prévia com nenhum dos teclados, para equilibrar o nível inicial de experiência. Há vários fatores aleatórios no entanto que podem afetar os resultados. Um número suficiente de participantes deve ser utilizado para que os resultados sejam significativos estatisticamente. É difícil determinar quantos mas, se houver trabalhos semelhantes publicados usando dois teclados distintos, que indiquem que resultados significativos foram encontrados com, digamos, 10 participantes, essa pode ser uma boa escolha.

Como vimos, um outro fator importante que deve afetar o resultado é o aprendizado, que faz o desempenho melhorar quanto maior a prática. Se considerarmos que nenhum participante tem experiência, a velocidade de digitação depois de algumas horas de prática deve melhorar, mas o número de horas necessárias para cada participante atingir seu limite de desempenho pode variar bastante.

**Estudos longitudinais** são experimentos que consideram a quantidade de treinamento como uma variável independente. No caso de pessoas aprendendo a digitar, tipicamente as pessoas ganham bastante velocidade nas primeiras horas até atingir um patamar, onde a velocidade deixa de aumentar. Nesse instante podemos dizer que não há mais efeito do treinamento e podemos avaliar então o efeito do teclado.

Outra decisão importante para o estudo é se devemos separar os participantes em dois grupos distintos, onde cada grupo deve aprender a usar um teclado apenas, ou se todas as pessoas participantes devem aprender a usar os dois teclados.

Quando cada pessoa é testada com cada nível (todas usam os dois teclados), dizemos que a condição do teste é **intra-sujeitos** (*within-subjects*). Essa condição é também chamada de **medidas repetidas** (*repeated measures*) já que cada condição de teste é repetida por cada participante.

Quando cada pessoa é testada com apenas um nível (um dos teclados), dizemos que a condição do teste é **entre-sujeitos** (*between-subjects*). Nesse caso, para o exemplo do teclado, formam-se dois grupos distintos de participantes.

Claramente há vantagens e desvantagens em cada condição. Considerando um mesmo número total de participantes, usando intra-sujeitos teremos mais dados para cada método. No entanto, atingir o patamar nos dois métodos deve requerer um tempo bem maior de treinamento de cada participante. Ao considerar grupos distintos (entre-sujeitos), corremos um risco de um grupo desempenhar melhor devido a fatores aleatórios. No entanto, elimina-se o efeito de interação entre o aprendizado dos dois teclados. Por exemplo, aprender a usar um segundo teclado, após aprender a usar um primeiro, pode ser bem mais rápido e, ao final, obter um desempenho ainda melhor. No caso de estudos intra-sujeitos, uma forma de reduzir esse efeito de interação é alternar as sessões entre o uso de um teclado e outro, de forma que o efeito de aprendizado obtido nas sessões anteriores seja distribuído.

Por exemplo, em um estudo longitudinal que mede a velocidade de digitação diariamente, uma pessoa poderia fazer duas sessões de testes por dia, a primeira usando sem predição e a segunda usando AugKey. No dia seguinte, a ordem é invertida. Na primeira sessão, metade dos participantes devem começar com um teclado e a outra metade com o outro. Um outro desenho possível seria fazer metade do grupo começar a treinar com um método até atingir o patamar e só aprender a utilizar o segundo teclado após aprender o primeiro. Como cada metade começa aprendendo um teclado diferente, pode ser possível também inferir se há algum efeito na velocidade de aprendizado de um segundo teclado.

Para experimentos com duas variáveis independentes, é possível também utilizar um desenho misto, onde uma variável é testada intra-sujeitos e a outra entre-sujeitos. No caso dos teclados, podemos considerar se há diferença na curva de aprendizado entre homens e mulheres. Nesse caso, considerando o fator treino com por exemplo 10 níveis (10 sessões de treino com um teclado) e o fator sexo com dois níveis (feminino e masculino), o fator treino pode ser testado intra-sujeitos e o fator sexo de forma entre-sujeitos.

#### **2.2.4. Coleta dos dados**

Após definido todo o desenho experimental e preparado os materiais (hardware e software), deve-se conduzir um experimento piloto com pelo menos 2 ou 3 voluntários para testar se todo o processo está satisfatório e nenhum detalhe importante foi esquecido. Esses voluntários podem fazer parte do grupo de pesquisadores que, com uma atitude neutra, devem avaliar se todas as etapas do experimento, desde a recepção dos participantes, sua instrução, esclarecimento e assinatura do termo de consentimento esclarecido,

treinamento da tarefa, execução do experimento, até a despedida, foram implementadas satisfatoriamente.

O piloto é importante para revelar pequenos problemas ou inconveniências que devem ser resolvidas antes de iniciar a coleta verdadeira, com o protocolo experimental revisado e corrigido.

Durante o experimento, é necessário também que os experimentadores se conduzam de forma neutra para não influenciar, positiva ou negativamente, os participantes. Por exemplo, frases como "agora você vai usar esse teclado que é bem melhor que o primeiro", devem ser evitadas.

### 2.2.5. Avaliação dos resultados

Lembre-se que o objetivo do experimento é responder uma pergunta de pesquisa, comprovando ou refutando uma hipótese. Para o exemplo dos teclados, o experimento deve coletar dados de várias pessoas e a seguir precisamos comprovar se há diferença entre os desempenhos medidos de cada teclado. Chamamos de **hipótese nula** o caso em que não há diferença, ou seja, todos os teclados apresentam o mesmo desempenho. Os testes de hipótese são métodos estatísticos que podem comprovar ou refutar a hipótese nula.

O resultado de um teste é denominado estatisticamente significativo se for considerado improvável de ter ocorrido por acaso, assumindo a hipótese nula verdadeira. Para isso, o teste calcula uma probabilidade (valor  $p$ ) e caso  $p$  seja menor que um limite pré-especificado (nível de significância, tipicamente 5%), a hipótese nula pode ser rejeitada. Caso contrário, dizemos que não há diferença significativa. No caso dos teclados, isso serviria como evidência que o AugKey não tem efeito (nem melhora nem piora) sobre a velocidade de digitação usando um teclado sem predição de palavras.

A escolha do método a ser empregado depende do tipo de dado coletado. Testes **não paramétricos** são tipicamente usados para o tratamento de dados nas escalas ordinal e nominal. Dados intervalares e proporcionais são tipicamente avaliados por meio de testes **paramétricos**. Como esses dados podem ser reduzidos a dados nominais ou ordinais, eles também podem ser tratados por testes não paramétricos.

Testes paramétricos são assim chamados pois dependem de distribuições de probabilidade, como uma distribuição normal ou distribuição  $t$ , enquanto os testes não paramétricos não assumem qualquer distribuição em particular.

Um procedimento padrão para analisar dados nominais (em escala nominal ou categórica) é o teste chi-quadrado ( $\chi^2$ ). Nesse teste, os dados são organizados na forma de uma tabela de contingência, onde cada linha e coluna contém as frequências das observações de cada categoria. O teste  $\chi^2$  compara as frequências observadas com os valores esperados, assumindo que todas as categorias devem possuir comportamentos similares (hipótese nula). A aplicação do teste confirma ou não essa hipótese.

Para dados ordinais os testes mais comuns são o teste de Mann-Whitney, o teste de postos sinalizados de Wilcoxon, o teste de Kruskal-Wallis e o teste de Friedman [Robertson and Kaptein 2018]. Esses testes seguem o mesmo procedimento geral de calcular uma grandeza estatística relativa ao teste usando os dados para rejeitar ou não a hipótese nula.

Apesar da relevância dos testes não paramétricos, os testes paramétricos são os mais utilizados em pesquisas na área de IHC por serem capazes de tratar os dados quantitativos resultantes de medidas de desempenho humano. A próxima seção introduz a análise de variância ou ANOVA, por se tratar do teste paramétrico mais comum e amplamente utilizado.

### 2.3. Análise de variância

A ANOVA, ou teste-F, é uma forma de teste de hipótese estatística amplamente utilizada na análise de dados em experimentos fatorados. ANOVA pode ser usada tanto em estudos entre-sujeitos quanto em estudos intra-sujeitos. Também é aplicável quando existe mais de uma variável independente.

Vamos supor que nosso estudo tem apenas uma variável independente, como no exemplo dos teclados virtuais, onde a variável independente é o tipo de teclado e possui dois níveis: sem predição e AugKey. Como em todo teste estatístico, na ANOVA a hipótese nula é que não existe diferença significativa entre as médias de cada condição experimental: as médias de velocidade de digitação com o teclado sem predição e com o teclado com AugKey são similares. Esta é a hipótese que o teste ANOVA irá rejeitar ou não, dado o nível de significância preestabelecido.

Quando a hipótese nula é rejeitada (pois obtivemos um valor de  $p$  menor que o nível de significância, usualmente  $p < 0.05$ ), podemos concluir que a variável independente tem um efeito significativo na variável dependente. Ou seja, existem pelo menos dois níveis da variável independente cujas médias são significativamente diferentes. Como temos apenas 2 níveis (tipos de teclado), então podemos concluir que a diferença entre as duas é significativa. Este é o caso mais simples de aplicação da ANOVA. Em caso de existir diferença significativa, uma simples conferência da média, nos revela qual método apresentou melhor desempenho nos experimentos.

O que acontece quando existem mais de 2 níveis, por exemplo, se estivermos comparando 3 ou mais tipos de teclado? A interpretação é a mesma que para 2 níveis: a variável independente tem um efeito significativo na variável dependente, porém ANOVA não nos diz exatamente quais níveis são diferentes entre si.

Para entender melhor, vamos supor que nossa variável independente tem três níveis: A, B e C. Um resultado significativo de ANOVA significa que há uma diferença significativa entre as médias de alguns dos grupos, mas que pode ser entre A e B, B e C, A e C ou mais de uma combinação ao mesmo tempo. Para saber onde, precisamos fazer um teste conhecido como *post-hoc*.

Os testes *post-hoc* complementam um resultado significativo de ANOVA, nos ajudando a descobrir entre quais níveis a diferença é significativa. O mais comum é comparar todos os níveis da variável independente dois a dois. Se temos três condições A, B e C, temos no total 3 comparações:  $A \times B$ ,  $A \times C$  e  $B \times C$ . Não entraremos em detalhes sobre como os testes *post-hoc* são calculados, mas os mais comuns são o teste de Student com correção Bonferroni ou Holm e o teste de Scheffe [Robertson and Kaptein 2018].

participante	p0	p1	p2	p3	p4	p5	p6	p7	p8	p9
Sem Predição	5.3	3.5	5.1	3.6	4.6	4.1	4.0	4.8	5.2	5.1
Augkey	5.7	4.8	5.1	4.6	6.1	6.8	6.0	4.6	5.5	5.6

**Tabela 2.1. Tempos médios obtidos por participante no Exp1.**

participante	p0	p1	p2	p3	p4	p5	p6	p7	p8	p9
Sem Predição	2.4	2.7	3.3	6.1	6.7	5.4	7.9	1.2	3.0	6.6
Augkey	6.9	7.3	2.9	1.8	7.8	9.2	4.4	6.6	4.8	3.1

**Tabela 2.2. Tempos médios obtidos por participante no Exp2.**

### 2.3.1. Exemplo de ANOVA

Agora vamos aprofundar um pouco mais sobre o teste ANOVA. Para facilitar o entendimento dessa técnica, vamos considerar dois experimentos hipotéticos distintos, Exp1 e Exp2, ambos com o mesmo objetivo e metodologia para avaliar o desempenho de pessoas usando os teclados sem predição de palavras e o AugKey. Considere que 10 pessoas participaram de cada experimento e que a velocidade média de digitação medida em caracteres por minuto foi calculada para cada participante usando algumas frases neutras. A média dos tempos obtidos por participante em todas as sessões para o Exp1 são mostradas na Tabela 2.1 e as médias para o Exp2 são mostradas na Tabela 2.2.

Uma vez tabelados os dados, a média das médias pode ser calculada considerando todos os participantes como mostrado na Figura 2.6, indicando o desempenho de cada teclado. Observe que as velocidades médias usando cada teclado são as mesmas nos dois experimentos. Apesar do gráfico mostrar uma vantagem do AugKey, apenas no Exp1 observou-se diferença significativa entre as velocidades.

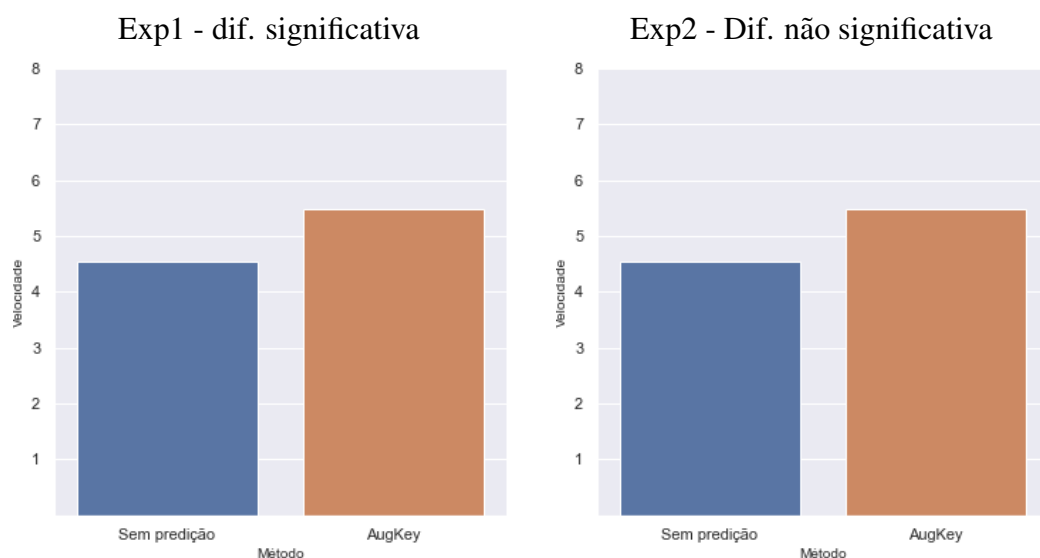
Na aplicação de ANOVA nesse exemplo de teclados, a hipótese nula seria que todas as pessoas participantes possuem o mesmo desempenho usando os dois modelos, ou seja, não há efeito do tipo de teclado (sem predição ou AugKey) sobre a velocidade de digitação. No entanto, como a velocidade atingida por cada participante deve ser diferente, podemos considerar cada velocidade medida como uma amostra aleatória da mesma população. A rejeição da hipótese nula significa que as diferenças nas velocidades entre os grupos de participantes provavelmente (dentro do nível de significância) não são devidas ao acaso.

Os testes-F recebem esse nome em homenagem ao estatístico Ronald Fisher. A estatística-F é simplesmente uma razão de duas variâncias e pode ser definida como:

$$F = \frac{\text{variância entre as médias das amostras}}{\text{variância dentro das amostras}}$$

Para entender essa fórmula, vamos voltar ao experimento onde 10 participantes usaram os teclados sem predição de palavras e AugKey. Porque a diferença foi significativa no gráfico à esquerda mas não foi significativa no da direita? Você já deve ter percebido pelas tabelas que a resposta está na variância das amostras, ou seja, os valores obtidos no experimento Exp2 estão mais espalhados (distantes da média) que no Exp1.





**Figura 2.6. Médias obtidas nos experimentos Exp1 e Exp2. Ambos os experimentos avaliam a velocidade de digitação usando os teclados sem predição de palavras e AugKey. Observe que as médias nos dois experimentos é a mesma. Mas observe também que apenas no Exp1 houve diferença significativa nos resultados.**

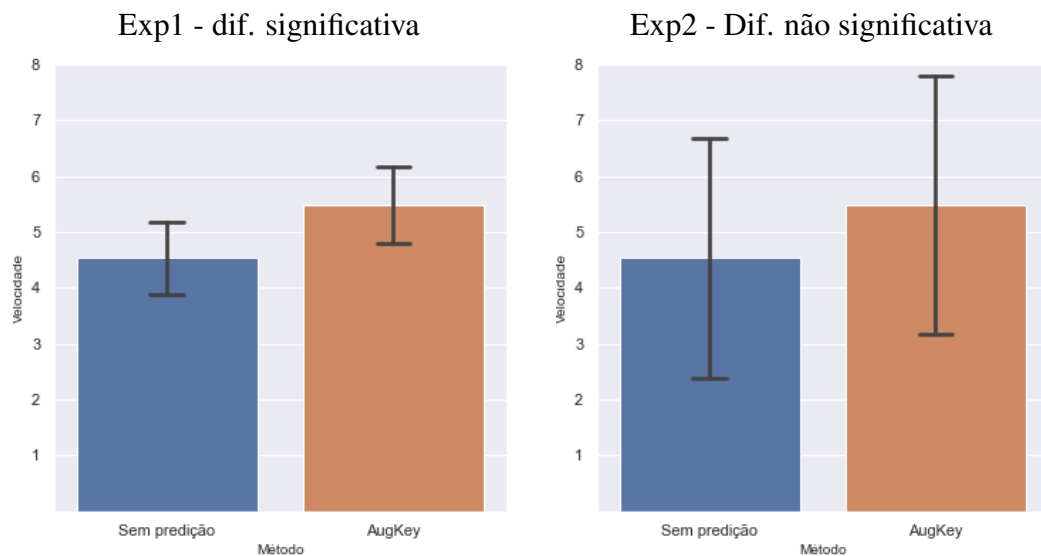
A variância é calculada como o quadrado do desvio padrão e indica a dispersão dos dados em relação à média. Variâncias pequenas indicam que os dados não fogem muito da média. A vantagem de usar o desvio padrão é que sua unidade é a mesma da média e portanto mais fácil de entender (pode ser colocada no gráfico da média por possuir a mesma unidade).

Como podemos observar na Figura 2.7, o desvio padrão, representado pelas linhas verticais no centro das barras que indicam as médias, é menor no Exp1 e maior no Exp2 (gráfico da direita). Com isso podemos afirmar que o valor de F no Exp1 será maior que o valor de F no Exp2, pois a variância dentro das amostras é maior no Exp2.

A Tabela 2.3 mostra os dados de média, desvio padrão, variância e resultado do teste ANOVA para esse exemplo. Como podemos observar, o valor de F foi maior no Exp1 e o valor de p foi menor que 0.05, indicando que a diferença nas médias de velocidade de digitação nos teclados foi estatisticamente significativa. Já no Exp2 o valor de F foi menor e o valor de p acima de 0.05, indicando que a diferença não foi estatisticamente significativa.

**Tabela 2.3. Dados de média, desvio padrão, variância, valor de F e valor de p para os dados de Exp1 e Exp2.**

Métrica	Experimento 1		Experimento 2	
	Sem predição	AugKey	Sem predição	AugKey
Média	4.53	5.48	4.53	5.48
Desvio padrão	0.65	0.68	2.15	2.31
Variância	0.42	0.46	4.64	5.33
F	10.5		0.7	
p	<b>0.01</b>		0.4	



**Figura 2.7. Médias e desvios padrão obtidos nos experimentos Exp1 e Exp2. Observe que o desvio é bem maior em Exp2 e por isso os resultados de Exp2 não apresentam diferença significativa.**

A modo de exemplo, vamos calcular o valor de F para o Exp1. A variância total observada nos dados coletados em um experimento intra-sujeitos pode ser quebrada em três termos: a variância explicada pelas condições, a variância explicada pelas variações entre os participantes e a variância que não pode ser explicada (o erro residual).

Para calcular o valor de F, primeiro vamos calcular o valores descritos a seguir:

- Soma dos quadrados total:  $SS_t$  é a variância total, considerando todas as amostras juntas. No Exp1  $SS_t = 13.3$ ;
- Soma dos quadrados entre as condições:  $SS_b$  é a parte da variância justificada pelas diferentes condições experimentais (por exemplo, por usar um teclado sem predição e outro com AugKey). No Exp1  $SS_b = 4.51$ ;
- Soma dos quadrados dos participantes:  $SS_s$  é a variância explicada pelas características de cada participante no desempenho das tarefas, que no Exp1 é  $SS_s = 3.87$ ;
- Soma dos quadrados do erro:  $SS_e$  é a variância que não é explicada nem pelas condições experimentais e nem pelos participantes (erro residual). Este pode ser calculado a partir dos outros valores. Como já sabemos, a soma dos quadrados total é a soma dos três já estudados:

$$SS_t = SS_b + SS_s + SS_e.$$

Logo temos que

$$SS_e = SS_t - SS_b - SS_s = 3.87.$$

Até agora calculamos a soma dos quadrados, mas precisamos calcular a média de cada soma. Para isso precisamos apenas dividir pelos graus de liberdade de cada um. O **grau de liberdade** da soma dos quadrados das condições é igual ao número de condições menos um, já o grau de liberdade da soma dos quadrados do erro é igual ao número de condições menos um vezes o número de participantes menos um. Vamos ver um exemplo.

As médias de interesse para calcular F são calculadas assim:

$$MS_b = \frac{SS_b}{n. \text{ condições} - 1} = \frac{4.51}{2 - 1} = 4.51$$

e

$$MS_e = \frac{SS_e}{(n. \text{ condições} - 1) * (n. \text{ part.} - 1)} = \frac{3.87}{(2 - 1) * (10 - 1)} = \frac{3.87}{9} = 0.43$$

Finalmente, o valor de  $F = \frac{MS_b}{MS_e} = 10.5$ . Os graus de liberdade de F são (n. condições - 1, (n. condições - 1)\*(n. part. - 1)) = (1, 9). Logo podemos afirmar que  $F(1, 9) = 10.5$ .

Agora, como descobrir se esse valor de F representa uma diferença significativa entre os dois teclados? Para isso é necessário consultar uma tabela de valores críticos de F. Esta tabela tem por colunas o grau de liberdade do numerador na expressão de F e por linhas os graus de liberdade do denominador. Após localizar o valor correspondente aos graus de liberdade da expressão de F, comparamos o valor da tabela com o F calculado: se o valor de F for maior ou igual ao valor da tabela, então o resultado foi estatisticamente significativo. Caso o valor de F for inferior ao valor da tabela, dizemos que o resultado não é significativo, ou seja, que não houve um **efeito** significativo. É importante levar em consideração que cada tabela serve para um determinado nível de significância, no nosso caso usamos o valor de 0.95 como nível de significância, mas caso outro nível seja usado (por exemplo 0.99 ou 0.90), será necessário consultar a tabela de valores de F correspondente a esse nível. Na prática porém, o mais comum é usar um nível de significância de 0.95.

Como vimos, a tabela de valores de F não nos fornece o valor exato de p, mas nos diz se o resultado é significativo ou não para um determinado nível de significância. Na prática, a maioria das bibliotecas estatísticas já calculam o valor exato de p quando executam o teste ANOVA e assim, podemos comparar o valor de p com (1 - nível de significância) e, se o valor de p for menor, então o resultado é estatisticamente significativo. No exemplo Exp1, o valor de p calculado pelo software resultou em 0.01. Como 0.01 < 0.05, o resultado é estatisticamente significativo. Note que o valor 0.05 é o resultado de 1 - 0.95, que é o nível de significância do nosso estudo. Uma outra forma de pensar nesses resultados é que a chance (probabilidade) de que os resultados serem diferentes é menor que 5%.

Como podemos reportar o resultado do teste ANOVA? Um outro fator importante da pesquisa é que seus resultados devem ser publicados. Afinal, encontrar uma diferença significativa é um resultado muito positivo para qualquer pesquisa. Recomendamos seguir o padrão sugerido na 6ª Edição do Manual da Associação Americana de Psicologia (APA, do inglês *American Psychological Association*), disponível em <http://www.apastyle.com>.

Esse padrão sugere que todos os valores estatísticos sejam arredondados para duas casas decimais, com exceção dos valores onde o nível de significância ( $p$ ) seja menor que 0.001, em cujo caso deve ser usada a notação " $p < 0.001$ ".

No caso de uma diferença estatisticamente significativa, como no Exp1, podemos escrever assim: *um teste ANOVA de 1 via mostrou um efeito significativo do tipo de teclado na velocidade da digitação,  $F(1, 9) = 10.5, p = 0.01$ .*

## 2.4. Processamento e análise de dados com Python

As medidas quantitativas coletadas durante os experimentos são tipicamente armazenadas na forma de tabelas que podem ser processadas por planilhas eletrônicas tipo Excel. No entanto, a exploração dos dados e a visualização de resultados pode ser feita de forma mais eficiente usando outros softwares voltados para análise de dados como Matlab, Octave, IDL e SciLab.

Nesse curso vamos utilizar o JupyterLab (<https://jupyter.org>), um ambiente web interativo que permite desenvolver Jupyter Notebooks em Python (e que pode ser também utilizado com outras linguagens como R e Julia). Um Jupyter Notebook é um documento que pode misturar texto, código, equações e visualizações que facilitam a exploração dos dados.

Enquanto R é uma linguagem dedicada ao processamento estatístico dos dados e portanto adequada à análise experimental, adotamos Python nesse curso por ser uma linguagem de programação de propósito geral, poderosa, flexível e mais fácil de aprender por possuir uma sintaxe simples. Isso deve facilitar o acompanhamento dessa seção mesmo pelas pessoas com pouca experiência de programação e/ou experiência com outras linguagens. Por ser uma linguagem interpretada, ela também permite o desenvolvimento rápido de aplicativos em muitas áreas distintas, incluindo a análise exploratória de dados. Todos os exemplos que ilustram esse curso foram desenvolvidos usando Jupyter Notebooks em Python e estão disponíveis no endereço <https://ime.usp.br/~hitoshi/jai2021>.

### 2.4.1. Um pouco de Python

Nesse curso, vamos assumir que você já possui alguma experiência com uma linguagem de programação imperativa, como C, C++, Java ou Python (suficiente, por exemplo, para manipular uma estrutura de dados indexada como uma lista, vetor ou matriz) e também algum conhecimento de estatística (como saber calcular a média e a variância de um conjunto de valores). Assim podemos focar no uso do Python para a exploração e análise dos dados.

Caso você não tenha muita experiência e esteja interessado em um curso introdutório de programação em Python, recomendamos o material que utilizamos em nosso curso de introdução à computação disponível no endereço <https://panda.ime.usp.br/pensamentos>. Esse material é utilizado também no curso Introdução à Ciência da Computação com Python no Coursera, ministrado pelo Prof. Fábio Kon. Outra referência que recomendamos caso você deseje aprender probabilidade e estatística com Python é o livro *Think Stats* do Prof. Allen Downey disponível no endereço (<https://greenteapress.com/wp/think-stats-2e>), embora essa última referência esteja disponível

apenas em inglês.

O Python (<https://www.python.org>) oferece uma extensa coleção de módulos disponíveis gratuitamente para as principais plataformas computacionais modernas, como Linux, MacOS e Windows. Nessa seção vamos visitar alguns módulos que lhe podem ser úteis no processamento e análise estatística dos dados coletados em seus experimentos.

Caso você deseje instalar Python em seu computador para acompanhar os exemplos desse curso, sugerimos a instalação do pacote Anaconda (<http://anaconda.com>), desenvolvido para usar Python na área de ciência de dados. O Anaconda facilita a instalação e manutenção de todos os módulos e ferramentas que vamos utilizar nessa seção, como o JupyterLab, Numpy, Seaborn, Pandas e SciPy. Você pode baixar e utilizar a versão individual do Anaconda gratuitamente.

#### 2.4.1.1. Criando e visualizando dados usando Numpy e Seaborn

Vamos iniciar com a descrição dos módulos Numpy e Seaborn como ilustrado no trecho de programa 2.1.

A primeira linha do programa 2.1 carrega o módulo Numpy (Numerical Python) sob o nome `np`. O Numpy inclui funções eficientes para criação e manipulação de matrizes ( qualquer estrutura n-dimensional), que inclui operações matemáticas, lógicas, álgebra linear, estatística, transformações etc. O Numpy se tornou em um módulo fundamental para a computação científica. Vários outros módulos científicos e matemáticos são derivados do Numpy, alguns dos quais veremos nessa mesma seção. Uma estrutura de dados fundamental do Python é a lista (tipo `list`), que permite manipular uma sequência indexada com elementos de qualquer tipo, inclusive outras listas. Uma lista em Python utiliza colchetes (`[]`) como delimitadores. Por exemplo:

```
lista = [ 12, 3.14, 'hello', [5, 6]],
```

corresponde a uma lista com 4 elementos, o inteiro 12 na posição 0, o float 3.14 na posição 1, a string 'hello' na posição 2, e a lista [5, 6] na posição 3 da lista.

##### Programa 2.1. Uso de Numpy e Seaborn

```
1 import numpy as np
2 import seaborn as sns
3
4 pts = np.linspace(-np.pi , np.pi , 30)
5 valCos = np.cos(pts)
6 graf = sns.scatterplot( x=pts , y=valCos )
```

A estrutura básica do Numpy é o **ndarray** (vetor n-dimensional), que permite representar elementos de um mesmo tipo (ou seja, ao contrário de listas, seu conteúdo é homogêneo). Enquanto as listas são alocadas dinamicamente, o tamanho de um ndarray não é mais alterado após sua criação. A alteração de tamanho implica na criação de um novo ndarray. Uma lista homogênea pode ser convertida para um ndarray usando a função de conversão `array()` como

```
nda = np.array([1, 2, 3, 4]).
```

Apesar (ou devido) a essas limitações, as operações com Numpy são executadas de forma mais eficiente e com menos código do que usando listas pois o Numpy permite operações vetoriais, evitando o uso explícito de laços.

Um exemplo de operação vetorial pode ser visto na linha 5. Na linha 4, a função `np.linspace()` do Numpy (observe a notação com ponto que indica módulo.função) cria um `ndarray` com 30 pontos no intervalo  $[-\pi, \pi]$ , uniformemente espaçados, e os atribui à variável `pts`. A função `np.cos()` é aplicada a todos os pontos do `ndarray`, sem necessidade de criar um laço explícito para percorrê-lo. Os valores resultantes são atribuídos à variável `valcos`.

A segunda linha do programa 2.1 carrega o módulo Seaborn sob o nome `sns`. O Seaborn (<https://seaborn.pydata.org/>) é baseado no Matplotlib, um módulo do Python para criação de visualizações de dados bastante flexível e utilizado, por exemplo, para a apresentação de gráficos. O Seaborn é um módulo mais simples que o Matplotlib, voltado a criar gráficos visualmente mais atraentes. Mas caso necessário, o Seaborn pode ser utilizado em conjunto com o Matplotlib para combinar seus recursos.

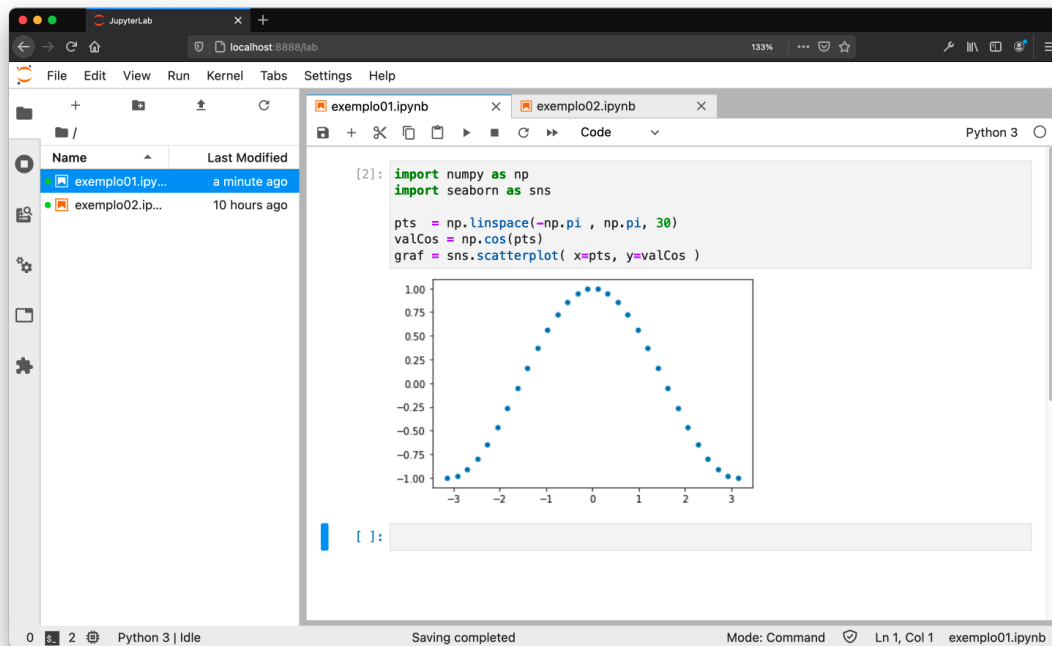
A linha 6 mostra como é simples criar um gráfico de pontos (*scatterplot*) usando Seaborn. Nesse caso, basta associar o vetor de pontos horizontais em `pts` com seus valores correspondentes no eixo vertical em `valcos`. Observe que o gráfico é atribuído à variável `graf`, para podermos alterar certas propriedades mais tarde, como os nomes de cada eixo.

## 2.4.2. Explorando dados usando Jupyter Lab

O Jupyter Lab (<https://jupyterlab.readthedocs.io>) é um ambiente web (ou seja, roda em um navegador da Internet como o Mozilla Firefox, Google Chrome ou Microsoft Edge) que fornece blocos de construção flexíveis para computação exploratória interativa. Embora o Jupyter Lab tenha muitos recursos encontrados em ambientes de desenvolvimento integrado (conhecidos como IDEs do inglês *Integrated Development Environment*) tradicionais, ele permanece focado na computação exploratória e interativa.

A interface do Jupyter Lab pode ser vista na Figura 2.8, rodando dentro do navegador Mozilla Firefox. A figura mostra o navegador de arquivos (*file browser*) aberto, que mostra os Jupyter Notebooks (arquivos com a extensão “.ipynb”) disponíveis na pasta do projeto. Quando esses arquivos são abertos, como ilustrado na figura, cada arquivo recebe uma aba na área principal de trabalho (para edição/exploração de dados), como se fossem páginas distintas do navegador.

A Figura 2.8 mostra o conteúdo do Jupyter Notebook armazenado no arquivo “exemplo01.ipynb”, que corresponde ao programa 2.1, após a execução do bloco. A área de trabalho atua como um ambiente de desenvolvimento Python, permitindo a edição, visualização e testes do programa. Cada bloco pode ser executado de forma independente dos demais e o resultado é exibido instantaneamente, logo abaixo do bloco executado. No entanto, como todos os blocos compartilham do mesmo interpretador (iPython kernel), a ordem de execução dos blocos pode afetar o resultado. Essa flexibilidade facilita a exploração de dados mas pode se tornar confusa em problemas mais complexos. Por isso



**Figura 2.8. Interface do Jupyter Lab**

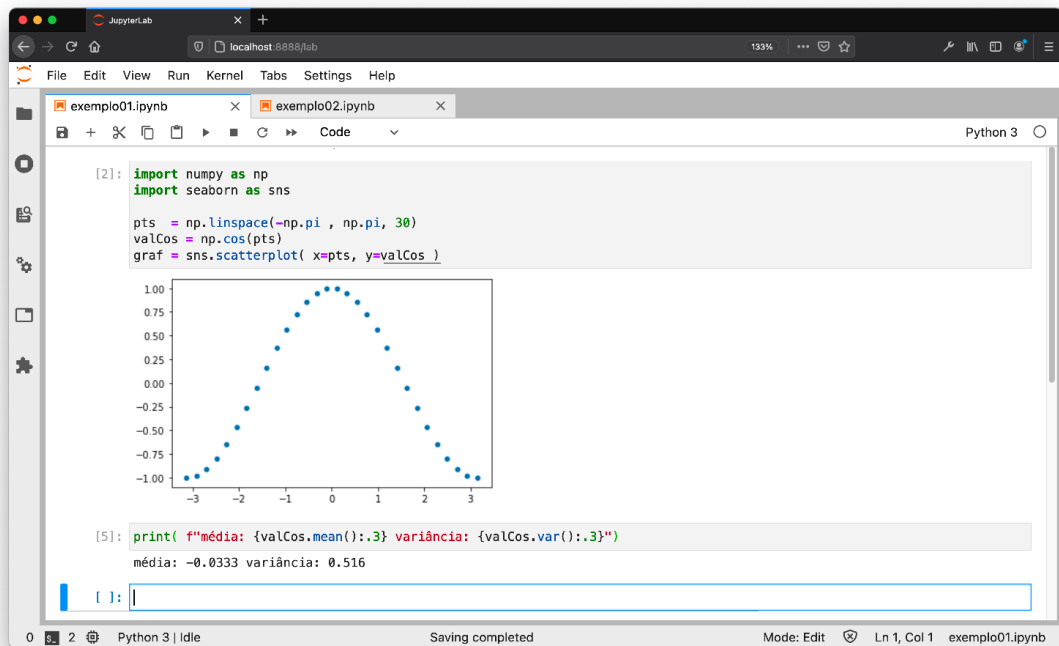
o iPython kernel pode ser reinicializado quando necessário. O Jupyter Notebook permite também alterar a ordem dos blocos e selecionar apenas um grupo de blocos para serem executados.

Para ilustrar essa característica, a Figura 2.9 mostra a mesma interface agora com o navegador de arquivos fechado e, após o gráfico de pontos, vemos que a pessoa usuária digitou um comando Python para imprimir a média e a variância de `valCos`. Ao escrever a linha com a função `print()`, executamos a linha algumas vezes, por exemplo para alterar o padrão como limitar em 3 o número de dígitos após a vírgula, e por isso executamos o mesmo bloco algumas vezes. Por isso o número entre colchetes desse bloco aparece como “[5]” na figura.

### 2.4.3. Pandas: processamento de dados em tabelas

O Numpy oferece várias funções para a geração de dados sintéticos segundo algumas distribuições comuns que são muito úteis para simulação e testes. Vamos criar uma tabela com dados sintéticos para introduzir o Pandas e usar um pouco mais do Jupyter Notebook.

O Pandas (<https://pandas.pydata.org>) é um módulo do Python voltado para a análise de dados usando **DataFrames**, uma estrutura equivalente a uma tabela ou planilha (aliás, a origem do nome “Pandas” vêm da combinação das palavras “panel” + “data” em inglês). Embora o Pandas ofereça muitos recursos próprios, nesse curso vamos utilizar o Pandas apenas para carregar, salvar e manipular tabelas. Um DataFrame é diferente de um ndarray de duas dimensões do Numpy pois cada coluna possui nomes, e os tipos de dados de cada coluna podem ser distintos, além de fornecer mecanismos elaborados para seleção de dados e sua manipulação.



**Figura 2.9.** Interface do Jupyter Lab: cada bloco (indicado pelo número em colchetes) contém um pedaço de código que pode ser explorado de forma independente dos demais.

### Programa 2.2. Criação de um DataFrame

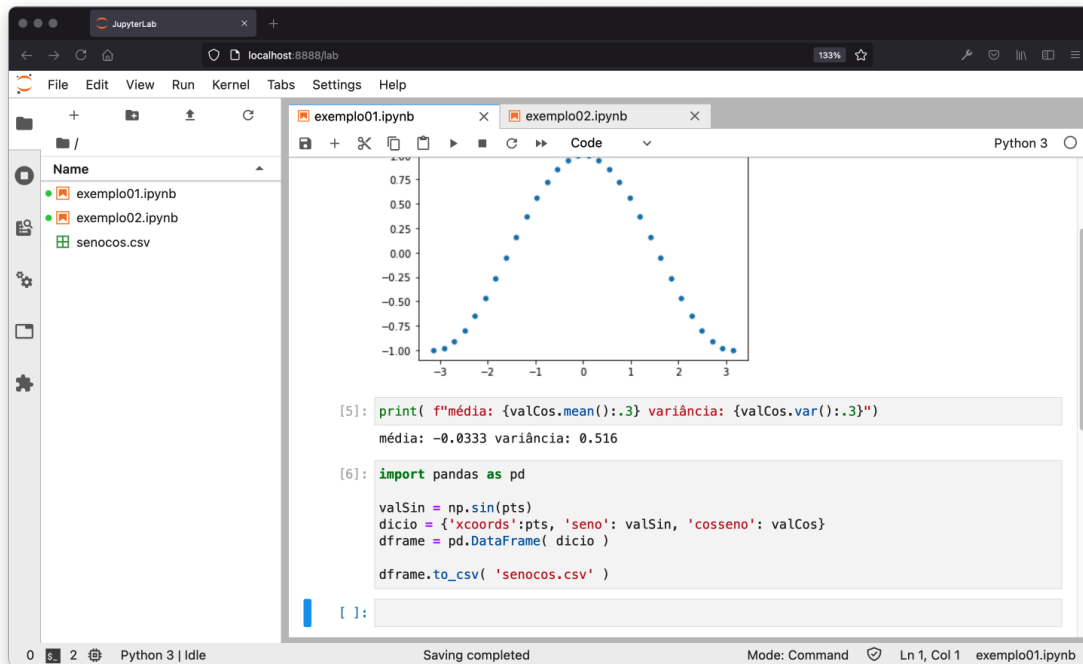
```
1 import pandas as pd
2
3 valSin = np.sin( pts )
4 dicio = { 'xcoords': pts, 'seno': valSin, 'cosseno': valCos }
5 dframe = pd.DataFrame( dicio )
6
7 dframe.to_csv( 'senocos.csv', sep=';' )
```

O trecho de código 2.2 ilustra como criar um DataFrame. Você pode colocar esse código em um novo bloco do Jupyter Notebook e executá-lo. A linha 1 mostra que o módulo pandas é carregado com o nome `pd`. Estamos considerando que o trecho de código 2.1 já tenha sido executado e portanto a variável `pts` já esteja carregada e pode ser usada para criar um novo ndarray com os valores do seno de `pts`.

Um DataFrame pode ser criado a partir de um dicionário do Python como nas linhas 4 e 5. Na linha 4, as chaves do dicionário indicam o nome de cada coluna, e os valores da coluna são definidos como uma lista ou ndarray. A linha 5 cria o DataFrame a partir do dicionário `dicio` e na linha 7 o DataFrame `dframe` é salvo no arquivo de nome "senocos.csv" e o caractere ';' como separador de colunas.

A Figura 2.10 ilustra o estado do Notebook após a criação e execução do bloco 2.2. A figura mostra o navegador de arquivos aberto, onde podemos notar que o arquivo "senocos.csv" foi criado no mesmo diretório do projeto.





**Figura 2.10. Notebook após a execução de um novo bloco para criação de um DataFrame.**

Assim como o "senocos.csv", os resultados dos experimentos são tipicamente representados na forma de tabelas e armazenados em arquivos semelhantes. Há vários outros formatos possíveis, como o formato XLS ou XLSX usados pelo Excel. Nesse curso vamos adotar o formato CSV (*Comma Separated Values*) por ser simples e facilmente portátil para outras aplicações, inclusive para o Excel. Outra vantagem desse formato é que seu conteúdo é legível e o caractere separador entre colunas pode ser redefinido, ou seja, não precisa ser uma "vírgula" e podemos adotar outros caracteres como "ponto-e-vírgula" (;) ou "dois-pontos" (:).

O trecho abaixo mostra as primeiras linhas do arquivo "senocos.csv" salvo pelo Pandas, onde as colunas aparecem separadas pelo caractere ";". Observe que a primeira linha contém os nomes das colunas (que podem corresponder ao nome das propriedades medidas em um experimento) e cada linha seguinte contém uma amostra com os valores das medidas realizadas de cada propriedade. Observe que o Pandas inicializa a primeira coluna com o índice de cada amostra. Como essa coluna não recebe um nome, a primeira linha do arquivo já começa com um ponto-e-vírgula.

```
;xcoords;seno;cos seno
0;-3.141592653589793;-1.2246467991473532e-16;-1.0
1;-2.9249310912732556;-0.21497044021102427;-0.9766205557100867
2;-2.708269528956718;-0.4198891015602648;-0.9075754196709569
...
```

Para testar se o arquivo foi salvo corretamente, experimente digitar e executar o trecho de código 2.3 em outro bloco do seu Jupyter Notebook.

### Programa 2.3. Leitura de um DataFrame

```
1 salvo = pd.read_csv( 'senocos.csv' )
2 print( salvo )
```

#### 2.4.4. Visualizando múltiplos dados

Como vimos o Seaborn permite criar visualizações simples rapidamente e com gráficos visualmente agradáveis. Mas para criar gráficos mais elaborados, precisamos recorrer ao módulo **Matplotlib**. Por exemplo, para plotar os dados de seno e cosseno no mesmo gráfico vamos usar o pacote **Pyplot** que faz parte do Matplotlib.

### Programa 2.4. Visualizando múltiplos dados

```
1
2 from matplotlib import pyplot as plt
3 sns.set()
4
5 fig, ax1 = plt.subplots()
6 ax2 = ax1.twinx()
7 ax1.plot(dframe[ 'xcoords' ], dframe[ 'cosseno' ])
8 ax2.scatter(dframe[ 'xcoords' ], dframe[ 'seno' ], color='r')
9
10 plt.show()
11 fig.savefig( 'sencos.png' )
```

O trecho de código 2.4 ilustra uma forma para combinar os dados do seno e do cosseno usando Pyplot. Digite esse programa em um novo bloco e verifique o resultado, que é mostrado na Figura 2.11. Esse trecho começa carregando o `pyplot` com o nome `plt`. Apesar de usar recursos do Pyplot, o comando na linha 2 especifica que os gráficos gerados devem continuar usando o padrão visual do Seaborn.

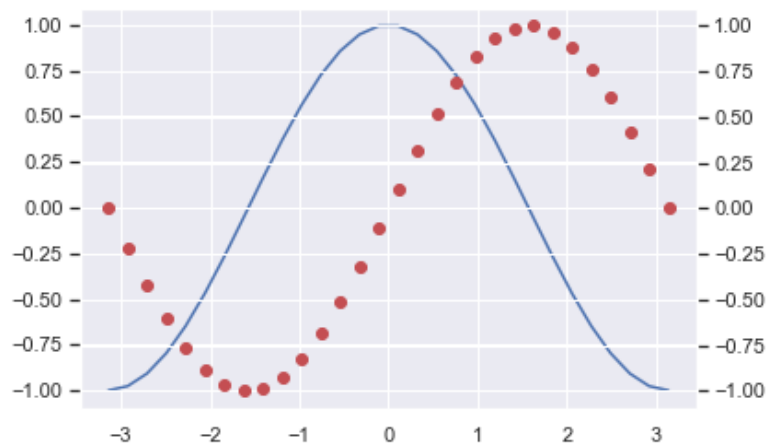


Figura 2.11. Gráfico gerado pelo trecho de código 2.4.

A função `subplots()` chamada na linha 5 devolve dois objetos: a figura e

um eixo (*axis*) para plotar o conteúdo gráfico. No caso das funções seno e cosseno, criamos um segundo eixo `ax2` na linha 6 para plotar um segundo conjunto de dados, mas compartilhando o mesmo eixo horizontal. Na linha 7, o conjunto `cosseno` é plotado na forma de uma **linha contínua** (usando `plot()`) no eixo `ax1`, usando os valores na coluna `'xcoords'` do DataFrame como pontos no eixo x. Na linha 8 o conjunto `seno` é plotado na forma de **pontos** (usando `scatter()`) no eixo `ax2`, na cor vermelha (`'r'`) – experimente `'b'` lue, `'g'` reen, etc).

Há várias outras opções que podem ser configurados (consulte a documentação do Seaborn e Pyplot) e, quando terminar de configurar, a linha 10 mostra como fazer o Pyplot exibir o gráfico. A figura pode também ser salva usando o método `savefig` como mostrado na linha 11.

#### Programa 2.5. Visualizando múltiplos dados separadamente

```
1 fig2, eixos = plt.subplots(1,2, figsize=(10,5))
2
3 sns.scatterplot(ax=eixos[0], x=dframe['xcoords'],
4                 y=dframe['cosseno'])
5 eixos[0].set_title('Cosseno')
6
7 sns.lineplot(ax=eixos[1], x=dframe['xcoords'],
8              y=dframe['seno'])
9 eixos[1].set_title('Seno')
```

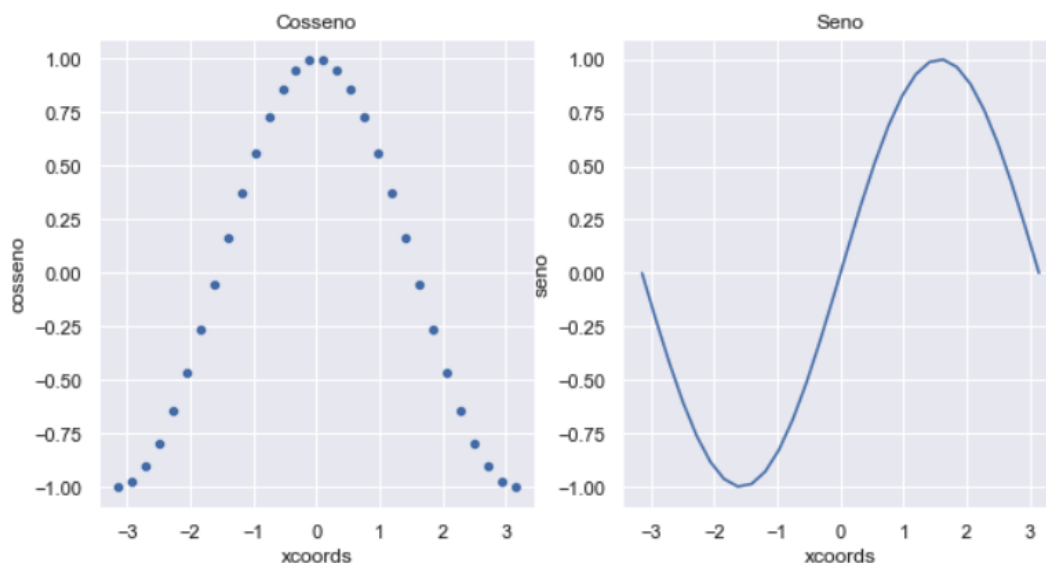
A função `subplot()` pode criar vários eixos para desenhar gráficos independentes, como ilustra a Figura 2.12, criado pelo trecho de programa 2.5. Você pode copiar e executar esse trecho em um novo bloco do Jupyter Notebook. Observe que na linha 1 a função `subplot()` recebe 3 valores, o número de eixos horizontais, o número de eixos verticais e o tamanho da figura `figsize` (nesse caso, a unidade do tamanho é em polegadas). A função portanto cria dois lugares (eixos), lado a lado, para desenhar gráficos.

As linhas 3 e 7 desenham as funções `cosseno` e `seno`, respectivamente, usando formas diferentes de desenho (funções de plotagem). Os parâmetros `ax` de cada chamada de desenho indicam uma posição (eixo) criada pela `subplot()` na figura `fig2`.

O Seaborn (e o Matplotlib) permite a criação de gráficos de vários tipos distintos, que podem ser classificados em 3 categorias: gráficos relacionais (como o do seno e cosseno exibidos nessa seção), gráficos de distribuições (como histogramas) e gráficos de categorias (como o gráfico de barras usado para exibir médias). Uma forma até divertida para aprender a gerar esses gráficos é dando uma olhada na galeria de exemplos (<https://seaborn.pydata.org/examples>) do Seaborn para procurar algum gráfico que você acha apropriado para exibir os seus dados e então copiar e adaptar o trecho de código em um bloco do Jupyter Notebook.

#### 2.4.5. Cálculo de ANOVA de 1 variável para um experimento intra-sujeitos

Agora podemos descrever como calcular uma ANOVA de 1 variável para o exemplo do estudo de teclados virtuais usando Python. Os passos para calcular o valor de F foram



**Figura 2.12.** Exemplo de uso do `subplot()` para mostrar os gráficos isolados na mesma figura.

apresentados na Seção 2.3.1. Recorde que no experimento foram coletados dados de 10 participantes que usaram os dois teclados: sem aceleração e AugKey. Para cada participante foi calculada a média do desempenho com cada teclado e armazenados no arquivo "dados\_teclados.csv". Você pode acessar esse arquivo junto ao material disponibilizado na página desse curso. O trecho de código Python abaixo cria as listas A e B com os mesmos valores dessa tabela, para que você possa copiar em seu programa. Fica como exercício transformar essas listas em um DataFrame. Esses são os mesmos valores usados na Seção 2.3.1 para o Exp1, ilustrados abaixo na forma de listas em Python.

```
A = [5.3, 3.5, 5.1, 3.6, 4.6, 4.1, 4.0, 4.8, 5.2, 5.1]
B = [5.7, 4.8, 5.1, 4.6, 6.1, 6.8, 6.0, 4.6, 5.5, 5.6]
```

Você pode ainda criar seu próprio arquivo .CSV com esses valores e, a seguir, abrir o arquivo e carregar os dados em um Pandas DataFrame, como mostrado no trecho de código 2.6.

**Programa 2.6. Código para carregar os dados do experimento dos teclados em um DataFrame**

```
1 df = pd.read_csv('dados_teclados.csv', sep=';', index_col=0)
2 print(df)
```

Para calcular a soma dos quadrados total, entre as condições e entre as pessoas participantes, precisamos calcular a média e a soma de todos os dados. Para isso usamos a função "flatten" que transforma os dados organizados em colunas em uma lista só, como mostrado no trecho de código 2.7.

**Programa 2.7. Código para calcular um termo comum usado em todas as expressões de soma de quadrados**

```
1 # Precisamos de media e a soma total de todos os dados
2 all_data_list = df.to_numpy().flatten()
3 mean_all = np.mean(all_data_list)
4 sum_all = np.sum(all_data_list)
5
6 # Este termo e subtraido de cada uma das somas de quadrados
7 common_term = sum_all**2 / (df.shape[0] * df.shape[1])
```

Agora calculamos a soma dos quadrados total, entre condições e entre participantes, como mostrado no trecho de código 2.8.

**Programa 2.8. Código para calcular as somas dos quadrados usadas no cálculo de F**

```
1 # Soma dos quadrados total
2 SST = sum(np.power(all_data_list, 2)) - common_term
3
4 # Soma dos quadrados entre as condicoes
5 SSB = sum(df.sum(axis=0).pow(2))/df.shape[0] - common_term
6
7 # Soma dos quadrados entre os participantes
8 SSS = sum(df.sum(axis=1).pow(2))/df.shape[1] - common_term
9
10 # Soma dos quadrados do erro
11 SSE = SST - SSS - SSB
```

Finalmente, calculamos os graus de liberdade e as médias da soma dos quadrados necessários para calcular o valor de F, como mostrado no trecho de código 2.9. No final da execução, o valor de F (arredondado) deve ser igual a 10.5.

**Programa 2.9. Código para calcular aos graus de liberdade, a média dos quadrados e o valor de F**

```
1 # Graus de liberdade de cada soma
2 SSS_df = df.shape[0]-1
3 SSB_df = df.shape[1]-1
4 SSE_df = SSS_df*SSB_df
5
6 # Calculando as medias da soma dos quadrados
7 MSb = SSB / SSB_df
8 MSs = SSS / SSS_df
9 MSe = SSE / SSE_df
10
11 # Finalmente, calculamos o valor de F
12 F = MSb / MSe
```

#### 2.4.6. Análise estatística usando Scipy

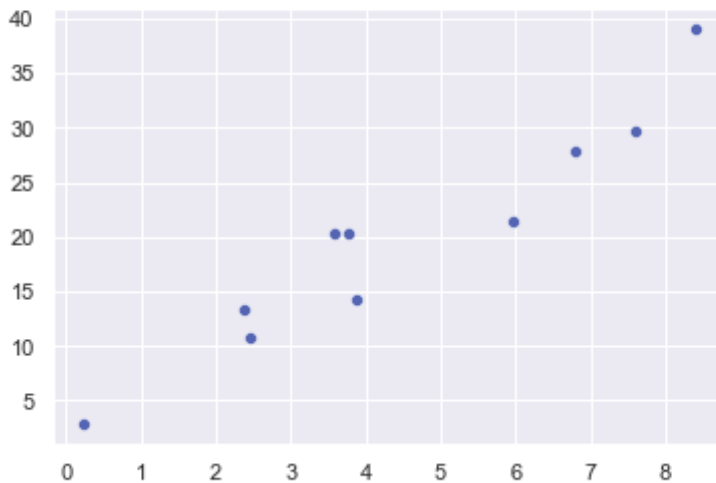
Por fim vamos introduzir o Scipy, um módulo de código aberto em Python que possui muitas ferramentas úteis para o processamento de dados científicos e complementam os

outros módulos já citados como Numpy e Seaborn.

Vamos ver um exemplo de como calcular a regressão linear usando Scipy, que é um recurso que vamos utilizar na próxima seção. A regressão linear permite estimar a relação linear entre dois conjuntos de dados: um de entrada X e um de saída Y. Vamos mostrar um exemplo de como fazer uma regressão linear usando Scipy e mostrar os resultados com Seaborn. A Figura 2.13 mostra um trecho de código com as listas X e Y que serão utilizadas como entrada para regressão linear, além do gráfico com a distribuição desses dados.

```
X=[2.45, 5.96, 3.58, 3.78, 0.24, 2.38, 3.87, 6.80, 8.39, 7.6]  
Y=[10.8, 21.5, 20.3, 20.4, 2.79, 13.4, 14.2, 27.76, 39.1, 29.6]  
  
sns.scatterplot(x=X, y=Y)
```

<AxesSubplot:>



**Figura 2.13.** Distribuição dos dados de entrada nas listas X e Y usados para regressão linear.

A seguir vamos importar o módulo "stats" do Scipy e rodar a regressão linear. O resultado da regressão retorna, entre outros, o valor de intersecção e a inclinação da reta obtida pela regressão linear, assim como o valor de  $R^2$ . O valor de  $R^2$  varia entre 0 e 1 e representa quão bom a reta resultante da regressão se ajusta aos dados de entrada (quanto mais perto de 1 melhor o resultado).

### Programa 2.10. Preparando dados para regressão linear

```
1  from scipy import stats
2
3  regressao = stats.linregress(X, Y)
4
5  slope = regressao.slope
6  inter = regressao.intercept
7  R2 = regressao.rvalue**2
8
9  print("Slope: _{ :0.2 f }".format(slope))
10 print("Intersecao: _{ :0.2 f }".format(inter))
11 print("Ajuste R2: _{ :0.2 f }".format(R2))
```

O resultado do trecho de código 2.10 é o seguinte:

```
Slope: 3.84
Interseção: 2.69
Ajuste R2: 0.92
```

Agora podemos mostrar a reta resultante da regressão nos dados de entrada, para ver quão perto cada ponto ficou da reta. O trecho de código no topo da Figura 2.14 pode ser usado para exibir os resultados no mesmo gráfico. Para desenhar a reta usamos os valores mínimo e máximo de  $X$  e calculamos os pontos correspondentes de acordo com os parâmetros da reta que obtivemos na regressão linear. Na figura podemos observar também como a reta obtida passa próximo da maioria dos pontos usados na regressão linear.

## 2.5. Parte Experimental

O objetivo dessa seção é realizar um experimento prático, onde poderão ser aplicados os conhecimentos teóricos apresentados no curso e usar as ferramentas estudadas para análise de dados. Trata-se de um típico experimento para avaliar o desempenho humano em uma tarefa simples, para levantar uma curva de desempenho conhecida como lei de Fitts [Fitts 1954].

### 2.5.1. Motivação

Muitas tarefas realizadas no dia a dia exigem movimentos mecânicos das nossas mãos, braços, dedos, pés, etc. Por exemplo, quando digitamos em um teclado de computador, apontamos e clicamos objetos com o mouse, ou ainda quando utilizamos um telefone celular para enviar uma mensagem de texto, navegar pela internet ou brincar com algum jogo eletrônico.

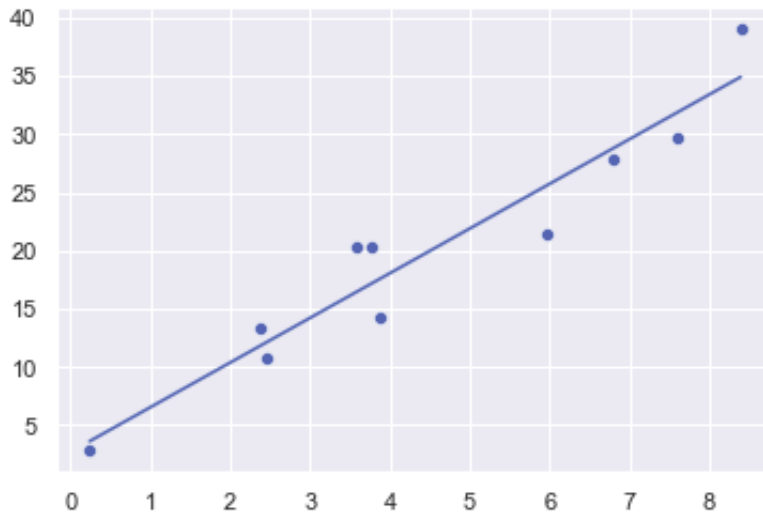
Suponha que queremos desenvolver um novo apontador laser para substituir o apontamento pelo mouse convencional. Algumas perguntas que podemos estar interessados em responder são:

- Como saber se o novo dispositivo será mais rápido e eficiente para apontamento comparado com o mouse?

```
X_reg = [min(X), max(X)]
Y_reg = [X_reg[0]*slope + inter, X_reg[1]*slope + inter]

sns.scatterplot(x=X, y=Y)
sns.lineplot(x=X_reg, y=Y_reg)
```

<AxesSubplot:>



**Figura 2.14.** Trecho de código e correspondente resultado da regressão linear. Observe como a maioria dos pontos usados na regressão linear estão bem próximos da reta.

- O novo dispositivo é mais rápido para apontar objetos que estão mais próximos entre si ou mais afastados?
- Qual é a relação entre o tamanho dos alvos e o tempo de apontamento no novo dispositivo?

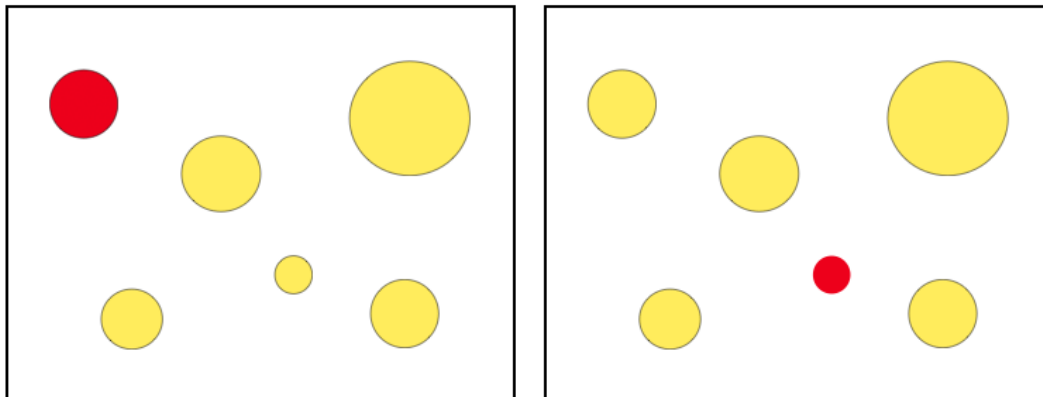
### 2.5.2. Experimento

Para responder cada uma dessas (e outras) perguntas, podemos fazer um experimento distinto. Por exemplo, podemos exibir alvos espalhados em um monitor, e pedir para uma pessoa voluntária que clique usando um dos dispositivos no alvo "vermelho", como mostrado na Figura 2.15. Após clicar em um alvo, um novo alvo se torna vermelho e deve ser clicado. O experimento pode consistir em clicar vários alvos e medir o tempo de seleção entre dois "cliques" consecutivos.

Nesse caso, o tempo médio de seleção por participante poderia ser utilizado para comparar o desempenho dos dois dispositivos usados para apontamento. A análise dos resultados usaria ANOVA com um fator (mouse ou laser).

Você mesmo pode fazer um experimento assim, por exemplo, recortando alguns pedaços de papel e escrevendo números sobre eles. Ou ainda, simplificando a tarefa para dois alvos apenas, e medindo o tempo para tocar em cada um, alternadamente, por





**Figura 2.15. Tarefa típica de apontamento. Considere a tarefa de clicar no alvo vermelho na tela da esquerda. Após selecionar o alvo, um novo alvo vermelho é exibido e deve ser selecionado (clicado na sequência).**

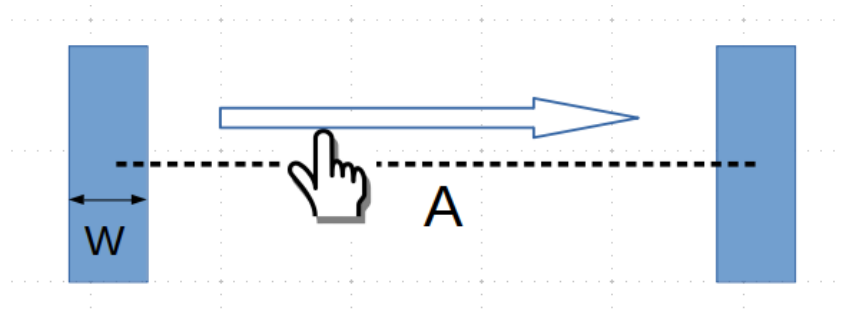
um certo número de vezes (como 10 ou 20 vezes). Peça para outras pessoas fazerem essa tarefa algumas vezes (3 ou 4 por exemplo) usando cada mão (esquerda e direita), enquanto você mede o tempo com um cronômetro e anota os resultados. Depois de coletar os dados, pode usar as ferramentas computacionais apresentadas na seção anterior para comparar o tempo de apontamento entre dois tipos distintos de apontamento (fator), que correspondem à mão direita e à mão esquerda. Talvez seja ainda melhor comparar o desempenho da mão dominante e não dominante, para considerar destros e canhotos.

A segunda e terceira perguntas indicam a possibilidade de haver outros fatores envolvidos e que podem afetar a velocidade de apontamento. Afinal, para apontar é necessário mover o cursor de uma certa distância (portanto quanto maior a distância menor deve ser a velocidade) e também parece ser mais difícil posicionar o cursor sobre um alvo pequeno. Parece intuitivo que, quanto menor o alvo, mais difícil é atingir ou tocar no alvo.

Novamente, podemos fazer mais experimentos com apenas 2 alvos, aumentando a distância e também variando o tamanho do papel usado como alvo. Por exemplo, poderíamos usar 3 tamanhos de alvo e 3 distâncias diferentes, resultando em 18 condições distintas: 2 métodos, 3 tamanhos e 3 distâncias. Repare como a complexidade do experimento aumenta com o número de fatores. Se cada participante do experimento precisar repetir uma condição 3 vezes, cada pessoa teria de fazer 54 tarefas de apontamento ao todo. A análise dos resultados também se torna mais complicada devido ao aumento do número de fatores e das possíveis interações entre eles.

Além de ajudar a testar hipóteses (responder as perguntas de pesquisa), outro objetivo de uma pesquisa pode ser comprovar uma teoria ou modelo. Note que esse é um objetivo bem mais ambicioso e que tem um maior potencial de contribuição científica também, como a Teoria da Relatividade de Einstein ou da Gravitação de Newton.

Mas não precisamos ser tão ambiciosos ou ir tão longe assim. Podemos tentar construir modelos (matemáticos) para prever o comportamento da resposta a certas



**Figura 2.16. Exemplo onde o tempo de movimento do dedo indicador entre os alvos de largura  $W$  a uma distância  $A$  pode ser modelado pela lei de Fitts**

variações dos fatores. Nesse sentido, a lei de Fitts [Fitts 1954] é um excelente exemplo muito utilizado em IHC que define um índice de dificuldade para uma tarefa de apontamento baseado na distância e no tamanho dos alvos, e pode ser usado para prever o desempenho humano sob certas condições.

### 2.5.3. Lei de Fitts

A lei de Fitts permite modelar o tempo de movimento  $MT$  entre alvos de largura  $W$  colocados a uma distância  $A$  (também chamada de amplitude do movimento), como é mostrado na Figura 2.16.

Para estimar o valor de  $MT$ , Fitts definiu antes o **índice de dificuldade** como sendo:

$$ID = \log_2\left(\frac{2A}{W}\right). \quad (1)$$

O tempo de movimento  $MT$  é definido com base no índice de dificuldade  $ID$  conforme a equação 2:

$$MT = a + b * ID. \quad (2)$$

Posteriormente, Mackenzie [MacKenzie 1992] sugeriu o uso de uma formulação semelhante à entropia de Shannon [Shannon and Weaver 1998] definida pela equação:

$$ID = \log_2\left(1 + \frac{A}{W}\right), \quad (3)$$

que oferece uma aderência melhor ao desempenho humano típico usando um mouse para apontamento, além de garantir que o índice de dificuldade é sempre um número não negativo.

Vamos ver agora como podemos interpretar, calcular os parâmetros e aplicar a Lei de Fitts para prever o desempenho humano em certas tarefas de apontamento. Para tal fim usaremos o índice de dificuldade como apresentado na equação 3.

Como podemos observar na equação 2, o tempo de movimento depende de dois valores  $a$  e  $b$  que são desconhecidos. O que eles representam e como podemos calculá-los? Se observarmos bem, o tempo de movimento é representado pela equação de uma

reta (equação linear ou de primeira ordem), onde o eixo  $X$  representa os valores do índice de dificuldade e o eixo  $Y$  representa o tempo de movimento.

Como já sabemos, em uma equação linear o valor de  $a$ , que é o termo independente, representa a intersecção da reta com o eixo  $Y$ . Quando o índice de dificuldade é zero, o valor de  $a$  é igual ao tempo de movimento (pois o termo  $b * ID$  é anulado). Como o valor de  $ID$  é sempre não negativo, podemos afirmar que o valor de  $a$  na equação 2 é o tempo mínimo possível de movimento. Mas porquê podemos afirmar que o  $ID$  é sempre não negativo? Analisando a equação 3 temos que o termo  $A/W$  é sempre não negativo, pois não faria muito sentido tocar ou clicar em um alvo de largura menor ou igual a zero (pois nesse caso o alvo seria invisível), assim como também não faria sentido pensar em distância entre alvos como sendo menor ou igual a zero, pois os alvos estariam sobrepostos. Logo temos que  $\log_2(1 + \frac{A}{W})$  será sempre não negativo.

O valor de  $b$  representa a inclinação da reta, ou seja, como varia o tempo de movimento em função do índice de dificuldade  $ID$ . Um valor positivo de  $b$  significa que a cada 1 unidade de incremento do índice de dificuldade, o tempo de movimento é acrescentado de  $b$  unidades. Já um valor negativo de  $b$  significa que um incremento no índice de dificuldade produz uma redução no tempo de movimento. Quanto maior for o valor absoluto de  $b$ , maior a sensibilidade do tempo de movimento em função do índice de dificuldade.

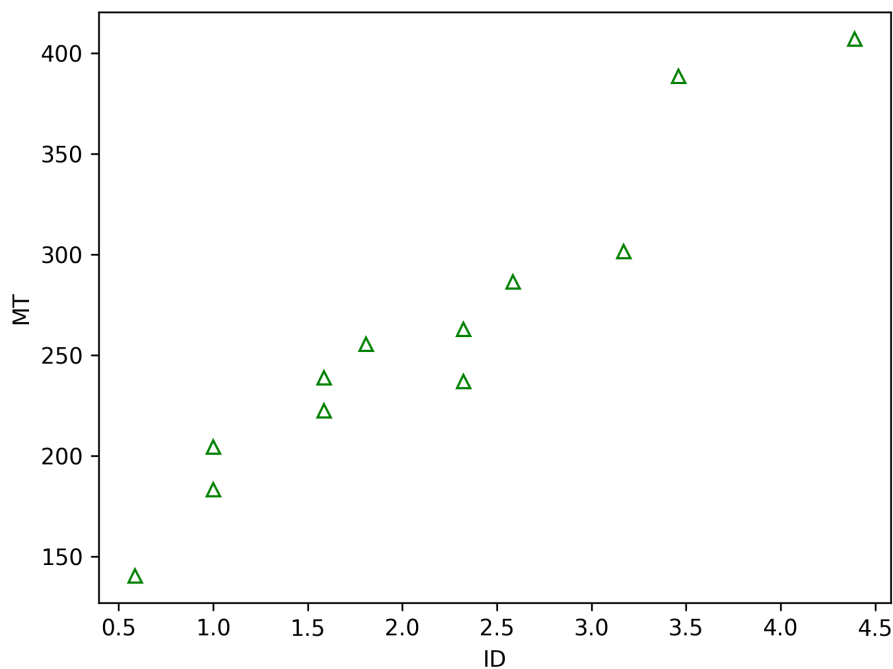
Agora que já sabemos interpretar os valores de  $a$  e  $b$ , como eles podem ser calculados? Uma alternativa é estimar esses valores de forma empírica, ou seja, medindo o comportamento real de várias pessoas, podemos estimar com uma boa precisão os valores de  $a$  e  $b$  para uma dada população.

Para isto precisamos calcular experimentalmente o tempo de movimento para vários valores de  $ID$ , afim de obter vários pontos para estimar os parâmetros da reta. Ou seja, precisamos avaliar o desempenho usando vários valores de  $ID$ , medindo o tempo  $MT$  para cada  $ID$  e depois usar uma técnica conhecida como regressão linear para estimar os valores de  $a$  e  $b$ .

A Figura 2.17 mostra os dados de tempo de movimento para vários valores de  $ID$  coletados de um participante. O eixo  $X$  representa os valores de  $ID$  e o eixo  $Y$  os valores de tempo de movimento medidos em milissegundos. Estes dados foram coletados durante um aula experimental sobre a lei de Fitts e, portanto, representam dados reais de um dos participantes do experimento.

Uma questão que surge é como definir os valores de  $ID$ . Como vimos na equação 3 o índice de dificuldade depende da largura dos alvos  $W$  e da distância entre eles  $A$ , que são as variáveis independentes do estudo. Então podemos escolher vários valores para  $W$  e  $A$  que façam sentido na vida real, que reflitam cenários onde os dispositivos de apontamento seriam usados. Como no exemplo que estamos estudando o apontamento foi realizado com o dedo sobre uma mesa, os valores de largura dos alvos foram 2, 5 e 10 cm e as distâncias entre os alvos foram 5, 10, 20 e 40 cm.

Uma vez definidos os valores de  $W$  e  $A$ , calculamos todas as combinações dessas variáveis independentes, que caracteriza um experimento com **desenho fatorial**, onde para cada nível de uma variável incluímos todos os níveis da outra. Para gerar todas as combinações de  $W$  e  $A$  escolhemos, para valor possível de  $W$ , todos os valores possíveis



**Figura 2.17.** Exemplo de coleta do tempo de movimento para vários valores de índice de dificuldade para um participante.

de  $A$ . Isto daria no total  $3 \times 4 = 12$  configurações possíveis. A Tabela 2.4 mostra todas as 12 configurações possíveis do estudo e a média do tempo de movimento para cada uma. Tal média foi calculada a partir de várias repetições para cada configuração experimental.

Como podemos notar na Tabela 2.4 o mesmo  $ID$  pode ser obtido a partir de mais de um par de valores de  $W$  e  $A$ . Esta situação não invalida o estudo realizado, pois existe um motivo por trás de tal repetição, que é a escolha de valores de  $W$  e  $A$  que fazem sentido na vida real. A escolha de outros valores de  $W$  e  $A$  de forma que o valor de  $ID$  não apareça repetido também seria um desenho válido.

O método de regressão linear fornecido pela pacote **Stats** da biblioteca **Scipy**, que faz parte do **Numpy**, recebe dois parâmetros de entrada: um vetor  $n$ -dimensional  $X$  com os valores das variáveis independentes (cada coluna corresponde a uma variável) e um vetor unidimensional  $Y$  com os valores da variável dependente.

No nosso exemplo temos apenas uma variável independente: o índice de dificuldade. Já a variável dependente é o tempo de movimento. O resultado da regressão linear com os dados de exemplo é mostrado na Figura 2.18. O valor de  $R^2$  obtido foi de 0.93, lembrando que quanto mais perto de 1, melhor a reta resultante da regressão se ajusta aos dados de entrada.

Os coeficientes  $a$  e  $b$  obtidos da regressão linear são 117.9 e 66.4 respectivamente. Assim, a equação da lei de Fitts para os dados do nosso exemplo resultante é:

$$\mathbf{MT} = 117.9 + 66.4 \times \mathbf{ID} \quad (4)$$

**Tabela 2.4. Configurações experimentais de largura dos alvos, amplitude do movimento, índice de dificuldade e tempo de movimento coletado experimentalmente.**

Largura $W$	Amplitude $A$	Índice de dificuldade $ID$	Média do tempo de movimento $MT$
2	5	2.32	225.5
	10	3.32	286.5
	20	4.32	388.5
	40	5.32	407
5	5	1	204.5
	10	2	239
	20	3	263
	40	4	301.5
10	5	0	140.4
	10	1	183.5
	20	2	222.5
	40	3	407.5

O valor de  $a=117.9$  é o tempo mínimo de movimento possível. Podemos entender esse valor como sendo o limite fisiológico, considerando que o nosso corpo precisa de um tempo para perceber o estímulo, transmitir a informação até o cérebro e acionar os músculos, que trabalham coordenadamente para completar a tarefa. Já o valor de  $b=66.4$  significa que a cada incremento de 1 unidade no índice de dificuldade, o tempo de movimento aumenta em 66.4 ms.

O que acontece se queremos estimar o tempo de movimento para outros valores de  $ID$  não considerados inicialmente no experimento?

Vamos supor que queremos saber como seria o tempo de movimento para alvos de largura 5 cm colocados a uma distância de 30 cm. Primeiro devemos calcular o  $ID$  conforme a equação 3:

$$ID_{teste} = \log_2\left(1 + \frac{30}{5}\right) = 2.8. \quad (5)$$

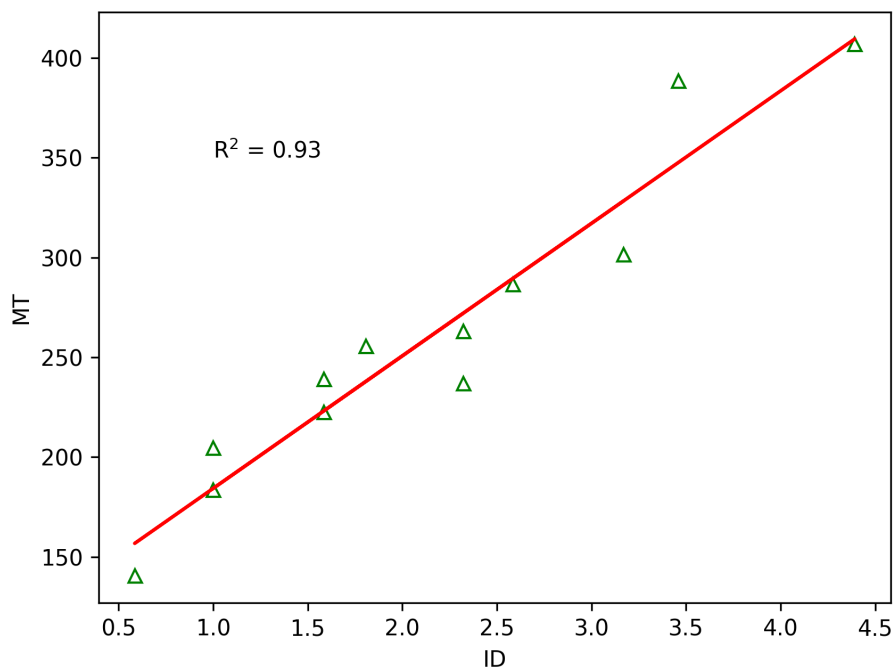
Agora calculamos o tempo de movimento:

$$MT_{teste} = 117.9 + 66.4 \times 2.8 = 304.3ms. \quad (6)$$

A Figura 2.19 mostra o novo ponto estimado usando a expressão da lei de Fitts.

## 2.6. Onde chegamos e para onde você pode ir

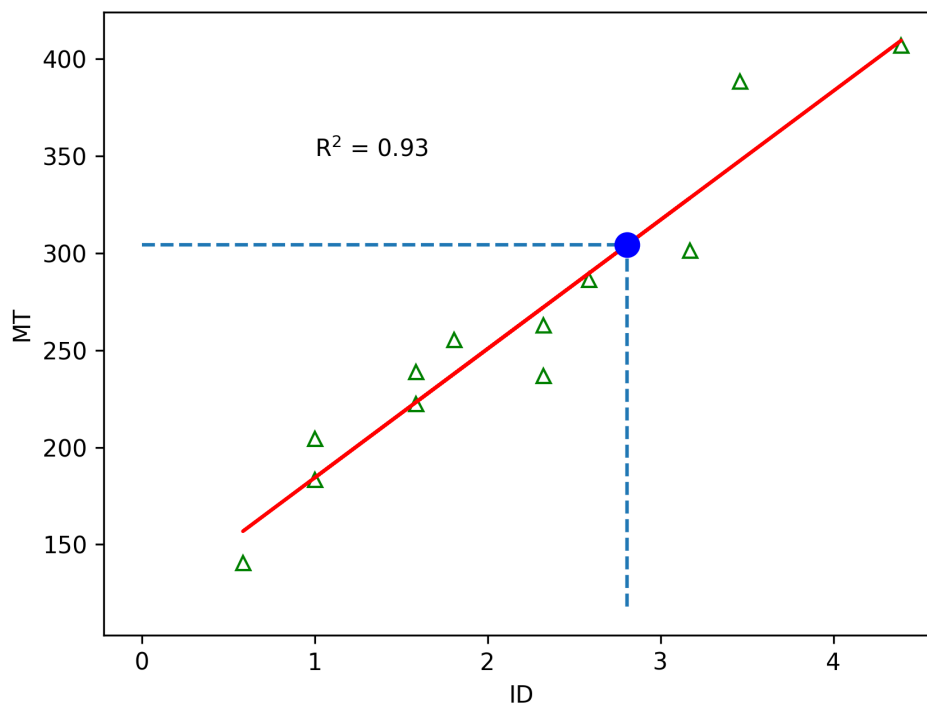
Nesse curso apresentamos brevemente vários tópicos relacionados ao método científico, análise estatística, e ferramentas computacionais para a exploração de dados. Nosso propósito foi acompanhar você numa rápida visita a alguns desses tópicos para, quem sabe, atrair você para fazer algumas visitas futuras, mais prolongadas, que conduzam você a realizar estudos com usuários que venham a enriquecer ainda mais a área de IHC e contribuir em alguma inovação tecnológica. Para se aprofundar nesses tópicos recomendamos a leitura do livro do Prof. Scott Mackenzie [Mackenzie 2012].



**Figura 2.18. Resultado da regressão linear mostrando a reta  $MT = 117.9 + 66.4 \times ID$  para os dados de exemplo.**

Nesse curso descrevemos como aplicar o método científico no projeto de experimentos com usuários de algum sistema computacional, além de algumas ferramentas computacionais existentes para a análise e exploração de dados na linguagem Python. Vimos que o projeto de qualquer experimento envolvendo humanos deve considerar valores éticos desde o início, devendo inclusive ser aprovado por um comitê de ética independente para que o experimento possa ser realizado. No Brasil, a avaliação e acompanhamento dos experimentos é realizado pelo Ministério da Saúde. Na área de IHC, vimos que uma grande maioria de experimentos busca utilizar dados quantitativos devido a sofisticação do conhecimento que resultados quantitativos podem apresentar em relação a dados qualitativos. Isso não diminui, no entanto, a importância de dados qualitativos nas pesquisas.

Seja quantitativo ou qualitativo, há vários testes estatísticos que podem ser aplicados para comprovar ou não a hipótese de pesquisa sendo investigada. Nos utilizamos o AugKey [Diaz-Tula and Morimoto 2016], um teclado de realidade aumentada desenvolvido pelos autores para auxiliar pessoas com deficiência motora na tarefa de entrada de textos com o olhar, como exemplo para introduzir e discutir vários conceitos e problemas relacionados ao desenho experimental. Para conhecer outros exemplos de aplicação do método científico em IHC, recomendamos a leitura de outros trabalhos publicados no CHI (*Conference on Human Factors in Computing Systems*) organizado pela ACM (*Association for Computing Machinery*), uma das principais conferências internacionais na área de IHC e com grande foco em estudos com usuários. O IHC (Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais) organizado pela SCB (*Sociedade Brasileira de Computação*) também é uma excelente referência para trabalhos nacionais



**Figura 2.19.** Estimando o tempo de movimento para um ponto não incluído no experimento, com largura 5 cm e distância de movimento 30 cm. O índice de dificuldade é de 2.8 e o tempo de movimento de 304.3 ms.

e, alguns deles, em português.

Um outro passo importante para a condução de experimentos é um bom conhecimento sobre as técnicas de análise paramétricas e não paramétricas. Dada a importância das técnicas paramétricas para analisar dados quantitativos e, em particular, da técnica de Análise de Variâncias (ANOVA) para a IHC, discutimos ANOVA com um pouco mais de cuidado para que você possa entender melhor os resultados apresentados em vários trabalhos na área.

Por fim, as ferramentas computacionais evoluem muito rapidamente e, apesar de existirem ferramentas mais específicas para análise estatística, achamos que apresentar um conjunto de ferramentas para análise e exploração de dados científicos baseadas na linguagem Python e no ambiente Jupyter Lab facilite o entendimento dos exemplos e a rápida adequação das ferramentas às suas necessidades. Todas as ferramentas apresentadas, como Numpy, Seaborn, Pandas e Scipy, são muito poderosas e merecem um visita futura mais cuidadosa, caso você tenha se interessado por alguma. Todas as ferramentas utilizadas também são de domínio público (*open source*) e podem ser baixadas e utilizadas gratuitamente. Todos os exemplos apresentados nesse curso estão disponíveis no endereço <https://www.ime.usp.br/~hitoshi/jai2021>.

## Referências

- [Buxton et al. 1985] Buxton, W., Hill, R., and Rowley, P. (1985). Issues and techniques in touch-sensitive tablet input. *SIGGRAPH Comput. Graph.*, 19(3):215–224.
- [Diaz-Tula and Morimoto 2016] Diaz-Tula, A. and Morimoto, C. H. (2016). Augkey: Increasing foveal throughput in eye typing with augmented keys. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3533–3544, New York, NY, USA. ACM.
- [Fitts 1954] Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 74:381–391.
- [Kaptein et al. 2010] Kaptein, M., Nass, C., and Markopoulos, P. (2010). Powerful and consistent analysis of likert-type rating scales. In *Proceedings of the ACM SIGCHI Conference on Human-Factors in Computing Systems—CHI 2010*, page 2391–2394, New York, NY, USA. Association for Computing Machinery.
- [Lazar et al. 2017] Lazar, J., Feng, J. H., and Hocheiser, H. (2017). *Research Methods in Human-Computer Interaction*. John Wiley & Sons.
- [Mackenzie 2012] Mackenzie, I. (2012). *Human-Computer Interaction: An Empirical Research Perspective*. Morgan Kauffmann Publishers, New York.
- [MacKenzie 1992] MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7(1):91–139.
- [Mackenzie et al. 1999] Mackenzie, I. S., Zhang, S. X., and Soukoreff, R. W. (1999). Text entry using soft keyboards. *Behaviour & Information Technology*, 18(4):235–244.
- [Majaranta 2009] Majaranta, P. (2009). *Text Entry by Eye Gaze*. PhD thesis, University of Tampere, Department of Computer Science, Tampere, Finland.
- [Morimoto and Mimica 2005] Morimoto, C. H. and Mimica, M. (2005). Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding*, 98:4–24.
- [Nielsen 1994] Nielsen, J. (1994). Enhancing the explanatory power of usability heuristics. In *Conference Companion on Human Factors in Computing Systems*, CHI '94, page 210, New York, NY, USA. Association for Computing Machinery.
- [Robertson and Kaptein 2018] Robertson, J. and Kaptein, M. E. (2018). *Modern Statistical Methods for HCI*. Springer.
- [Selker 2008] Selker, T. (2008). Touching the future. *Commun. ACM*, 51(12):14–16.
- [Shannon and Weaver 1998] Shannon, C. and Weaver, W. (1998). *The Mathematical Theory of Communication*. University of Illinois Press.



- [Sharp et al. 2019] Sharp, H., Preece, J., and Rogers, Y. (2019). *Interaction Design: Beyond Human-Computer Interaction*. Wiley, 5th edition.
- [Urbina and Huckauf 2010] Urbina, M. H. and Huckauf, A. (2010). Alternatives to single character entry and dwell time selection on eye typing. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, pages 315–322, New York, NY, USA. ACM.
- [Wobbrock et al. 2008] Wobbrock, J. O., Rubinstein, J., Sawyer, M. W., and Duchowski, A. T. (2008). Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the 2008 symposium on Eye Tracking Research & Applications*, ETRA '08, pages 11–18, New York, NY, USA. ACM.



## Capítulo

# 3

## Privacidade de Dados de Localização: Modelos, Técnicas e Mecanismos

Javam C. Machado, Eduardo R. Duarte Neto

### *Abstract*

*The growing development of mobile devices has promoted the increasing popularity of location services. However, the preservation of the users' privacy, especially the location data, has been questioned. This short-course describes the problem of the violation of the privacy of individuals' location and presents an in-depth analysis of the main techniques for their preservation, including differentially private mechanisms. Initially, fundamental concepts of data privacy will be presented, as well as vulnerabilities and threats to the privacy of individuals when exposing their location data when using applications on mobile devices. Then the state of the art in preserving location data privacy will be presented and discussed. Finally, we will point out research opportunities in the area and present relevant conclusions on the topic.*

### *Resumo*

*O desenvolvimento crescente de dispositivos móveis tem promovido uma crescente popularidade dos serviços de localização. Entretanto, a preservação de privacidade dos usuários destes serviços, em especial dos dados de localização, tem sido bastante questionada. Este minicurso descreve o problema da violação da privacidade de localização de indivíduos e apresenta um aprofundamento das principais técnicas para sua preservação, incluindo mecanismos diferencialmente privados. Inicialmente serão apresentados conceitos fundamentais de privacidade de dados, bem como vulnerabilidades e ameaças à privacidade de indivíduos ao expor seus dados de localização quando do uso de aplicações em dispositivos móveis. Em seguida será apresentado e discutido o estado da arte em preservação de privacidade de dados de localização. Por fim iremos apontar oportunidades de pesquisas na área e apresentar conclusões relevantes sobre o tema.*

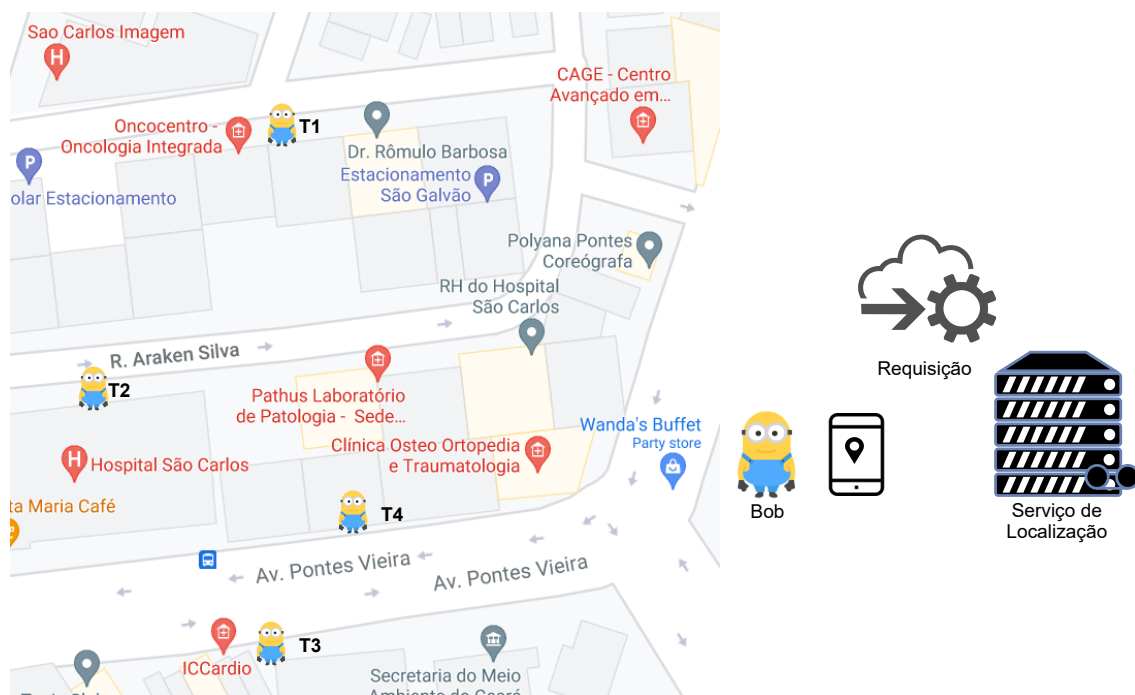
### 3.1. Introdução

Com o desenvolvimento dos dispositivos móveis, a quantidade de dados coletados por aplicativos a fim de prover os mais diversos serviços tem crescido bastante. Estes dados têm se mostrado bastante valiosos, sendo utilizados nas mais diversas áreas. Por exemplo, muitas empresas têm se utilizado da análise destes dados para traçar o perfil de seus consumidores e assim, adotarem estratégias que venham a potencializar seus lucros. Já na área de saúde, uma análise sobre dados de saúde combinado a dados de localização pode ajudar a identificar que áreas estão mais sujeitas a certos patógenos, e assim, adotar medidas que venham a melhorar o atendimento de seus cidadãos. Por exemplo, um estudo sobre a taxa de infecção de novos casos de COVID-19 por região poderia ajudar em conter o avanço da doença, tratando de forma mais eficiente as áreas mais afetadas. Esse tipo de análise requer acesso a dados privados, o que levanta questionamentos quanto à privacidade dos indivíduos a quem os dados pertencem. Logo, encontrar uma forma de permitir esta análise sem que haja riscos a exposição dos mesmos tem sido objeto de estudo na área de privacidade de dados.

O crescimento da popularidade dos serviços baseado em localização (LBS) tem contribuído bastante para o aumento da quantidade de dados gerados, em especial, dados referentes à localização dos seus usuários. Através dos sensores destes dispositivos, as coordenadas de latitude e longitude são obtidas e utilizadas por estes serviços. Schiller [43] define os serviços de localização como serviços que integram a localização ou posição de um dispositivo móvel a outras informações, de modo a fornecer valor agregado a um usuário. Estes numerosos serviços, tais como navegação, redes sociais, serviços de recomendação, jogos de realidade aumentada, entre outros, têm sido desenvolvidos e integrados às atividades diárias das pessoas, provendo informações úteis sobre seus arredores e sendo capazes de responder perguntas do dia a dia como: qual a melhor rota a ser percorrida para um determinado endereço? Quais os pontos turísticos mais próximos da minha localização atual? Em quanto tempo o táxi que eu solicitei irá demorar para chegar em meu apartamento?

Estas informações de localização geradas por estes serviços podem potencializar vários outros serviços. Empresas e agências governamentais têm utilizado as informações de localização, tais como atividades praticadas nas localizações, para melhorar o serviço prestado, para o lançamento de um novo produto, ou até mesmo para gerar uma nova política pela empresa. Entretanto, acessar dados de localizações de usuários desses serviços, mesmo que com permissão, levanta severas preocupações de privacidade para a maioria dos usuários. Dessa forma, a utilização de serviços baseados em localização pode levar a sérios riscos de violação de privacidade devido a provedores de serviços não confiáveis [28], que podem expor os dados de localização de seus usuários ou até mesmo vender suas informações de localizações a terceiros [54]. De posse dessas informações, os dados obtidos por terceiros são utilizados para descoberta de dados sensíveis dos usuários, *i.e.*, dados de saúde, crenças religiosas, ideologias políticas, questões raciais, preferências sexuais, dentre várias outras.

Para exemplificar o risco da exposição dos dados de localização, podemos observar a Figura 3.1. O usuário Bob realiza em vários momentos requisições a um serviço de localização qualquer. A cada requisição ele envia sua localização corrente. No tempo

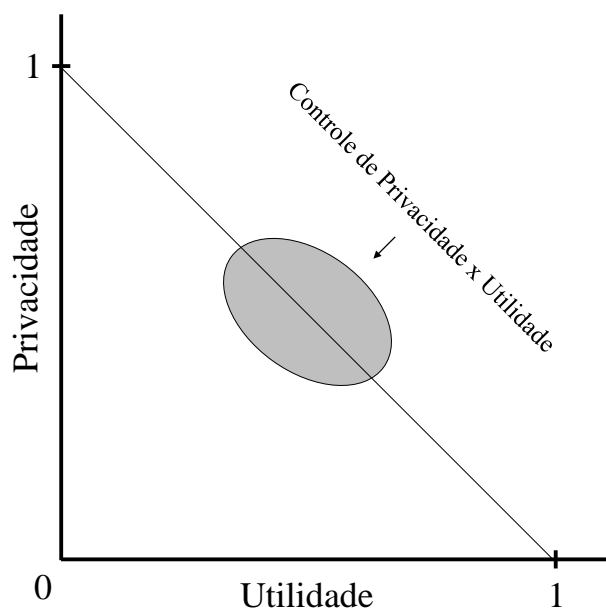


**Figura 3.1. Exemplo de requisições realizadas próximas a hospitais e clínicas, permitindo inferências de dados sensíveis do usuário.**

$t_1$ , Bob estava próximo a uma clínica de oncologia. No tempo  $t_2$ , Bob realiza uma nova consulta próxima ao pronto socorro de um hospital. Em outros dois momentos sua localização também está próxima a localizações associadas a área de saúde. Considerando que é de conhecimento do provedor do serviço as requisições feitas pelos usuários, o próprio provedor como um possível agente malicioso pode inferir, com alta probabilidade, que Bob possui algum tipo de doença, ou que ele é da área de saúde, ou está acompanhando alguém enfermo. Estas informações, juntamente com outras informações de contexto, podem aumentar ainda mais o sucesso da inferência sobre dados sensíveis de Bob. Desta forma, o risco de uma violação de privacidade é bastante alto, deixando o usuário exposto.

A aplicação de modelos de privacidade sobre requisições de usuários é imprescindível para evitar que as localizações dos indivíduos não sejam identificadas pelos provedores no uso destes serviços. Todavia, em geral os modelos de privacidade acabam provocando mudanças nos dados, afetando diretamente a sua utilidade, com impacto direto na qualidade do serviço. Portanto, gerenciar essa solução de compromisso (*trade-off*, Figura 3.2) entre privacidade dos indivíduos e utilidade dos seus dados se torna um outro grande desafio. Desta forma, vários modelos de privacidade de dados têm sido propostos por pesquisadores com o objetivo de resolver esta questão.

Este capítulo tem por objetivo introduzir os fundamentos e técnicas para preservação da privacidade de dados dos indivíduos, procurando apresentar os riscos mais comuns e as técnicas mais populares na solução do problema. Em seguida, apresentaremos um aprofundamento sobre o tema privacidade em serviços de localização, apontando os conceitos básicos sobre dados de localização, os tipos de ataques a que estão sujeitos, e os principais modelos de preservação de privacidade de dados de localização na atualidade.



**Figura 3.2. Trade-off entre privacidade e utilidade**

Na Seção 3.2 apresentaremos os princípios básicos sobre o tema, que tipos de dados estão sujeitos a violação, os principais tipos de ataques, e como a preservação de privacidade pode ser alcançada. Os modelos sintáticos mais populares para preservação de privacidade são descritos na Seção 3.3. A Seção 3.4 apresenta o modelo de privacidade diferencial, estado da arte em preservação de privacidade. Na Seção 3.5 abordaremos o tema privacidade em Serviços de localização, onde descreveremos os serviços de localização e sua arquitetura. Iremos também apontar o problema da exposição temporal sem dados de localização. A Seção 3.5.4.3 descreve os principais tipos de ataque que dados de localização estão sujeitos. Os modelos de privacidade em dados de localização são descritos na Seção 3.6. Apresentaremos desafios de pesquisa na Seção 3.7. E por fim, a Seção 3.8 apresenta as considerações finais do capítulo.

### **3.2. Fundamentos da Privacidade de Dados**

A privacidade é o direito que um indivíduo tem de manter seus assuntos pessoais e relacionamentos secretos [10]. O debate em torno do conceito de privacidade é uma matéria de extrema complexidade que muitas vezes é confundida com o conceito de segurança. Embora privacidade e segurança sejam temas relacionados, elas tratam de pontos bem distintos. No contexto de dados, a segurança define o controle de acesso durante o ciclo de vida do dado. Este controle se refere a regras específicas de quem está autorizado a acessar (ou não) determinados recursos. A forma como o acesso é realizado é papel da privacidade. Normalmente regida por leis e políticas de privacidade que definem o controle de acesso, permitindo a revelação da informação apenas por usuários autorizados. Entretanto, este controle de acesso não é suficiente para garantir a privacidade dos indivíduos, visto que os usuários com acesso àquelas informações podem ser maliciosos, e assim capazes de divulgar informações sensíveis acerca dos donos dos dados.

A quantidade de dados coletados tem aumentado bastante com o desenvolvimento e popularidade dos dispositivos móveis, tais como celulares, relógios inteligentes, dispositivos veiculares, dentre outros. Diversos serviços têm sido ofertados. Muitos dos quais exigem que o usuário abra mão da sua privacidade em favor da prestação destes serviços. Dessa forma, é fundamental identificar quais tipos de dados não devem ser divulgados, e portanto, aplicar técnicas que permitam a proteção destes dados garantindo a privacidade. Todavia, a qualidade de certos serviços pode depender diretamente da precisão destes dados privados. Tornando essencial que mesmo após a aplicação destas técnicas, os dados mantenham uma certa utilidade.

### 3.2.1. Privacidade em Microdados

De uma forma geral, os dados são representados por tabelas, onde cada linha da tabela corresponde a um registro no conjunto de dados, e as colunas contém os atributos dos registros. A esta representação dá-se o nome de microdados [20]. Os indivíduos estão associados a registros nestas tabelas. Os atributos são características ou propriedades dos indivíduos. No contexto de privacidade de dados, os atributos podem ser classificados em [9]:

1. **Identificadores explícitos:** são aqueles atributos que identificam de maneira única os indivíduos, como "CPF", "nome", etc., e devem ser removidos antes da publicação dos dados;
2. **Semi-identificadores:** são aqueles que não são identificadores explícitos, mas podem identificar o usuário, quando relacionados. "Data de nascimento" e "CEP" são exemplos de atributos semi-identificadores;
3. **Atributos sensíveis:** possuem informações sensíveis a cerca dos indivíduos, como "doença", "salário", etc.;
4. **Atributos não sensíveis:** são aqueles que não se enquadram em nenhuma das categorias citadas anteriormente.

Em privacidade de dados, os atributos sensíveis são aqueles de maior interesse porque apresentam potenciais danos ao seus donos em caso de divulgação. Por esse motivo, tais atributos necessitam ser protegidos. A Tabela 3.1 ilustra um exemplo de registros de indivíduos contendo atributos identificadores explícitos e semi-identificadores, que precisam ser protegidos.

### 3.2.2. Proteção e ataques à Privacidade

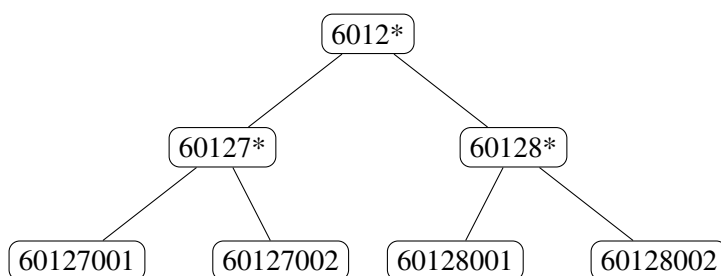
Como explanado na Seção 3.1, a análise dos dados é uma atividade fundamental, seja para melhorar a eficiência de um serviço, seja para ajudar na adoção de estratégias governamentais, ou para auxiliar na atividade econômica. Dessa forma estabelecer a confiança dos indivíduos e assim obter o consentimento para a utilização dos seus dados é um desejo de seus curadores. Portanto, é necessário garantir a proteção dos dados pessoais coletados. A anonimização é uma abordagem promissora para solucionar o problema de preservação de privacidade. Através de transformações do dados antes de sua publicação

**Tabela 3.1. Exemplos de identificadores explícitos e semi-identificadores em dados tabulados de indivíduos.**

Identificadores Explícitos		Semi-identificadores			
ID	Nome	Idade	Gênero	Endereço	CEP
1	Carla	24	Feminino	Av. I	60127002
2	João	21	Masculino	Av. K	60128001
3	Marcos	27	Masculino	Av. K	60128002
4	Ana	41	Feminino	Rua J	60127001

[21] procura-se impedir a exposição dos dados sensíveis dos indivíduos. Neste processo, um conjunto de dados  $D$  é transformado em um conjunto de dados  $D'$ , por meio de modificações sobre os dados. Técnicas de generalização, supressão e perturbação, são aplicadas sobre os dados para garantir esta transformação do conjunto de dados.

A generalização modifica os atributos semi-identificadores dos registros no conjunto de dados por valores mais gerais, aumentando a incerteza de um adversário associar um indivíduo a seus dados, ou em especial, a seus atributos sensíveis. Na abordagem mais comum de generalização, o valor de um atributo semi-identificador que se deseja proteger nos diferentes registros é substituído por um valor generalizado. Para exemplificar a aplicação da generalização sobre um conjunto de dados apresentamos a Figura 3.3. Ela ilustra a aplicação do processo de generalização sobre o atributo CEP (Código de Endereçamento Postal) presente nos registros da Tabela 3.1. Podemos observar que nas folhas da árvore têm-se os valores originais para o atributo de 4 registros. No segundo nível agrupam-se os CEPs cujos 5 primeiros dígitos correspondem, enquanto que no nível seguinte em direção ao topo da hierarquia são agrupados os CEPs cujos 4 primeiros dígitos correspondem.



**Figura 3.3. Exemplos de generalização do atributo semi-identificador CEP.**

Como resultado da generalização sobre o atributo CEP da Tabela 3.1 temos a Tabela 3.2. Todos os CEPs dos registros foram substituídos pelo CEP generalizado da raiz da árvore, tornando este atributo indistinguível dos demais registros. Desta forma, este processo dificulta a re-identificação de um dos indivíduos caso seja de conhecimento externo o CEP de seu endereço. Por outro lado, a generalização diminui a precisão da localização, afetando diretamente a utilidade dos dados. Dessa forma, uma análise sobre os dados de CEP da 3.2 pode não ser útil, por exemplo, para se identificar uma distribuição



de gênero dentro das micro-regiões de uma cidade no caso dessa generalização diminuir bastante a precisão das localizações.

**Tabela 3.2. Exemplo de semi-identificador anonimizado por generalização.**

Identificadores Explícitos		Semi-identificadores			
ID	Nome	Idade	Gênero	Endereço	CEP
1	Carla	24	Feminino	Av. I	6012****
2	João	21	Masculino	Av. K	6012****
3	Marcos	27	Masculino	Av. K	6012****
4	Ana	41	Feminino	Rua J	6012****

A supressão de dados ocorre pela remoção dos valores de atributos ou pela sua substituição. Neste último caso, um ou mais valores do conjunto de dados é substituído por algum valor especial que dificulte a tarefa de descoberta de semi-identificadores por adversários. Alguns dos principais tipos de supressão:

- **Supressão de registro:** a supressão de registro remove um registro inteiro do conjunto de dados, conseqüentemente nenhum valor de atributo é disponibilizado para uso [6, 27].
- **Supressão de valor:** a supressão de valor remove ou substitui todas as ocorrências de um valor de um atributo semi-identificador por um valor especial, como “\*”. Por exemplo, em uma tabela de funcionários de uma empresa, os valores de atributo salário abaixo de R\$ 30.000,00, podem ser removidos ou substituídos por “\*”, enquanto os demais valores não sofrem distorções [50, 51].
- **Supressão de células:** nessa técnica, apenas algumas instâncias de valores de um atributo são removidas ou substituídas por um valor especial, caracterizando uma *supressão local* [36]. Por exemplo, pode-se remover apenas metade dos valores de atributo salário abaixo de R\$ 30.000,00, em uma tabela de empregados. Assim, instâncias de salário podem conter valores abaixo ou acima de R\$ 30.000,00, além de valores suprimidos. Entretanto, essa estratégia pode levar a inconsistências em eventuais análises de dados.

Por último, mas, não menos importante, a perturbação substitui os valores dos atributos semi-identificadores originais por valores fictícios, de modo que informações estatísticas calculadas a partir dos dados originais não se diferenciem significativamente de informações estatísticas calculadas sobre os dados perturbados. As técnicas mais comuns de perturbação de dados são:

- **Adição de ruído:** essa técnica é aplicada comumente sobre atributos numéricos. O valor original “ $v$ ” de um atributo será substituído por “ $v + r$ ”, onde  $r$  corresponde ao ruído adicionado ao valor original. Para alcançar a proteção desejada, este ruído em geral é escolhido de forma aleatória seguindo alguma distribuição. Outra forma

de se aplicar a adição de ruído é através de um fator multiplicativo, onde o valor original é substituído por " $v \times r$ ". Os valores dos atributos são portanto, perturbados com um determinado nível de ruído, que pode ser adicionado ou multiplicado pelo valor original de cada atributo [47];

- **Permutação de dados:** nesta abordagem os valores de um mesmo atributo de dois registros diferentes são permutados. Isso mantém algumas características estatísticas dos dados, como frequência dos atributos e contagem [15]. Apesar desta técnica não alterar o domínio dos atributos, as possíveis permutações de valores diferentes podem levar a valores nos registros sem sentido, e com isso, informações equivocadas, tendo um impacto indesejável na utilidade dos dados;
- **Geração de dados sintéticos:** nesta técnica, um modelo estático é inicialmente gerado a partir do conjunto de dados e, após isso, são gerados dados sintéticos que seguem o modelo gerado [1]. Esses dados sintéticos são os que devem ser disponibilizados para o uso final. A vantagem desta técnica é que todas as propriedades estatísticas dos dados são mantidas. Entretanto, assim como na permutação de dados, nesta técnica pode-se gerar alguns valores sem sentido e que não são condizentes com o mundo real, embora as propriedades estatísticas mantenham-se fiéis às dos dados originais [33].

Esta transformação dos dados através da generalização, supressão ou perturbação protege os dados, permitindo, portanto, o seu compartilhamento a outras entidades. O nível de utilidade alcançado poderá garantir o uso das informações contidas nos dados sem que haja uma exposição dos indivíduos presentes no conjunto de dados. Todavia, assim como estas técnicas buscam garantir a privacidade dos indivíduos, pessoas mal intencionadas, comumente chamadas de atacantes, ou adversários, buscam se opor a esta proteção utilizando todos os recursos a sua disposição para retirar o máximo de informação dos registros. Esta informação a disposição do atacante utilizada no auxílio de inferências sobre o conjunto de dados é identificada por conhecimento prévio. Um exemplo de conhecimento prévio é o do adversário que trabalha no mesmo local da vítima, e conhece algumas informações privadas da vítima, tais como, endereço residencial, cargo na empresa, além de outras, permitindo a inferência de informações sensíveis, como localização, opção sexual, etc. Em se tratando de publicação de dados, o atacante pode ter acesso a outros conjuntos de dados previamente publicados, e assim, cruzar referências para descobrir novas informações sensíveis da vítima. Portanto, podemos concluir que o conhecimento do adversário é imensurável e imprevisível e deve ser levado em consideração nas soluções de preservação de privacidade, apesar de suas características.

Um adversário é capaz de violar a privacidade dos usuários por meio de diversos ataques que citaremos a seguir:

- **Ataque de Ligação ao Registro:** este ataque tem por objetivo re-identificar o registro de um usuário, cujas informações pertencem ao conjunto de dados publicado. Por exemplo, o atacante tem conhecimento do indivíduo João, mas não tem qualquer conhecimento sobre alguns de seus atributos, como seu endereço. Dessa forma, o atacante busca identificar que o registro de ID 2 da Tabela 3.1 pertence ao indivíduo João;

- **Ataque de Ligação ao Atributo:** o objetivo do adversário é ser capaz de inferir atributos sensíveis do usuário mesmo sem re-identificar seu registro, com base nos valores sensíveis relacionados ao grupo que o usuário pertence. Nesse tipo de ataque o adversário sabe que um usuário pertence a um certo grupo de registros, como por exemplo, que todos os usuário possuem CEP igual a 6012\*\*\* como na Tabela 3.2. Através desse conhecimento o adversário busca identificar informações sensíveis comuns a todos do grupo.
- **Ataque de Ligação à Tabela:** este tipo de ataque assume que o adversário sabe que o registro do usuário foi publicado. Neste ataque o intuito é inferir se a vítima está presente ou ausente nos dados publicados.
- **Ataque Probabilístico:** este ataque tem o foco de destacar como o adversário mudaria seu pensamento probabilístico sobre um usuário depois de ter acesso ao conjunto de dados disponível. Por exemplo, após analisar um conjunto de dados publicado, inferir a probabilidade de o usuário ser do sexo masculino.

### 3.3. Modelos de Privacidade Sintáticos

Os modelos de privacidade sintáticos exigem que o conjunto de dados anonimizados possuam uma forma definida que ajuda a reduzir o risco de quebra de privacidade [14]. Os modelos sintáticos, em geral, aplicam transformações nos registros por meio de técnicas de supressão e/ou generalização até que esta forma seja alcançada. Iremos apresentar alguns dos modelos de privacidade sintáticos mais utilizados em preservação de dados.

#### 3.3.1. *k*-anonimato

Modelo de privacidade mais conhecido no campo da anonimização de dados [46], o *k*-anonimato assegura que, para cada combinação de valores de semi-identificadores, existem pelo menos *k* registros no conjunto de dados, formando uma classe de equivalência. O *k*-anonimato atua sobre o princípio da indistinguibilidade, isto é, cada registro em um conjunto de dados *k*-anônimo é indistinguível de pelo menos outros *k* - 1 registros em relação ao conjunto de semi-identificadores. Desta forma, a probabilidade de se ligar qualquer indivíduo a um registro no conjunto de dados é de no máximo  $\frac{1}{k}$ .

O nível de privacidade é ajustado em função do parâmetro *k*, afetando diretamente o equilíbrio entre utilidade e privacidade. Assim, um valor de *k* grande implica em uma maior proteção dos dados, entretanto, diminui a utilidade dos mesmos, por ser necessário adicionar grande volume de ruído a fim de se alcançar classes de equivalência com pelo menos *k* registros. É importante ressaltar que não existem abordagens analíticas para determinar um valor ótimo para o parâmetro *k* [12], sendo este um problema NP-difícil [36]. Dessa forma, cabe aos *dataholders* esta complexa tarefa de se definir o nível desejado de privacidade que seja adequado para garantir um equilíbrio adequado entre privacidade e utilidade.

A Tabela 3.3 será tomada como base para aplicar alguns modelos de privacidade sintáticos, entre eles o *k*-anonimato. São atributos identificadores explícitos: Placa, Motorista e CPF. São atributos sensíveis: Tipo de Multa e Valor da Multa. Os demais atributos são semi-identificadores.

**Tabela 3.3. Dados sobre infrações de trânsito [31].**

	Placa	Motorista	CPF	Data de Nascimento	Data da Infração	Tipo de Multa	Valor da Multa (R\$)
1	UVW-1840	Gigi	223.512.956	14/03/1980	03/01/2013	1	170
2	AXO-2064	André Luis	523.512.511	04/03/1980	03/01/2013	2	250
3	AUG-1046	Juçara Silva	123.998.687	24/05/1980	03/01/2013	1	170
4	FBI-1001	Bruno Lima	230.320.523	20/04/1982	04/01/2013	1	170
5	ACO-6241	Abu Ali	221.320.876	20/05/1982	04/01/2013	2	250
6	ABA-5012	Pedro Ramires	210.329.890	13/05/1982	05/01/2013	2	250
7	HBV-2002	Eduardo Neto	538.687.045	15/05/1982	05/01/2013	1	170

A Tabela 3.4 foi gerada após a aplicação de técnicas de supressão nos identificadores explícitos e de generalização nos atributos semi-identificadores. Nesta tabela podemos perceber quatro classes de equivalência para os semi-identificadores: Classe A = “03/1980, 01/2013” nas linhas 1 e 2; Classe B = “05/1980, 01/2013” registro 3; Classe C = “04/1982, 01/2013” com o registro 4 e Classe D = “05/1982, 01/2013” nas linhas 5, 6 e 7. Observe, que após aplicar o processo de anonimização,  $k$ -anonimato ainda não foi alcançado para um  $k = 2$ , já que as classes B e C, não possuem uma quantidade mínima requerida de 2 registros, sendo, portanto, necessário, algum novo processo de transformação. Uma estratégia válida seria remover os registros 3 e 4, como podemos observar na Tabela 3.5, onde a tabela agora contém apenas 2 classes de equivalência, e alcançando o  $k$ -anonimato para  $k = 2$ .

**Tabela 3.4. Dados sobre infrações de trânsito anonimizados [31].**

	Placa	Motorista	CPF	Data de Nascimento	Data da Infração	Tipo de Multa	Valor da Multa (R\$)
1	*	*	*	03/1980	01/2013	1	170
2	*	*	*	03/1980	01/2013	2	250
3	*	*	*	05/1980	01/2013	1	170
4	*	*	*	04/1982	01/2013	1	170
5	*	*	*	05/1982	01/2013	2	250
6	*	*	*	05/1982	01/2013	2	250
7	*	*	*	05/1982	01/2013	1	170

**Tabela 3.5. Tabela no modelo 2-anonimato [31].**

	Placa	Motorista	CPF	Data de Nascimento	Data da Infração	Tipo de Multa	Valor da Multa (R\$)
1	*	*	*	03/1980	01/2013	1	170
2	*	*	*	03/1980	01/2013	2	250
3	*	*	*	*	*	*	*
4	*	*	*	*	*	*	*
5	*	*	*	05/1982	01/2013	2	250
6	*	*	*	05/1982	01/2013	2	250
7	*	*	*	05/1982	01/2013	1	170

### 3.3.2. *l*-diversidade

Assim como o *k*-anonimato, o *l*-diversidade age sobre o princípio da indistinguibilidade. Entretanto, apesar de sua popularidade, simplicidade e eficácia contra ataques de ligação ao registro, o *k*-anonimato não se mostra adequado contra ataques de ligação ao atributo, *i.e.*, ataques em que um adversário procura inferir informações sensíveis sobre registros mesmo sem identificá-los. Observando a Tabela 3.5 que garante o *k*-anonimato, para  $k = 2$ , podemos identificar pelo menos dois registros em cada uma das classes de equivalência. Entretanto, imagine o cenário hipotético, em que o atacante tem conhecimento de um outro conjunto de dados não anonimizado, em que é possível identificar um indivíduo cujo nome é Pedro que nasceu em maio de 1982, e sofreu uma infração em janeiro de 2013. Ele poderá inferir com uma probabilidade de  $\frac{2}{3}$  que a multa recebida por Pedro foi do tipo 2, e seu valor foi de 250 reais. Superior à  $\frac{1}{2}$ , desejada pelo modelo *k*-anonimato.

O *l*-diversidade busca prover proteção a ataques de ligação ao atributo, garantindo que para cada classe de equivalência, exista pelo menos *l* valores distintos para cada atributo sensível. Assim, o que se pretende é que um atacante, mesmo com conhecimento prévio sobre a classe de equivalência de um registro, não seja capaz de inferir o atributo sensível do mesmo com probabilidade maior que  $\frac{1}{l}$ . Na Tabela 3.6, a probabilidade de se identificar que o indivíduo tem asma, valor do atributo sensível "Doença", caso o atacante tenha conhecimento de que o CEP do indivíduo é 540040, é de 100%, superior a  $\frac{1}{4}$  exigido por um modelo 4-anonimato. Convertendo a Tabela 3.6 para o modelo 3-diversidade, não é preciso fazer nenhuma alteração nos registros da classe A (linhas 1 a 4), pois esta já possui no mínimo 3 valores distintos para o atributo sensível. Entretanto, a classe B (linhas 5 a 8) possui todos os valores de atributos sensíveis iguais. Uma solução simples seria suprimir os registros das linhas 5 a 8. Outra solução seria modificar os valores do atributo sensível destas linhas por valores diferentes que garantam a diversidade, conforme Tabela 3.7 que atende, portanto, o modelo 4-anonimato e 3-diversidade.

**Tabela 3.6. 4-anonimato [31].**

	Idade	CEP	Cidade	Doença
1	<70	560001	*	Sinusite
2	<70	560001	*	Gripe
3	<70	560001	*	Zika
4	<70	560001	*	Hérnia
5	<35	540040	*	Asma
6	<35	540040	*	Asma
7	<35	540040	*	Asma
8	<35	540040	*	Asma

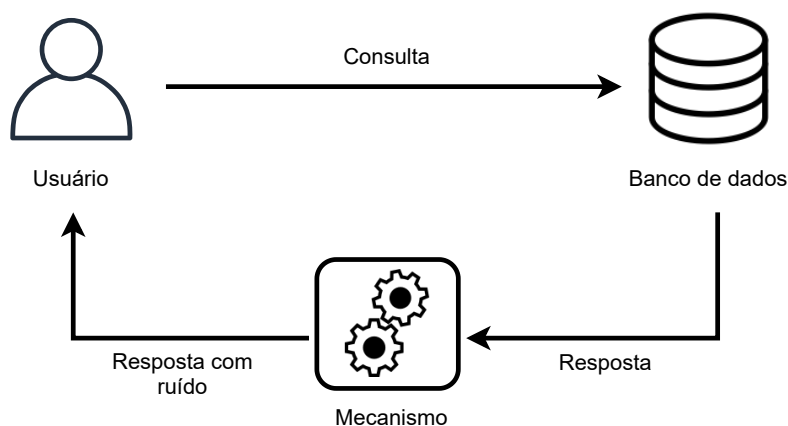
**Tabela 3.7. 4-anonimato e 3-diversidade [31].**

	Idade	CEP	Cidade	Doença
1	<70	560001	*	Sinusite
2	<70	560001	*	Gripe
3	<70	560001	*	Zika
4	<70	560001	*	Hérnia
5	<35	540040	*	Sinusite
6	<35	540040	*	Zika
7	<35	540040	*	Asma
8	<35	540040	*	Asma

Variações do *k*-anonimato e *l*-diversidade também foram propostas como uma extensão desses modelos com a finalidade de prover uma maior garantia de preservação de privacidade tanto contra ataques de ligação ao registro, como ao atributo [29, 49].

### 3.4. Privacidade Diferencial

Nos modelos de privacidade sintáticos uma violação de privacidade ocorre quando o atacante utilizando de quaisquer meios de conhecimento consegue re-identificar indivíduos



**Figura 3.4. Ambiente interativo no modelo de Privacidade Diferencial.**

dentro do conjunto de dados publicado em formato tabulado. Sob outra perspectiva, a Privacidade Diferencial investiga a ideia de publicar resultados de consultas, ao invés de dados tabulados, de tal forma que são adicionados ruídos a esses resultados. Dessa forma, a privacidade é garantida ao se anonimizar a resposta de consultas ao conjunto de dados. Assim, um atacante não será capaz de concluir algo com 100% de confiança. A sua principal convicção é de que as conclusões obtidas sobre um indivíduo são referentes aos dados de toda a tabela, e não apenas a um registro em particular que possa ser ligado a um indivíduo. Por esse motivo, o modelo de privacidade em questão propõe evitar ataques probabilísticos.

### 3.4.1. Conceitos Básicos

Proposto em [17], a Privacidade Diferencial fornece sólidas garantias de privacidade. Seu objetivo é solucionar o paradoxo entre garantir o aprendizado de informações úteis sobre a população de um conjunto de dados sem permitir obter qualquer informação específica sobre indivíduos desta população [19]. Um mecanismo diferencialmente privado garante a privacidade das consultas através da adição de um ruído aleatório controlado. Este modelo foi projetado em um ambiente interativo, onde os usuários submetem consultas a um conjunto de dados e este, por sua vez, responde por através do mecanismo. A Figura 3.4 apresenta este modelo iterativo, onde o mecanismo garantirá a privacidade ao introduzir “aleatoriedade” na geração do ruído, e portanto, protegendo os resultados das consultas realizadas sobre um conjunto de dados em seu formato original.

A Privacidade Diferencial assegura que qualquer sequência de resultados (isto é, resposta de consultas) é igualmente possível de acontecer independente da presença de qualquer indivíduo no conjunto de dados [19]. Assim, a adição ou remoção de um indivíduo não afetará consideravelmente o resultado de qualquer análise estatística realizada no conjunto de dados [13]. Portanto, o conhecimento adquirido por um atacante sobre qualquer indivíduo presente no conjunto de dados após realizar consultas a este conjunto não deve ser maior ao que ele já possuía antes de realizar qualquer consulta a este mesmo conjunto.

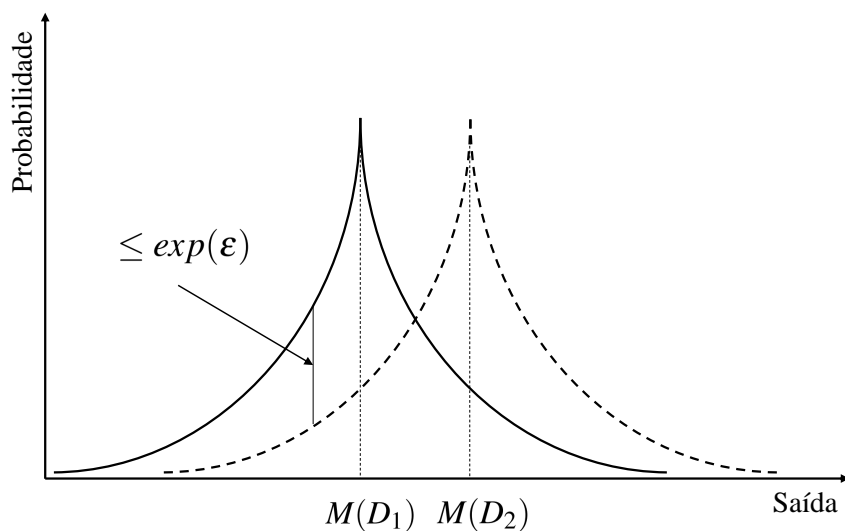
### 3.4.2. Definição Formal

Dado um algoritmo aleatório (mecanismo)  $M$ , este mecanismo garante  $\epsilon$ -Privacidade Diferencial se para todos os conjuntos de dados vizinhos  $D_1$  e  $D_2$  no conjunto de dados, que diferem de no máximo um elemento, e para todo  $S$  contido na variação de resultados de  $M$ , isto é, para todo  $S \subseteq \text{Range}(M)$ ,

$$Pr[M(D_1) \in S] \leq \exp(\epsilon) \times Pr[M(D_2) \in S],$$

onde  $Pr$  é a probabilidade dada a partir da “aleatoriedade” de  $M$ . Em outras palavras, a definição formal afirma que a diferença máxima entre as distribuições de probabilidade de uma consulta retornar o mesmo resultado para dois conjuntos de dados vizinhos é limitada pelo parâmetro  $\epsilon$ . Portanto, para qualquer par de entradas que diferem de apenas um registro, para cada saída, um adversário não será capaz de distinguir entre os conjuntos de dados  $D_1$  e  $D_2$  baseado apenas na resposta fornecida pelo mecanismo.

A Figura 3.5 mostra um exemplo de probabilidades de saída de um algoritmo  $M$ , nos conjuntos de dados vizinhos  $D_1$  e  $D_2$ , a partir de um valor de  $\epsilon$ . O algoritmo  $M$  fornece garantias de privacidade adicionando ruído aleatório no seu retorno, i.e.,  $M(D) = f(D) + \text{ruído}$ , onde  $f$  é a resposta de uma consulta realizada por um usuário. No exemplo apresentado, foi utilizado o mecanismo de Laplace, seguindo a distribuição de probabilidades de mesmo nome, o que resulta em uma distribuição com pico mais acentuado do que se utilizasse uma distribuição em função da normal. O conceito de mecanismo é definido na Seção 3.4.3.



**Figura 3.5. Probabilidades de saída de um algoritmo aleatório  $M$  sobre os conjuntos de dados vizinhos  $D_1$  e  $D_2$ .**

Dessa forma, se um indivíduo pertence a um conjunto de dados  $D$  onde análises estatísticas serão feitas através de um mecanismo que é  $\epsilon$ -Diferencialmente Privado, esse mecanismo irá garantir que a probabilidade de violação de privacidade não irá aumentar caso este indivíduo não pertencesse ao conjunto de dados. Dessa forma, podemos concluir que, como a Privacidade Diferencial é uma propriedade estatística restritiva a que o

mecanismo está sujeito, as garantias que ela oferece são altas, inclusive essas garantias não dependem de poder computacional ou informações que um atacante possa ter obtido.

O parâmetro  $\epsilon$  que controla o nível de ruído adicionado pelo mecanismo não possui uma correlação explícita com a privacidade dos indivíduos como em outras técnicas vistas. Esse parâmetro depende da consulta que está sendo feita e dos próprios dados que estão no conjunto de dados, devendo, portanto, ser escolhido por um especialista com o objetivo de se obter o melhor equilíbrio entre privacidade e utilidade das respostas. A literatura em geral concorda que o valor de  $\epsilon$  deva ser pequeno, como por exemplo 0,01, 0,1 ou até logaritmo natural  $\ln 2$  ou  $\ln 3$  [18]. Quanto menor o valor de  $\epsilon$ , maior a privacidade. Tradicionalmente, a escolha do valor de  $\epsilon$  se dá de forma empírica, portanto, para cada mecanismo, deve ser feita uma análise para escolher o parâmetro adequado utilizando métricas [39] para avaliar a precisão da resposta do mecanismo com diversos valores de  $\epsilon$  [26].

### 3.4.3. Mecanismo e Sensibilidade

Como dito anteriormente nesta seção, a Privacidade Diferencial é idealizada em um modelo interativo, onde o usuário submete consultas a uma base de dados  $D$ , e um determinado mecanismo fornece uma resposta  $\epsilon$ -Diferencialmente Privada. Porém, existem diversas formas de se atingir a Privacidade Diferencial através de um mecanismo. O objetivo das técnicas que utilizam esse modelo de privacidade é criar um mecanismo  $M$  que irá adicionar um ruído adequado para produzir uma resposta a uma consulta  $f$  feita pelo indivíduo, de forma que esse ruído seja independente do conjunto de dados  $D$ .

A quantidade de ruído necessária depende do tipo de consulta  $f$  aplicada sobre um conjunto de dados. Dessa forma precisamos definir o que é a sensibilidade de um conjunto de dados  $D$ . Antes disso, porém, precisamos definir de forma prática o que são conjuntos de dados vizinhos.

**Definição 1** *Dado um conjunto de dados  $D$ , todos os conjuntos de dados  $D_i$  decorrentes da remoção de um indivíduo  $i$  do conjunto de dados original  $D$  são definidos como vizinhos [10].*

Por exemplo, considere o conjunto de dados  $D$  na Tabela 3.8. Um possível conjunto de dados vizinhos pode ser obtido através da remoção do registro de  $ID = 6$ , resultando na Tabela 3.9.

A sensibilidade por sua vez procura quantificar a diferença que um usuário faz ao ser removido do conjunto de dados na resposta da função de consulta. Isso é fundamental para o cálculo adequado do ruído a ser adicionado pelo mecanismo, uma vez que quanto maior o valor de  $\Delta f$ , mais ruído terá de ser adicionado à resposta do mecanismo para mascarar a remoção de um indivíduo, de forma a assegurar a privacidade do mesmo [13].



**Tabela 3.8. Conjunto de dados  $D$ .**

	Idade	CEP	Cidade
1	35	560001	Fortaleza
2	40	560001	Aquiraz
3	55	560001	Messejana
4	21	560001	Eusébio
5	35	540040	Aracati
6	35	540040	Caucaia
7	45	540040	Sobral
8	22	540040	Fortaleza

**Tabela 3.9. Conjunto de Dados  $D'$ .**

	Idade	CEP	Cidade
1	35	560001	Fortaleza
2	40	560001	Aquiraz
3	55	560001	Messejana
4	21	560001	Eusébio
5	35	540040	Aracati
7	45	540040	Sobral
8	22	540040	Fortaleza

**Definição 2** *Seja  $D$  o domínio de todos os conjuntos de dados. Seja  $f$  uma função de consulta que mapeia conjuntos de dados a vetores de números reais. A sensibilidade global da função  $f$  é:*

$$\Delta f = \max_{x,y \in D} \| f(x) - f(y) \|_1$$

*para todo  $x, y$  diferindo de no máximo um elemento, ou seja, vizinhos [18].*

O mecanismo de Laplace é o algoritmo de adição de ruído controlado mais comum e simples para alcançar a Privacidade Diferencial. A adição de ruído é baseada na geração de uma variável aleatória da distribuição de Laplace com média  $\mu$  e escala  $b$  de forma que

$$Laplace_{\mu,b}(x) = \frac{1}{2b} \exp\left(-\frac{|x|}{b}\right).$$

Podemos então definir formalmente o mecanismo de Laplace.

**Definição 3** *Dada uma função de consulta  $f : D \rightarrow \mathfrak{R}$ , o mecanismo de Laplace  $M$ :*

$$M_f(D) = f(D) + Laplace(0, \Delta f / \epsilon)$$

*fornece  $\epsilon$ -Privacidade Diferencial. Onde  $Laplace(0, \Delta f / \epsilon)$  retorna uma variável aleatória da distribuição de Laplace com média zero e escala  $\Delta f / \epsilon$ .*

Utilizaremos a Tabela 3.10 para demonstrar a aplicação do mecanismo de Laplace de forma simplificada para melhor compreensão. A figura contém hipoteticamente um conjunto de dados da Receita Federal, que contém o número de imóveis que um determinado indivíduo declarou.

Suponha a consulta  $f$  que retorne o total de imóveis de todos os indivíduos da base de dados. Primeiramente, é necessário calcular a sensibilidade da função  $f$  sobre o conjunto de dados. Para isso calculamos  $f$  para cada conjunto de dados vizinho. A resposta real da consulta é 14. A Tabela 3.11 mostra os conjuntos de dados vizinhos gerados a partir da base original e suas respectivas respostas da consulta  $f$ .

**Tabela 3.10. Exemplo de conjunto de dados original contendo o número de imóveis de cada indivíduo.**

ID	Contribuinte	Nº de Imóveis
1	João	4
2	Bruno	2
3	Iago	7
4	Malu	1

**Tabela 3.11. Conjuntos de dados vizinhos gerados a partir da base original e suas respectivas respostas da consulta  $f$  (soma).**

ID	Contribuinte	Nº de Imóveis
2	Bruno	2
3	Iago	7
4	Malu	1

$$f(D_1) = 2 + 7 + 1 = 10$$

ID	Contribuinte	Nº de Imóveis
1	João	4
2	Bruno	2
4	Malu	1

$$f(D_3) = 4 + 2 + 1 = 7$$

ID	Contribuinte	Nº de Imóveis
1	João	4
3	Iago	7
4	Malu	1

$$f(D_2) = 4 + 7 + 1 = 12$$

ID	Contribuinte	Nº de Imóveis
1	João	4
2	Bruno	2
3	Iago	7

$$f(D_4) = 4 + 2 + 7 = 13$$

Portanto, a sensibilidade é dada pela variação máxima que a ausência de um indivíduo provoca no resultado da consulta. Essa variação é obtida quando da remoção do registro de  $ID = 4$ , cuja diferença máxima é de 7. Por fim, o ruído a ser adicionado para atender ao modelo de Privacidade Diferencial, utilizando o mecanismo de Laplace, deve ser igual a  $Laplace(0, \frac{7}{\epsilon})$ .

Como dito anteriormente, o parâmetro  $\epsilon$  é definido por um especialista, o detentor dos dados. A Tabela 3.12 apresenta cinco exemplos de ruído, respostas e probabilidade de ocorrência após a aplicação da Privacidade Diferencial sobre o conjunto de dados original da Tabela 3.10, considerando  $\epsilon = 1$ . Assim, após utilizar o mecanismo de Laplace, o valor de ruído de  $-4,58$  possui probabilidade de ocorrência de  $3,7\%$  sobre o valor original da consulta  $f$  (cuja soma do número de imóveis original é igual a 14), resultando em um valor anonimizado de  $9,42$  imóveis. De forma análoga, o valor de ruído de  $-0,15$  possui uma probabilidade um pouco maior de ocorrência ( $6,98\%$ ), caso a mesma consulta seja realizada nesse conjunto de dados, conforme mostra a Tabela 3.12.

Apesar do mecanismo de Laplace apresentar bons resultados, em se tratando de consultas do tipo categórica, que não permitem uma aproximação de seus valores categóricos, o mecanismo exponencial emergiu como uma alternativa eficaz satisfazendo  $\epsilon$ -privacidade diferencial. [35] propôs o mecanismo exponencial justamente para garantir a privacidade em situações onde se deseja a melhor resposta à consulta. Diferente do mecanismo de Laplace, que adiciona ruído aleatório à resposta, o mecanismo exponencial para uma consulta qualquer busca selecionar de forma aleatória uma resposta do conjunto de possíveis respostas. Portanto, este mecanismo é ideal em anonimizar consultas onde necessita medir a utilidade da resposta enquanto se preserva a privacidade diferencial.

**Tabela 3.12. Cinco possíveis valores de ruído, resposta e probabilidade de ocorrência após a aplicação da Privacidade Diferencial.**

<i>Ruído</i>	$f(D) + \text{ruído}$	$Pr(f(D) + \text{ruído})\%$
-4,58	9,42	3,70
-0,15	13,85	6,98
12,15	26,15	1,25
-6,43	7,57	2,85
2,89	16,89	4,72

**Tabela 3.13. Tabela de número de imóveis dos contribuintes e a pontuação retornada pela função de utilidade.**

<i>Contribuinte</i>	<i>Nº de Imóveis</i>	<i>Pontuação</i>
João	4	16
Bruno	2	10
Iago	7	28
Malu	1	4

A distribuição de probabilidades de respostas a uma consulta irá considerar uma função de utilidade que irá mapear todos os possíveis conjuntos de dados  $D$  e todas as possíveis respostas  $O$  em uma pontuação de utilidade  $u : (D \times O) \rightarrow \mathfrak{R}$  para um  $\epsilon$ .  $\Delta u$  é a sensibilidade da consulta em termos da função de utilidade  $u$ :

$$\Delta u = \max_{o \in O} \max_{D_1, D_2: \|D_1 - D_2\|_1 \leq 1} |u(D_1, o) - u(D_2, o)|,$$

para todo  $D_1, D_2$  diferindo de no máximo 1 elemento.

**Definição 4** Para qualquer função de utilidade  $u : (D \times O) \rightarrow \mathfrak{R}$ , e um orçamento de privacidade  $\epsilon$ , o mecanismo exponencial  $M_u^\epsilon(D)$  produz o como resposta com uma probabilidade proporcional a  $\exp(\frac{\epsilon u(D, o)}{2\Delta u})$ , onde  $\Delta u$  é a sensibilidade da função de utilidade:

$$Pr[M_u^\epsilon(D) = o] = \frac{\exp(\frac{\epsilon u(D, o)}{2\Delta u})}{\sum_{o' \in O} \exp(\frac{\epsilon u(D, o')}{2\Delta u})}$$

Utilizando os dados da Tabela 3.10 podemos aplicar o mecanismo exponencial para responder uma consulta que busque o nome do indivíduo que possui o maior número de imóveis. Para aplicar o mecanismo, primeiramente precisamos definir uma função de utilidade que indique o quão adequado é cada possível resposta. Dessa forma, nossa função de utilidade precisa estar relacionada ao número de imóveis. Suponha que nossa função de utilidade seja o nome do contribuinte vezes o número de imóveis que ele possui, ou seja,  $u(D, o) = \text{len}(o.\text{contribuinte}) \times o.\text{imoveis}$ , onde  $D$  é o conjunto de dados e  $o$  é um registro em  $D$ .

A Tabela 3.13 apresenta as possíveis respostas para a consulta e a pontuação obtida pela função de utilidade  $u$ . Observe que a resposta correta para a consulta é *Iago*, justamente a resposta com a maior pontuação retornada pela função de utilidade. O próximo passo é medir a sensibilidade de  $f$ . A Tabela 3.14 apresenta o impacto na função de utilidade ao se remover um elemento do conjunto de dados original. Como  $\Delta_u$  é a máxima diferença na função de utilidade em conjuntos de dados vizinhos, a sensibilidade da função para os conjuntos de dados vizinhos da Tabela 3.14 é de  $\Delta_u = 28$ , quando se remove o contribuinte *Iago* do conjunto de dados.

**Tabela 3.14.**  $\Delta_u = 28$ , quando se remove o contribuinte Iago do conjunto de dados.

ID	Contr.	Nº. Imóveis	Pont.
2	Bruno	2	10
3	Iago	7	28
4	Malu	1	4

$$|u(D, o) - u(D_1, o)| = 16$$

ID	Contr.	Nº. Imóveis	Pont.
1	João	4	16
2	Bruno	2	10
4	Malu	1	4

$$|u(D, o) - u(D_3, o)| = 28$$

ID	Contr.	Nº. Imóveis	Pont.
1	João	4	16
3	Iago	7	28
4	Malu	1	4

$$|u(D, o) - u(D_2, o)| = 10$$

ID	Contr.	Nº. Imóveis	Pont.
1	João	4	16
2	Bruno	2	10
3	Iago	7	28

$$|u(D, o) - u(D_4, o)| = 4$$

Uma vez calculada a sensibilidade da consulta em termos da função de utilidade, a Tabela 3.15 contém as probabilidades de cada uma das possíveis respostas serem retornadas pelo mecanismo exponencial, onde a probabilidade da resposta correta ser retornada pelo mecanismo é de 31,45 %.

### 3.4.4. Aplicações

Leis de proteção a dados pessoais quanto à coleta e uso têm sido cada vez mais frequentes em diversos países, tais quais o Brasil. Dessa forma, muitas organizações detentoras de dados têm desenvolvido aplicações para garantir a privacidade sobre os dados coletados de seus clientes. A Microsoft, por exemplo, desenvolveu o PINQ [34], uma plataforma de análise de dados projetada para fornecer garantias de privacidade para os registros contidos em suas bases de dados. Agindo como uma camada intermediária entre a base de dados e o dono dos dados, o PINQ utiliza um sistema de consultas próprio, denominado LINQ, permitindo-o realizar consultas sem comprometer a privacidade dos indivíduos pertencentes a base.

**Tabela 3.15.** Cinco possíveis respostas e probabilidade de ocorrência após a aplicação da Privacidade Diferencial através do Mecanismo Exponencial.

Resposta	$Pr(f(D))\%$
João	25,43
Bruno	22,75
Iago	31,35
Malu	20,45

A Google através dos seus diversos aplicativos coleta uma quantidade enorme de dados de seus usuários, utilizando para diversos fins, incluindo melhorar a qualidade de seus serviços. Visando proteger estas informações ela lançou uma ferramenta denominada *Rappor* (Randomized Aggregatable Privacy Preserving Ordinal Responses). Essa ferramenta utiliza a perturbação na coleta de dados, mantendo as informações estatísticas necessárias para realizar suas análises e preservando a privacidade dos usuários que utilizam seu navegador.

A Apple em 2016 anunciou que aplica nos dispositivos de sua marca a privacidade diferencial na coleta de dados dos usuários. De acordo com a companhia, os dados são coletados de maneira privada, a fim de proteger a privacidade do usuário mas permite fazer análise de dados agregados e implementar melhorias no serviço. Dessa forma, recursos como Siri e até o QuickType poderão prever melhor as palavras que, por exemplo, um determinado conjunto de usuários mais utilizam.

Nesta subseção apresentamos vários modelos e técnicas utilizados por grandes organizações para garantir a privacidade. Além delas, é possível identificar muitos exemplos de aplicações no mundo real que também utilizam as técnicas vistas neste minicurso, como [37], [8] e [32].

### 3.5. Serviços de Localização

Como já destacado na Seção 3.1, o desenvolvimento dos dispositivos móveis equipados com GPS, juntamente com a disponibilidade de redes sem fio, tem contribuído com um aumento significativo da popularidade dos Serviços de localização [3]. O serviço prestado utiliza da localização do usuário, muitas vezes em tempo real, para prover alguma valia ao solicitante do serviço. São alguns exemplos comuns de serviços de localização:

- **Navegação:** provê ao usuário direcionamentos a um ponto de interesse geograficamente localizado. Os dados de localização do usuário em tempo real são utilizados como referência às instruções de direção. São algumas aplicações: Google Maps e Waze.
- **Aplicações de tempo (clima):** estes serviços apresentam condições climáticas em tempo real, bem como previsões. A localização da solicitação é usada para obter informações relevantes sobre o clima local.
- **Jogos:** utilizam a localização do usuário no contexto do ambiente virtual do jogo. Os mais recentes usam tecnologia de realidade aumentada, onde a movimentação do usuário em tempo real se reflete no jogo. Exemplo desse tipo de jogo é Pokemon GO.
- **Serviços de Recomendação:** estes serviços utilizam da localização do usuário para enviar recomendações de locais de interesse próximos. São exemplos: Foursquare e Yelp.

Nesta seção, iremos destacar a arquitetura dos serviços de localização, a natureza dos dados de localização, e o porquê destes dados serem de grande valia na análise de dados, ao mesmo tempo que são objeto de ataques à privacidade.

### 3.5.1. Arquitetura

Em um sistema tradicional de serviço de localização as informações de localização são obtidas por meio de sistemas de posicionamento global (GPS), presente na maioria dos dispositivos móveis da atualidade. O usuário, através de seu dispositivo móvel, realiza uma requisição ao provedor do serviço tendo como referência sua localização, como por exemplo, a previsão do tempo. O serviço então atenderá esta requisição.

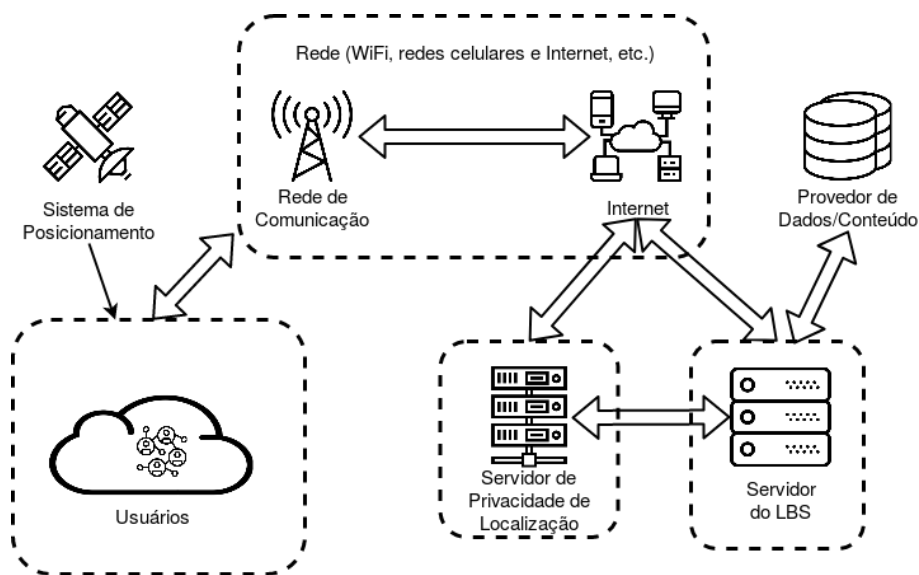
A Figura 3.6 ilustra um típico serviço baseado em localização com preservação de privacidade [30]. São alguns componentes básicos de um LBS:

- **GPS:** permite determinar a localização dos objetos envolvidos, *i.e.*, usuários, ou outra entidade qualquer. O GPS é o mais popular sistema de posicionamento. Ele é um mecanismo de posicionamento por satélite que fornece a um aparelho receptor a sua posição.
- **Usuários:** são participantes que irão usufruir do serviço baseado em localização prestado. Através de dispositivos como *smartphones*, *notebooks*, *wearables*, os usuários se conectam ao meio de comunicação e enviam requisições ao provedor do serviço
- **Rede de comunicação:** é o meio através do qual acontece o tráfego de informações entre os participantes. Normalmente o meio utilizado é a rede de banda larga móvel, como a 4G.
- **Servidor do LBS:** é o responsável por receber as requisições dos usuários e prestar o serviço baseado em localização de acordo com sua natureza, seja para encontrar uma localização, seja para auxiliar na navegação do usuário, ou um outro tipo de serviço qualquer que utiliza a informação de localização enviada na requisição.
- **Provedor de Conteúdo/Dados:** provedores de Conteúdo/Dados fornecem dados e conteúdo ao servidor LBS. Alguns provedores de LBS possuem seus próprios dados e conteúdo, enquanto outros usam um terceiro para fornecer esse serviço.
- **Servidor de Privacidade:** o servidor de privacidade de localização executa os algoritmos de preservação de privacidade, como anonimização e criptografia e pode ser de propriedade e operado pelo provedor LBS ou por terceiros.

### 3.5.2. Dados de localização

Os dados de localização, em geral, possuem informações agregadas. Principalmente quando estamos falando de Pontos de Interesse (*POI*), tais como restaurantes, hospitais, fábricas, dentre outros. Um hospital por exemplo, está associado a doenças, horário de atendimento, exames. Uma fábrica está associada a trabalhadores, produtos, horário de expediente. Esta informação agregada garante a existência de uma relação entre as localizações, podendo ser medida através da correlação entre elas [53].

A informação de localização é uma componente chave em uma requisição a um serviço de localização. As informações de localização são constituídas de três partes principais: identidade, tempo e posição [30], Figura 3.7.



**Figura 3.6. Modelo de sistema de serviços baseados em localização**



**Figura 3.7. Três atributos da informação de localização**

A *identidade* está relacionada ao usuário do serviço de localização. Pode ser um endereço de e-mail, nome ou qualquer outra informação que torne um indivíduo distinguível dos outros. Esta identidade pode ser: (i) consistente, que é aquela requisitada obrigatoriamente para o acesso a um serviço, como um nome de usuário; (ii) inconsistente, através do uso de pseudônimos; (iii) anônima, onde há a ausência de uma identificação.

O *tempo* é referente ao momento ao qual as localizações estão associadas. Em geral, os serviços de localização associam as localizações a marcos de tempo. Esta informação temporal pode ser acumulada ou corrente. As aplicações em tempo acumulado não publicam as informações de localização em tempo real, mas em um tempo posterior ao atual. Um exemplo destas aplicações é o sistema de rastreamento do *Fitbit*, que coleta as informações percorrida pelos usuários, mas só publica a trajetória computada depois de encerrada a atividade [30]. As aplicações em tempo real publicam as informações de localizações associadas ao marco de tempo atual, de forma imediata.

A *informação espacial (posição)* é a forma de se identificar no plano espacial a localização do usuário. Em geral, a posição é determinada em função das coordenadas

geo-espaciais do indivíduo, através de sua latitude e longitude, ou através de alguma outra representação desta localização. Ela pode ser única, também chamada de simples, quando as localizações, em geral pela natureza do serviço, não são correlacionadas entre si, ou podem formar uma trajetória no caso em que são fortemente correlacionadas, o que acaba por gerar maiores riscos de exposição.

Quanto à representação, as localizações podem ser classificadas em diretas ou indiretas. As localizações diretas representam a posição precisa da localização, através de suas coordenadas de latitude e longitude. Os serviços de localização tradicionais usam localizações diretas. As localizações indiretas são aquelas estabelecidas com base na proximidade física, substituindo a localização exata pelo POI mais próximo, dotado de informação complementar sobre o mesmo, como o nome do estabelecimento, horário de funcionamento, endereço, entre outras.

Como exposto na Seção 3.1, a análise da informação de localização pode ser extremamente útil, inclusive na otimização de performance e de qualidade do mesmo. Entretanto, essa exposição também traz riscos à privacidade, permitindo a inferência de dados sensíveis do usuário, como por exemplo, seu estado de saúde, suas crenças religiosas, posicionamentos políticos, dentre outros. A privacidade de localização pode ser alcançada através da proteção dos três atributos que formam a informação de localização de uma pessoa, identidade, posição e tempo. Entretanto, é importante destacar que a garantia de privacidade de localização de indivíduos não é absoluta, e não é esperado que seja. Uma privacidade de localização sem qualquer vazamento de informação, inviabilizaria completamente um serviço. Por exemplo, se o indivíduo quer saber a previsão do tempo de sua cidade. É esperado que a localização enviada seja da cidade em que ele quer saber a previsão. Caso contrário, o serviço pode não atingir um nível adequado de qualidade. Desta forma, pode-se identificar dois requisitos principais da privacidade de localização dos indivíduos: a expectativa de privacidade dos indivíduos de "circunstâncias normais", ou seja, o que o indivíduo espera em termos de exposição da sua localização, e a maneira como as informações são coletadas e usadas. A expectativa de privacidade de uma pessoa pode mudar com o tempo, assim como a forma como as informações de localização são coletadas e usadas também mudam. Logo, para avaliar a privacidade de localização do indivíduo, seus principais requisitos devem ser definidos do ponto de vista dos usuários.

### **3.5.3. Exposição Temporal**

A exposição temporal em serviços de localização constitui igualmente informação sensível dos usuários desses serviços. Informações referentes ao momento no tempo em que certas requisições foram realizadas, ou mais especificamente, em que momento no tempo se esteve em determinadas localizações, pode contribuir para a identificação de indivíduos, bem como, inferir informações sensíveis [52]. No contexto de trajetórias, além da exposição temporal, é possível identificar uma correlação forte entre as localizações que a compõem, permitindo a inferência de características semi-identificadoras [41].

Esta exposição é potencializada em função do período de observação. Quanto maior o período, maior é a exposição temporal gerada. O trabalho [40] define como calcular a probabilidade de se identificar a localização do usuário em curto período de observação e em longo período.



### 3.5.3.1. Curto período de observação Temporal

A localização do usuário é inferida em função de apenas um momento de tempo da localização ofuscada. Para isso, é necessário o conhecimento *a priori* sobre a localização ofuscada. Este conhecimento pode ser calculado de diversas formas. Uma forma bastante utilizada é o histórico do usuário de acessos ao serviço de localização. Desta forma, sendo  $L$  o conjunto de todas as localizações possíveis, a distribuição de probabilidade a posteriori  $Pr(l|l')$ , para  $l, l' \in L$ , pode ser calculada por:

$$Pr(l, l') = \frac{f(l)M(l'|l)}{\sum_{l \in L} f(l)M(l'|l)},$$

onde  $l'$  é a localização ofuscada,  $f(l)$  denota o conhecimento a priori sobre a localização atual e  $M(l'|l)$  representa a distribuição do mecanismo de ofuscação. Calculada a distribuição a posteriori, a localização atual  $l^*$  pode ser estimada de duas formas:

$$l^* = \arg \max_{l \in L} Pr(l|l'),$$
$$l^* = \arg \min_{l^* \in L} \sum_{l \in L} Pr(l|l') deuc(l^*, l).$$

### 3.5.3.2. Longo período de observação Temporal

As localizações ofuscadas reportadas pelo usuário são observadas por um período específico de tempo. Considerando uma situação em que o usuário aplica a ofuscação sobre a mesma localização várias vezes neste período de tempo, por exemplo, seu local de trabalho, o conjunto  $O = \{o_1, \dots, o_n\}$  é o conjunto das frequências com que as localizações são reportadas neste período. A estimativa da localização atual  $l^*$  pode ser dada por:

$$l^* = \arg \max_{l \in L} o_l,$$

### 3.5.4. Tipos e Métodos de Ataques

Um atacante, também chamado de adversário, é qualquer entidade que possa ter acesso aos dados de localização de um ou de vários indivíduos visando se beneficiar [30]. Em geral, modelos de privacidade consideram o provedor de serviço, ou o curador dos dados, honesto mas curioso [23]. Desta forma, um adversário pode ser desde o próprio provedor do serviço de localização, ou até mesmo um cientista de dados que tenha acesso a uma publicação dos dados [42]. O conhecimento adversário é justamente essa informação prévia acessível ao atacante, e que pode ser usada por exemplo para identificar um usuário, uma localização, ou uma sequência de lugares de um objeto móvel [41].

Os dados de localização, como explicado na Seção 3.5.2, por serem carregados de informação, permitem que informações de contexto sejam adicionadas a este conhecimento adversário gerando potenciais riscos de violação de privacidade. São exemplos de conhecimento de contexto: o número de usuários em uma área em uma determinada hora do dia; a relação entre diferentes usuários; as restrições de localização de uma determinada área, como rede de ruas, área de preservação; a distribuição e a probabilidade estatística associada às localizações.

Em particular, assume-se que o atacante possui qualquer base de dados que contém conhecimentos adicionais sobre a semântica das informações de localização dos usuários. Além disso, o provedor do LBS pode identificar que o usuário está utilizando alguma técnica de preservação de privacidade de localização a fim de garantir a utilização do serviço sem expor sua localização real. Ataques a privacidade de localização podem ser aplicados em função da identidade, ou em função da localização do usuário [30].

#### **3.5.4.1. Ataque de Identidade**

Os ataques de identidade procuram cruzar conhecimentos adversários de diversas fontes a fim de determinar a identidade do alvo. São alguns exemplos deste tipo de ataque:

- **Ataque de identificação pessoal:** através do conhecimento prévio pessoal de um indivíduo, busca-se identificar o indivíduo dentro do conjunto de dados, a fim de se obter toda a informação a ele associada no conjunto de dados. Considere o exemplo: o atacante tem o conhecimento sobre o endereço residencial de um indivíduo. Através dele, mesmo em um conjunto de dados anonimizado, se o atributo endereço não tiver sido protegido, o atacante poderá identificar o dono do registro em função do endereço residencial exposto.
- **Ataque de presença agregada:** identificar a identidade com base na relação entre dois indivíduos ou através de uma propriedade agregadora, por exemplo pessoas agrupadas próximas a um evento, uma estação de Pokemon Go, ou uma loja com ofertas, dentre outros eventos.

#### **3.5.4.2. Ataque de Localização**

Os ataques de localização consistem em identificar as informações espaciais e temporais referentes a um indivíduo. São alguns exemplos de ataques de localização:

- **Ataque a localizações sensíveis:** procura identificar localizações importantes, como residência ou local de trabalho.
- **Ataque de revelação de presença ou ausência:** determina se um usuário está presente ou ausente em determinadas localizações em um determinado horário do dia.
- **Ataque de rastreamento:** identifica uma sequência de eventos para rastrear um usuário.

#### **3.5.4.3. Métodos de Ataque**

Os métodos de ataque dizem respeito à forma como o ataque é realizado. São alguns destes métodos:

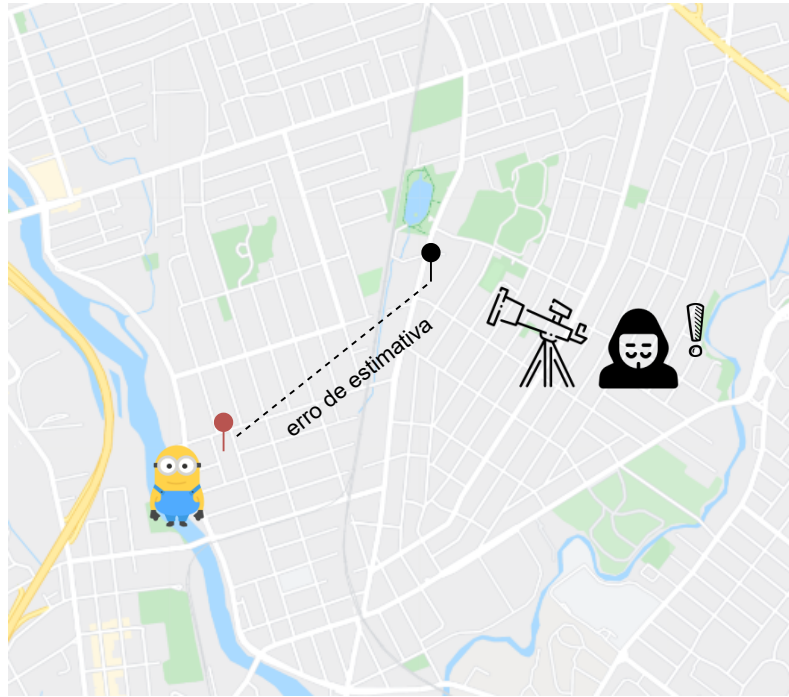
- **Ataques de vinculação de contexto:** é a forma mais comum em ataques de localização. O conhecimento de contexto é combinado com a informação de localização obtida para se chegar à localização precisa da vítima em um ataque de localização. Por exemplo, um indivíduo ao realizar um *check-in* em um hospital, preenche seus dados informando seu endereço residencial. Se um atacante tiver conhecimento do endereço residencial do indivíduo, ele poderá usá-lo para identificar este indivíduo na lista de *check-in* do hospital.
- **Ataques probabilísticos:** este tipo de ataque é baseado na coleta de informações estatísticas sobre o ambiente [44]. Pode ser aplicado tanto para ataques de identidade, como de localização. Sendo assim, a localização do usuário pode ser inferida em razão da probabilidade de o usuário estar em uma determinada localização em um horário preciso.
- **Ataque de conluio de usuários maliciosos:** é realizado por usuários que usam o mesmo provedor de serviços baseado em localização, que colidem para realizar vários ataques. Por exemplo, usuários em conluio utilizam sua posição para obter, do serviço, a distância da vítima, e baseado nisso calculam a exata localização da vítima.

### 3.6. Privacidade de Dados de Localização

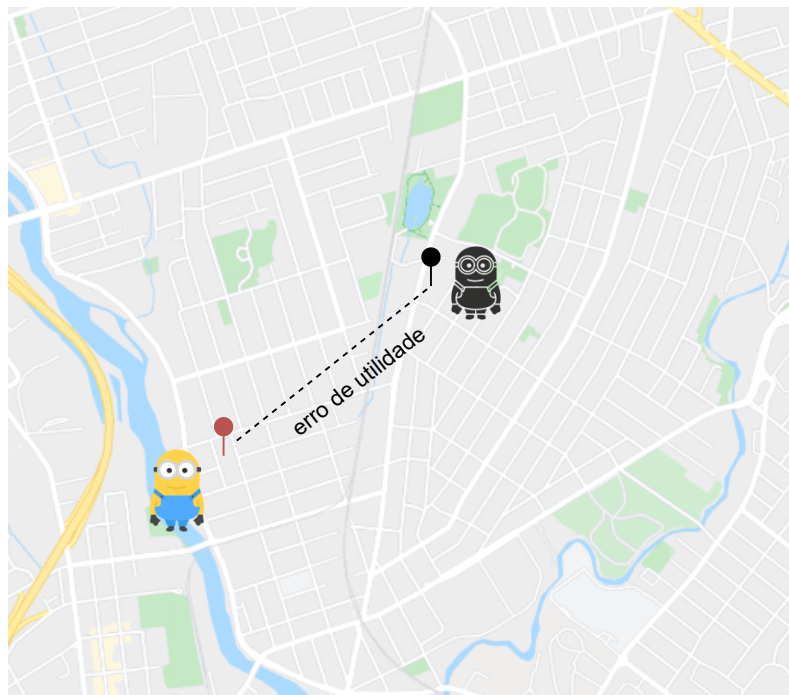
A privacidade de dados de localização busca proteger a privacidade dos usuários no uso dos serviços de localização. O que se pretende é entregar uma margem de segurança para os indivíduos em serviços de localização. Esta margem de segurança pode ser medida em função do erro de estimativa de um adversário, uma métrica muito utilizada para medir o nível de privacidade em termos da habilidade de um atacante em estimar a localização real de um usuário [2]. A Figura 3.8 exemplifica o erro de estimativa como sendo a distância euclidiana entre a localização real de um usuário e a localização estimada pelo adversário em preto.

Outro ponto importante em serviços de localização se refere a utilidade da informação de localização reportada. Quanto mais preciso for a informação reportada, maior é sua utilidade, e conseqüentemente, melhor é a qualidade do serviço. Esta utilidade pode ser medida em função do erro de utilidade. A Figura 3.9 exemplifica o erro de utilidade como sendo a distância euclidiana entre a localização reportada e a real localização do usuário.

O serviço de localização tem um impacto direto no nível de privacidade e utilidade desejado, portanto, é importante medir o nível de privacidade e utilidade em função destas métricas citadas. Por exemplo, um serviço de requisição de Pontos de Interesse, exige uma precisão na localização reportada bem maior do que um serviço de meteorologia, que certamente possui um relaxamento na precisão da localização na casa de quilômetros. Dessa forma, diversas técnicas de privacidade foram propostas a fim de preservar a privacidade dos indivíduos em serviços de localização. Esta seção apresenta uma visão geral das técnicas mais utilizadas para a preservação de dados de localização.



**Figura 3.8. Erro de estimativa de um adversário.**



**Figura 3.9. Erro de utilidade.**

### 3.6.1. *k*-anonimato de localizações

Conforme discutido na Seção 3.3.1, o *k*-anonimato é um modelo de privacidade proposto por Sweeney et al. [46] com o objetivo de prevenir ataques de ligação ao registro. Através de técnicas de generalização ou supressão de dados, busca dificultar a re-identificação do indivíduo apagando os valores dos semi-identificadores ou diminuindo a sua precisão de tal sorte que o indivíduo tenha suas propriedades assemelhadas às propriedades de outros indivíduos. O *k*-anonimato busca garantir que para cada combinação de *k* atributos semi-identificadores, existem pelo menos *k* registros distintos no conjunto de dados publicado, formando uma classe de equivalência.

No contexto de privacidade de localização, um sujeito é tido *k*-anônimo se sua localização é indistinguível da localização de outros  $k - 1$  usuários [22]. Portanto, a probabilidade de um usuário malicioso violar a privacidade de um indivíduo através de um ataque não será maior do que  $\frac{1}{k}$ .

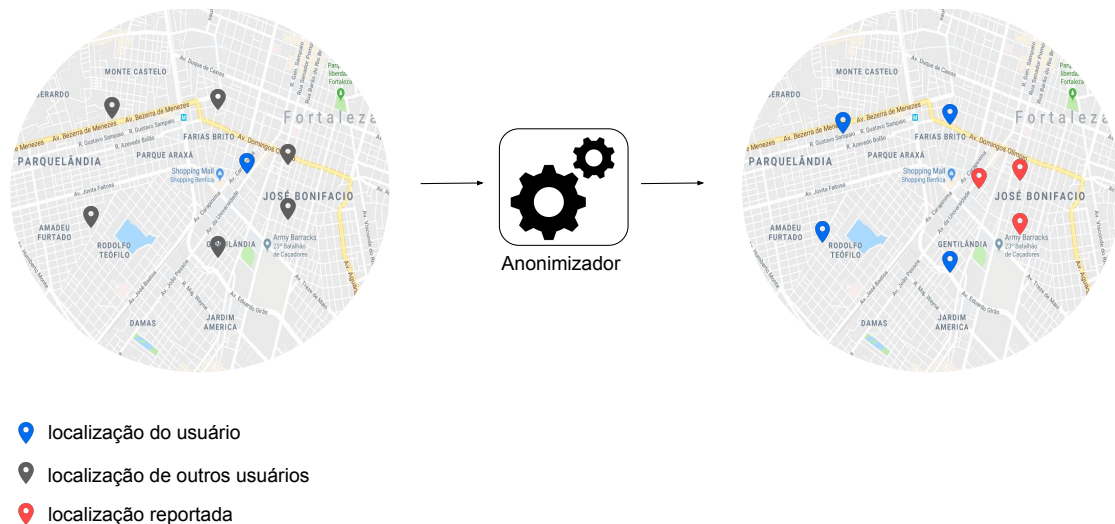
O parâmetro *k* do modelo define o nível de privacidade requisitada. Assim, quanto maior o valor de *k*, maior será a privacidade dos dados e, conseqüentemente, menor é a probabilidade de se identificar o indivíduo. Entretanto, um valor alto de *k* tem um impacto no desempenho do algoritmo de anonimização, afetando a qualidade do serviço. Além disso, o processo de aproximar coordenadas geográficas pode deslocar virtualmente o solicitante para uma localização semelhante a outros *k* indivíduos, diminuindo a precisão do serviço. Desta forma, encontrar um equilíbrio entre privacidade e qualidade se faz ainda mais importante no contexto de dados de localização. Contudo, encontrar um valor ótimo para o parâmetro *k* é um problema NP-difícil, como citado na Seção 3.3.1. Desta forma, os responsáveis pela anonimização devem especificar o grau de privacidade desejada em função desse parâmetro.

O modelo tradicional da aplicação do *k*-anonimato em dados de localização requer a participação de uma entidade confiável responsável pelo processo de anonimização, o anonimizador. Dessa forma, quando um usuário necessita realizar uma requisição, enviando sua localização, o anonimizador calcula um conjunto de *k* usuários e reporta uma área de ofuscação contendo *k* localizações, incluindo a localização do usuário.

A Figura 3.10 ilustra uma abordagem da utilização dessa técnica, para um  $k = 3$ , onde o usuário solicitante deseja enviar uma requisição ao serviço de localização informando sua posição em azul. Aplicando o *k*-anonimato para  $k = 3$ , um terceiro confiável responsável por anonimizar a sua localização, agrupa a localização do usuário a outras  $k - 1$  localizações, enviando uma requisição ao LBS contendo *k* localizações no total. Este, por sua vez, irá responder a requisição em função de cada uma das localizações enviadas. Como o terceiro confiável tem conhecimento das localizações dos usuários, ele irá filtrar a resposta referente a localização real presente na requisição e enviar o resultado para o usuário solicitante. Assim, para um atacante, a localização do usuário pode ser qualquer uma das *k* localizações que fazem parte da requisição, garantindo que a probabilidade de se identificar a localização do usuário não seja superior a  $\frac{1}{k}$ .

Uma desvantagem desse modelo é justamente a necessidade do anonimizador para realizar o processo de agrupamento de usuários. Dessa forma, se o atacante estiver agindo entre o anonimizador e o serviço, interceptando as requisições enviadas, estas estão ano-

nimizadas segundo o modelo  $k$ -anonimato. Entretanto, se o atacante estiver agindo entre o usuário e o anonimizador, a localização do usuário se encontra desprotegida.



**Figura 3.10. Processo de anonimização utilizando  $k$ -anonimato.**

### 3.6.2. Zonas de mixagem

As zonas de mixagem procuram proteger a privacidade do usuário de ataques de ligação, ao evitar que seja possível vincular a identidade dos usuários à sua localização. Entretanto, diferentemente do  $k$ -anonimato, a zona de mixagem pode ser aplicada sem qualquer informação de identidade do usuário. O conceito de zona de mixagem foi proposto por Beresford et al. [7]. Ele propõe um *framework*, onde os usuários utilizam pseudo-ids que são modificados constantemente garantindo que estes não sejam identificados no uso de serviços de localização. Sendo assim, a real identidade do usuário é protegida através do uso de pseudônimos.

As zonas de mixagens são definidas como áreas circulares de raio  $r$ , onde nenhum dos usuários dentro da zona de mixagem possui qualquer registro de chamada de retorno ao serviço, ou seja, estão anônimos em relação ao serviço. Esta técnica procura garantir a indistinguibilidade dos usuários no uso do serviço dentro da zona, através do uso de pseudo-ids e a ausência das informações de localização de seu usuário. Assim, como o  $k$ -anonimato, a técnica de zona de mixagem em sua forma tradicional exige a figura de um terceiro confiável, responsável pela anonimização. Este terceiro confiável tem o papel de gerenciar os pseudo-ids dos usuários, garantindo que sempre que um usuário entre na zona de mixagem, ele possua um pseudo-id único que não tenha sido registrado por nenhuma aplicação. Dessa forma, como o serviço não recebe qualquer informação de localização dos usuários, sua identidade está "misturada" com a dos outros usuários dentro da zona. Qualquer informação de localização presente na zona de mixagem diz respeito à zona de uma forma geral, como por exemplo o seu ponto central.

A eficácia desta técnica depende de ajustar um tamanho adequado destas zonas de mixagem em função de seu raio  $r$  e também em função da quantidade mínima de indivíduos presentes. Se o valor de  $r$  for muito grande, impossibilitando que um indivíduo

percorra a distância que ela cobre e ingresse em outra zona em um único período de tempo, é possível que esta informação seja usada para identificar usuários que saem e entram em outras zonas, tornando incompleta a função de mixagem. Além disso, se a quantidade de usuários presentes em uma zona de mixagem for muito pequena, novamente ela não cumpre seu propósito de proteger a privacidade dos indivíduos presentes dentro da zona.

A Figura 3.11 ilustra a aplicação da técnica de zona de mixagem. A princípio, um usuário qualquer, cuja localização se encontra marcada em azul, ingressa na zona de mixagem 1 e obtém um pseudo-id que será usado na comunicação com o serviço, e só então passa a dispor do serviço coberto pela zona de mixagem. Ao se deslocar para a zona de mixagem 2, o usuário se comunica com o anonimizador e obtém um novo pseudo-id para esta zona. Requisições realizadas ao serviço não conterão qualquer informação de localização do usuário, uma vez que a única informação de localização utilizada é referente à zona de mixagem a qual ele serve.

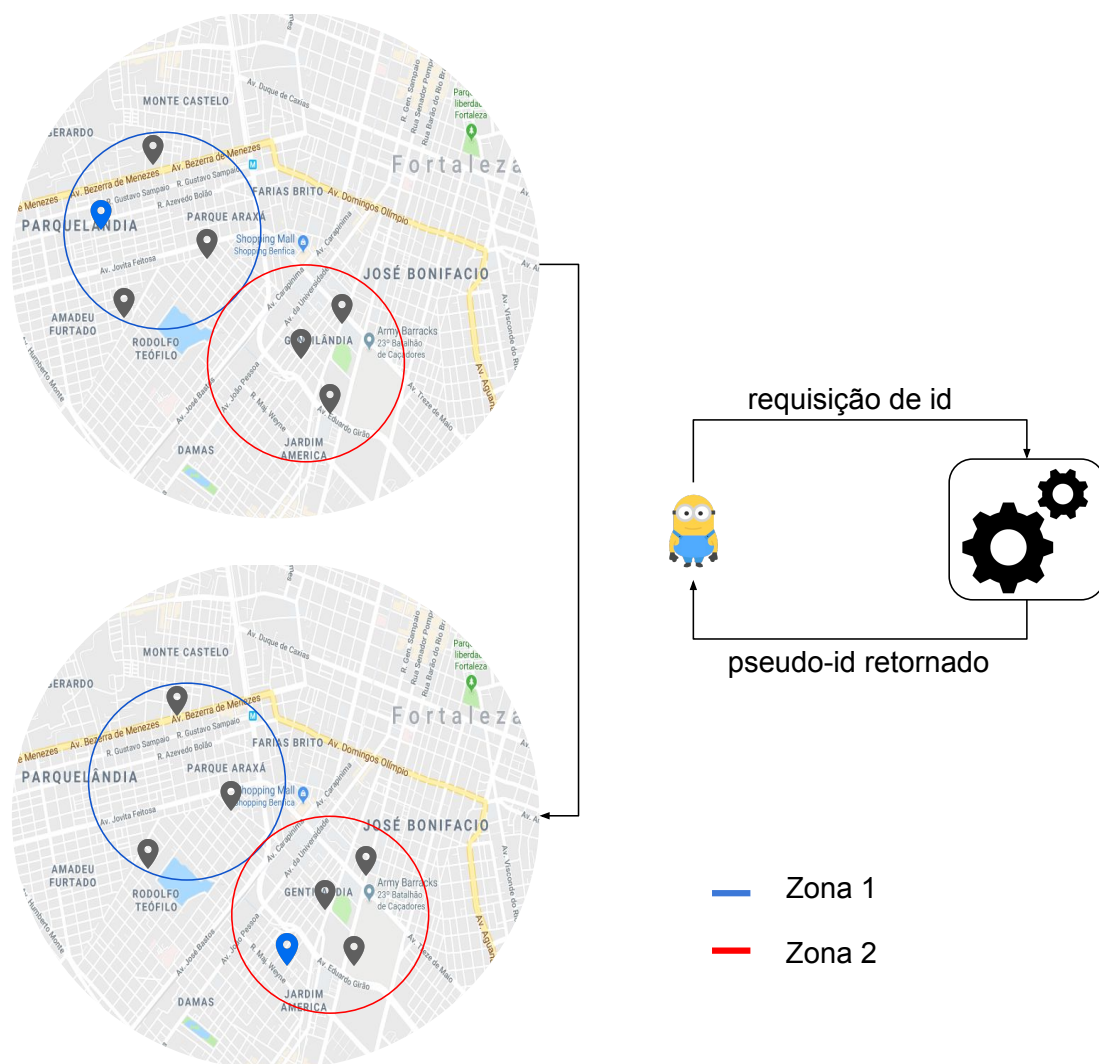


Figura 3.11. Processo de anonimização utilizando zona de mixagem.

### 3.6.3. Localizações falsas

Consultas realizadas a serviços de localização possuem dois componentes básicos: a informação de localização e o conteúdo da informação. Enquanto o primeiro contém qualquer informação relevante à localização, o segundo contém a requisição do serviço, por exemplo, a solicitação de dados de navegação em função da informação de localização informada [16].

A técnica de localizações falsas [25] procura garantir a privacidade dos dados de localização do usuário através de um processo de anonimização da *informação de localização* presente na requisição do usuário. Dessa forma, a informação de localização do usuário em uma requisição contém as informações da localização atual acrescida de localizações falsas, cujo objetivo é mascarar a localização verdadeira do usuário [16].

Diferentemente dos modelos tradicionais de  $k$ -anonimato e zona de mixagem, a técnica de seleção de localizações falsas não necessita da presença de servidores confiáveis para a realização do processo de anonimização, diminuindo o risco de exposição. Esse processo é realizado pelo cliente. A seleção de localizações falsas é realizada nos próprios dispositivos utilizados pelos usuários para consumir o serviço de localização.

O objetivo da presença de localizações falsas é garantir que a probabilidade de se identificar a localização real dentre aquelas presentes na requisição não seja maior que  $\frac{1}{k}$ , em que  $k$  é o grau de privacidade desejado para uma requisição com uma quantidade de  $k$  localizações presentes.



Figura 3.12. Técnica de Localizações Falsas.

A Figura 3.12 ilustra a aplicação da técnica de localizações falsas, onde o usuário, por intermédio de seu dispositivo, envia uma requisição anonimizada ao provedor de serviço. O processo de anonimização seleciona  $k - 1$  localizações, na Figura 3.12 representado em preto, que serão enviadas na informação de localização da requisição, juntamente com a localização real do usuário, em laranja na figura. O servidor receberá a requisição contendo  $k$  localizações, e responderá a solicitação contida no conteúdo de



requisição, tendo como referência cada uma das localizações presentes na informação de localização. O dispositivo então filtra a resposta referente à localização real.

O mecanismo de seleção das localizações falsas é fundamental para garantir a privacidade de localização do usuário, uma vez que esta técnica, assim como as técnicas que abordam o modelo de anonimização estão sujeitas a ataques de conhecimento, que, conforme discutido na seção 3.5.4.3, utilizam de conhecimentos prévios para inferir dados sensíveis dos usuários. Desta forma, várias abordagens foram propostas, a fim de solucionar este problema e garantir uma seleção de localizações falsas que minimize o risco de exposição dos dados de localização do usuário, utilizando para isto, do próprio conhecimento prévio disponível ao provedor de serviço [45, 38].

Embora esta técnica garanta que a qualidade do serviço não terá qualquer impacto pela anonimização das informações de localização, já que não existe qualquer perda de utilidade da mesma, a estratégia de seleção de localizações, bem como o valor de  $k$  tem um impacto direto tanto na privacidade garantida, como no esforço computacional desta seleção. É importante que a estratégia de seleção considere fatores que elimine localizações improváveis que aumentariam a probabilidade de se identificar a real localização. Por exemplo, se for selecionado localizações pouco habitadas, ou de circulação proibida, juntamente com localizações bem frequentadas, estas últimas possuem uma probabilidade bem superior de ser a localização real do usuário dentre as outras. Além disso, se o valor de  $k$  for muito grande, pode ser bem custoso selecionar localizações que atendem aos requisitos de seleção do algoritmo utilizado.

#### 3.6.4. Ofuscação de localização

As técnicas de ofuscação de localização procuram garantir a preservação de privacidade do usuário através da redução deliberada da precisão da localização do mesmo. Em sua abordagem tradicional apresentada por Ardagna *et al.* [4, 5], a informação de localização não mais apresenta as coordenadas da localização atual do usuário. Em vez disso, é enviado uma área circular, representada pela tupla  $Area(r, x_c, y_c)$ , centralizada nas coordenadas geográficas  $(x_c, y_c)$  e raio  $r$ , onde a probabilidade de a localização do usuário estar contida dentro dessa área é igual a 1. A Figura 3.13 ilustra uma requisição anonimizada pela técnica de ofuscação, onde o usuário, com localização em laranja, ao enviar sua requisição, envia como informação de localização uma área circular de raio  $r$ , centrada nas coordenadas do ponto central em preto, semelhante a todos os outros usuários presentes na área de ofuscação.

Uma abordagem alternativa é proposta por Gutscher [24], onde operações geométricas (*i.e.*, rotação, translação) são executadas sobre as coordenadas geográficas da localização atual, reduzindo sua precisão. A Figura 3.14 ilustra a aplicação desta técnica através da operação de translação sobre a localização atual do usuário.

Estas técnicas têm a vantagem de proteger a privacidade do usuário através da diminuição da precisão da localização apresentada, entretanto, este ganho de privacidade implica na perda de utilidade desta informação, que dependendo do serviço pode ter um impacto na qualidade do serviço, muitas vezes inviabilizando. Por exemplo, imagine que você realize uma requisição a um aplicativo de transporte e passe sua localização imprecisa com uns duzentos metros de imprecisão. Certamente, o motorista não vai lhe



Figura 3.13. Técnica de Ofuscação de Localização usando área circular.

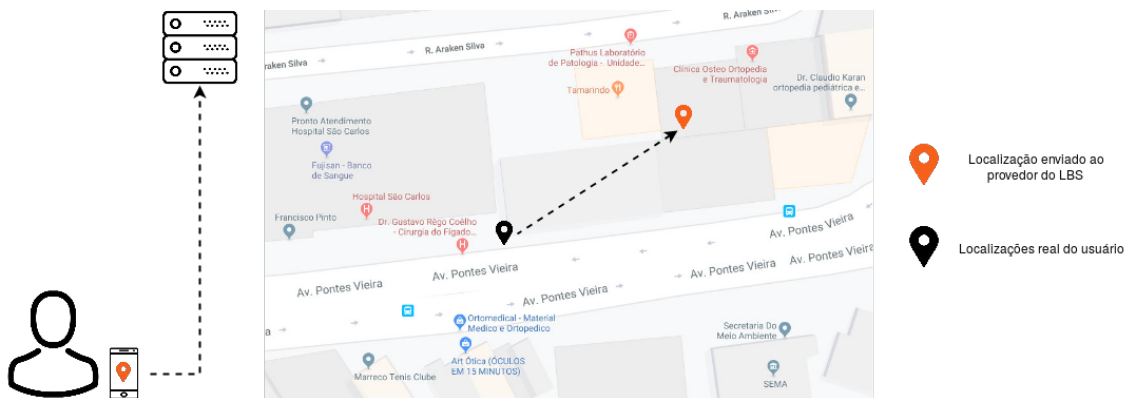


Figura 3.14. Técnica de Ofuscação de Localização usando operações geométricas.

encontrar. Entretanto, se você gostaria de utilizar um serviço buscando os restaurantes próximos dessa mesma localização imprecisa em duzentos metros, já será possível obter informação útil dessa requisição.

### 3.6.5. Geo-indistinguibilidade

Em se tratando de privacidade de localização, falta uniformidade quanto a conceitos básicos, como por exemplo, como quantificar a privacidade oferecida pelo serviço, ou qual a expectativa de privacidade do usuário no uso do serviço? O erro de estimativa de distância apresentado no início desta seção é uma métrica bastante usada para medir a privacidade de um mecanismo. Entretanto, essa noção de privacidade muitas vezes é definida em função do conhecimento adversário, que em geral é desconhecido.

A Geo-indistinguibilidade [3] é uma generalização da técnica de privacidade diferencial para o contexto de privacidade de localização. Desta forma, ela formaliza uma definição de privacidade de localização com sólidas garantias matemáticas que independe do conhecimento adversário. A ideia é garantir um nível de privacidade dentro de uma região geográfica de raio  $r > 0$ .

Antes de definir a noção de geo-indistinguibilidade é necessário definir o conceito de  $l$ -privacidade. Assim, podemos dizer que um indivíduo goza de  $l$ -privacidade dentro

de uma região de raio  $r$ , se quaisquer duas localizações a uma distância de no máximo  $r$  produzem distribuições similares, onde o nível de similaridade é definido em função de  $l$ . O parâmetro  $l$  representa o nível de privacidade dentro de  $r$ . Quanto maior for  $l$ , menor é o nível de privacidade. Para garantir uma maior utilidade do serviço,  $l$  precisa ser proporcional a  $r$ ,  $l = \epsilon r$ . Assim, quanto maior for o raio, menor é o nível de privacidade garantido pelo mecanismo.

Um mecanismo  $K$  garante  $\epsilon$ -geo-indistinguibilidade se para quaisquer duas localizações  $x$  e  $x' \in X$ :

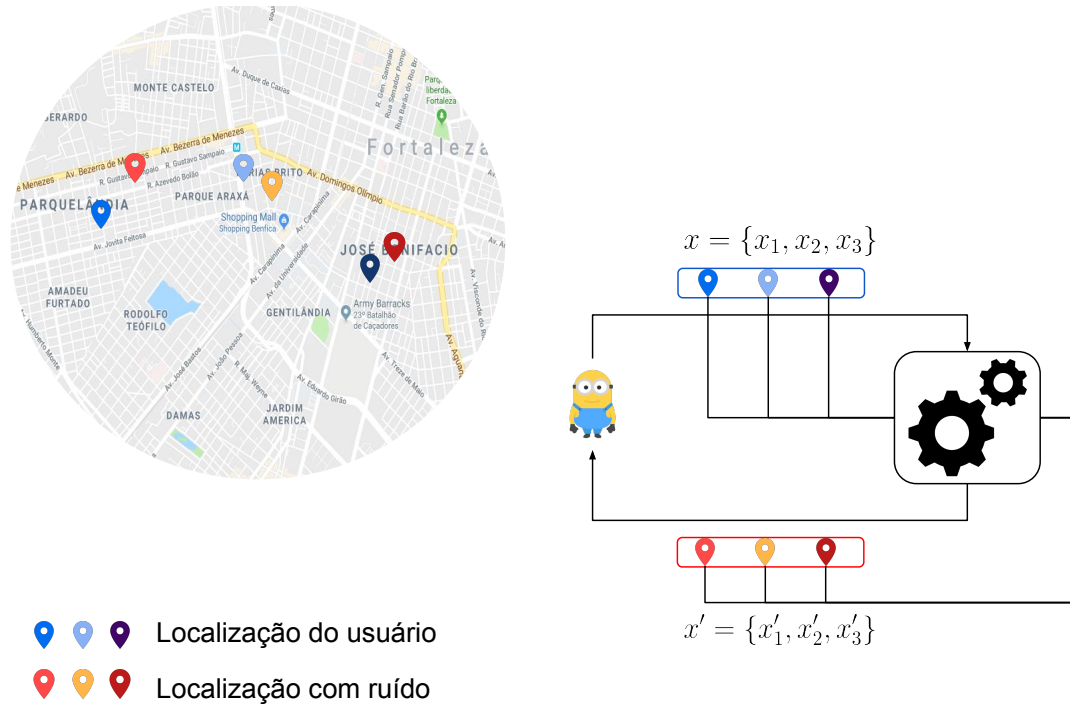
$$D_p(K(x), K(x')) \leq \epsilon d(x, x'),$$

onde  $D_p(K(x), K(x'))$  é a distância entre as distribuições produzidas por duas localizações  $x$  e  $x'$  pertencentes ao conjunto de todas as possíveis localizações  $X$ , e  $d(x, x')$  é a distância euclidiana entre as localizações  $x$  e  $x'$ . Desta forma, se considerarmos a distância máxima  $r$  entre  $x$  e  $x'$ , a definição força que a distância máxima entre as distribuições seja  $\epsilon r$ . Dessa forma, o mecanismo gera uma variável aleatória seguindo uma distribuição, onde para uma localização  $x$ , será retornado uma localização  $x'$  com certa probabilidade. Falaremos com mais detalhes sobre mecanismo na Seção 3.6.5.1.

Aplicar o mecanismo sobre uma localização é bem direta. Uma vez definido o orçamento de privacidade  $\epsilon$ , também chamado de *budget*, a região geográfica no entorno do usuário em função do raio  $r$ , basta aplicar o mecanismo e ele retornará uma localização que será usada na requisição do usuário. Entretanto, é possível que o usuário queira reportar mais de uma localização, como por exemplo, o conjunto de localizações frequentemente visitadas por ele. Nesse caso, a estratégia adotada de anonimização tem impacto no nível de privacidade garantido. Assim, considere o conjunto  $x = \{x_1, x_2, \dots, x_n\}$  contendo  $n$  localizações que o usuário pretende anonimizar. Uma estratégia é anonimizar cada localização de forma independente. Dessa forma, para o conjunto  $x$ , o mecanismo reportará o conjunto  $x' = \{x'_1, x'_2, \dots, x'_n\}$ . A Figura 3.15 demonstra a aplicação do mecanismo sobre as localizações do usuário de forma independente. Para cada localização do usuário é aplicado o mecanismo e reportado uma localização com ruído adicionado em função de um mecanismo geo-indistinguível.

Neste caso, como cada localização anonimizada  $x'_i$  a partir de  $x_i$  tem uma probabilidade de ocorrer em função da variável aleatória gerada pelo mecanismo, a probabilidade combinada do conjunto anonimizado é dada pelo produto das probabilidades de cada localização presentes no conjunto reportado, i.e.,  $Pr[K(x) = x'] = \prod_i Pr[K_o(x_i) = x'_i]$ . Como o mecanismo  $K$  permite uma combinação de  $n$  observações sobre as localizações de  $x$ , o nível de privacidade alcançado é de  $n\epsilon$ -geo-indistinguibilidade.

Uma alternativa para este problema de escalabilidade quando o  $n$  é muito grande seria aplicar o mecanismo sobre alguma função agregadora. Ou seja, caso seja possível, em virtude da natureza do serviço, realizar uma requisição reportando alguma informação agregada sobre  $x$ , como por exemplo, o centroide do conjunto. Assim, seja  $f$  uma função de agregação que retorna uma localização  $\hat{x}$  que represente a informação agregada de  $x$ , a localização anonimizada reportada pelo mecanismo  $K$  pode ser dada por  $K(f(\hat{x}))$ . A Figura 3.16 demonstra a aplicação do mecanismo sobre o centroide do conjunto de localizações de um usuário, calculado pela função  $f$ . Em vermelho tem-se a localização anonimizada a partir deste centroide. Portanto, se  $f$  for  $\Delta$ -sensível com respeito à  $d$  (distância



**Figura 3.15. Localizações anonimadas usando  $\epsilon$ -geo indistinguibilidade para cada localização de forma independente.**

euclidiana entre dois pontos) e  $d_\infty$  (distância máxima entre as localizações dos conjuntos de pontos), o que significa que  $d(f(x), f(x')) \leq \Delta d_\infty(x, x')$  para todos  $x$  e  $x'$ , e  $K$  satisfaz  $\epsilon$ -geo-indistinguibilidade, então a composição  $K \circ f$  satisfaz  $\Delta\epsilon$ -geo-indistinguibilidade.

### 3.6.5.1. Mecanismo

O mecanismo é o responsável pela adição do ruído aleatório à localização. A ideia é que para qualquer localização  $x$  em um plano contínuo, o mecanismo irá reportar uma localização  $x'$  no mesmo plano de acordo com a função de aleatoriedade. Esta função deve garantir que a probabilidade de se reportar um ponto dentro de uma certa área no entorno de  $x'$ , quando as localizações atuais são  $x_0$  ou  $x_1$  respectivamente, se difere de no máximo  $e^{\epsilon d(x_0, x_1)}$ . Intuitivamente, o que se busca é que quanto menor for a distância entre esta localização na área de  $x'$  e a localização atual, maior é a probabilidade de esta localização ser reportada.

O mecanismo de Laplace Planar [3] consegue capturar esta distribuição desejada. Para um  $\epsilon \in \mathbb{R}^+$ , e a localização atual  $x$ , a função de densidade de probabilidade (PDF) do mecanismo de Laplace centralizada em  $x$  é dada por:

$$\text{Laplace}_\epsilon(x)(x') = \frac{\epsilon^2}{2\pi} e^{-\epsilon d(x, x')},$$

onde  $\frac{\epsilon^2}{2\pi}$  é o fator de normalização. Portanto, o mecanismo  $K$  calcula uma localização  $x'$  que será reportado em função da distribuição de Laplace para a localização real  $x$  do



**Figura 3.16. Localizações anonimadas usando  $\epsilon$ -geo indistinguibilidade sobre a localização  $f(x)$ .**

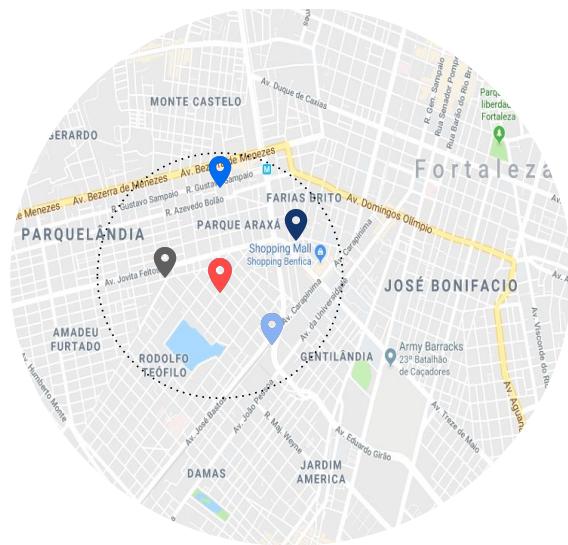
usuário.

A Figura 3.17 exemplifica esta situação. Em função da localização atual do usuário em vermelho na figura, o mecanismo apresenta uma probabilidade de reportar cada uma das localizações presentes. A localização reportada foi a localização em cinza, cuja probabilidade era de 12% segundo a tabela presente na figura. Observe, que as localizações que estão a uma distância mais próxima da localização atual possuem uma probabilidade maior de serem reportadas. Isto é decorrência da distribuição de Laplace utilizada.

### 3.6.6. Geo-indistinguibilidade Adaptativa

Buscando um melhor equilíbrio entre utilidade e privacidade, vários trabalhos se inspiraram na geo-indistinguibilidade [48, 11]. Entretanto, embora esta técnica tenha se mostrado promissora no controle da informação de localização publicada, ela não considera a potencial correlação existente entre as localizações em consultas contínuas ao serviço de localização. Conseqüentemente, um adversário pode explorar esta vulnerabilidade para reduzir o nível de privacidade da localização do usuário e obter uma melhor estimativa, violando os preceitos de privacidade garantidos pelo mecanismo.

Esta correlação é originada principalmente pela existência de padrões de movimento entre as diversas localizações que compõem as trajetórias dos usuários [2]. Como dito anteriormente, os serviços de localização necessitam que as informações de localização tenham um certo nível de utilidade para manter sua qualidade. Dessa forma, os mecanismos de privacidade precisam revelar parte das informações de localização, o que em geral, permite a existência dessa correlação mesmo que esta informação esteja anoni-



Localização	Prob.
	2,2%
	12%
	5,3\$
	3%

Localização do usuário

**Figura 3.17. Distribuição de probabilidade do mecanismo de Laplace planar.**

mizada.

O estudo da correlação entre as localizações pode ser bastante útil, servindo para aumentar a eficiência de serviços, como por exemplo, na predição das próximas localizações visando uma melhoria no tempo de resposta de requisições. Entretanto, esta correlação pode também ser usada para reduzir o erro de estimativa de um adversário que busque identificar a localização dos usuários do serviço.

A geo-indistinguibilidade adaptativa procura apresentar um modelo adaptativo que seja capaz de preservar a privacidade apesar da correlação existente entre as localizações. Garantindo, que o nível de ruído possa ser ajustável em função da correlação existente entre as localizações reportadas. Para isso, o mecanismo utiliza uma janela de predição, que define o período de tempo de observação das localizações reportadas, a fim de se medir a correlação existente entre elas e a localização atual do usuário.

### 3.6.6.1. Modelo

O objetivo da geo-indistinguibilidade adaptativa é reportar localizações que sejam resistentes a ataques de análise de correlação, ou seja, ataques que busquem identificar a correlação existente entre as localizações. Estes tipos de ataques utilizam das localizações reportadas anteriormente para prever a próxima localização do usuário. O mecanismo adaptativo busca ajustar o nível de ruído em função dessa capacidade de se prever a próxima localização.

Mensurar o nível adequado de privacidade e utilidade em função da correlação é uma tarefa complexa e precisa ser adequada também ao tipo de serviço. Serviços com maior precisão, como serviços de navegação, apresentam uma expectativa de estimativa de erro baixo, já que as localizações reportadas deverão ter uma maior correlação entre si, já que existe uma necessidade maior de acurácia na localização reportada. Já um serviço com baixa precisão, como serviços de meteorologia, permite uma expectativa de estimativa de erro alta, já que as localizações reportadas deverão apresentar uma correlação baixa. Para melhor ajustar a quantidade de ruído o modelo adaptativo proposto define quatro parâmetros a ser ajustado em função da exigência do serviço:

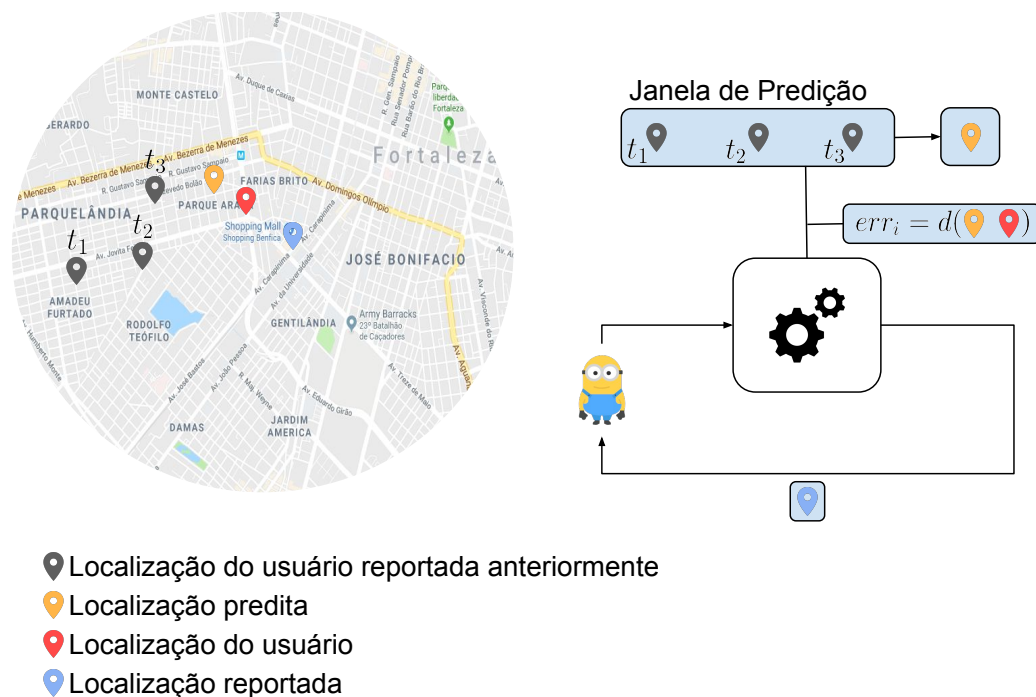
- $\Delta_1$ : Limite inferior de correlação;
- $\Delta_2$ : Limite superior de correlação;
- $\alpha$ : fator de multiplicação do orçamento de privacidade para baixa correlação;
- $\beta$ : fator de multiplicação do orçamento de privacidade para alta correlação;

O erro de estimativa pode ser classificado em alto, médio ou baixo. Um erro de estimativa alto indica que o nível de privacidade é alto, portanto, um ruído baixo pode ser utilizado para melhorar a qualidade do serviço sem afetar a privacidade. Um erro de estimativa baixo indica que o nível de privacidade está baixo, necessitando um maior ruído para garantir a privacidade. Um erro de estimativa médio indica que um ruído de nível mediano deve ser utilizado.

Seja  $T = \{x_1, \dots, x_n\}$  a trajetória do usuário, e  $x_i = (lat, lon, ts)$  uma localização com coordenadas de latitude  $lat$  e longitude  $lon$  em um momento de tempo  $ts$ .  $OT = \{z_1, \dots, z_n\}$  é a trajetória ofuscada do usuário ao aplicar um mecanismo geo-indistinguível, como o mecanismo de Laplace planar apresentado na Seção 3.6.5.1 para um orçamento de privacidade  $\varepsilon_i$ . A localização  $x'_i$  representa a localização predita dada a localização  $x_i$  ao se aplicar uma simples regressão linear sobre as localizações ofuscadas em duas janelas de tamanho  $ws$ ,  $LatW$  e  $LonW$ . As janelas de predição  $LatW$  e  $LonW$  são usadas para prever as coordenadas de latitude e longitude de  $x'_i$ . O erro estimado  $err_i$  é calculado em função da distância euclidiana entre  $x_i$  e  $x'_i$ . Assim, a correlação em função do erro estimado  $err_i$  pode ser definida como:

- baixa:  $err_i < \Delta_1$
- média:  $\Delta_2 \geq err_i \geq \Delta_1$
- alta:  $\Delta_2 < err_i$

A quantidade de ruído necessário é calculada em função da correlação existente. Se a correlação for baixa o ruído necessário é baixo, portanto, é aplicado um fator multiplicação  $\alpha$  sobre o  $\varepsilon$ , assim, o orçamento utilizado pelo mecanismo será de  $\varepsilon_i = \alpha\varepsilon$ . Se o nível de correlação for alto, é aplicado um fator multiplicação  $\beta$ , e o orçamento utilizado será de  $\varepsilon_i = \beta\varepsilon$ . Se o nível for médio, então o orçamento utilizado é o próprio  $\varepsilon$ .



**Figura 3.18. Geo-indistinguibilidade adaptativa.**

Dessa forma, o nível de privacidade, definido em função do valor do *budget*, é adaptado de acordo com a capacidade de um atacante prever a próxima localização do usuário ao analisar a correlação existente entre as localizações presentes na janela de predição.

A Figura 3.18 exemplifica a aplicação do mecanismo adaptativo. Em cinza estão as localizações ofuscadas reportadas anteriormente. Aplicando uma regressão linear sobre janelas de tamanho 3, a localização estimada está em laranja. O erro estimado para a localização correta é medido entre a localização predita e a localização atual em vermelho. Supondo que o erro é baixo, indicando uma forte correlação, será necessário um ruído maior a fim de diminuir a correlação na próxima janela de predição. Dessa forma é aplicado o mecanismo de Laplace sobre a localização atual, com o orçamento igual a  $\epsilon_i = \beta\epsilon$ , reportando a localização ofuscada em azul.

### 3.7. Desafios de Pesquisa

Em privacidade de dados de forma geral, alcançar um equilíbrio entre privacidade e utilidade dos dados já é uma tarefa bem complexa. Ao longo deste curso falamos bastante neste tema, inclusive, sendo talvez um dos maiores motivadores no surgimento de novos trabalhos que buscam otimizar técnicas já existentes buscando garantir um valor ótimo para os parâmetros de privacidade, justamente para que não seja adicionado mais ruído do que necessário, e nem que seja subestimado esta quantidade.

Em serviços de localização, onde a precisão da localização do demandante do serviço impacta diretamente na qualidade do serviço prestado ao usuário. Esta análise do ruído necessário para que a informação de localização exposta seja o suficiente para



garantir a privacidade e a qualidade do serviço continua sendo uma tarefa de extrema complexidade, e depende diretamente do serviço utilizado e do nível de privacidade desejado. A privacidade diferencial em serviços de localização, dada a sua estratégia de adição de ruído aleatório, embora promissora ainda apresenta limitações quanto à utilidade dos dados fornecido ao provedor do serviço, principalmente pela dificuldade em se tratar os dados correlacionados. Desta forma vemos grande potencial de estudo no estudo da correlação de dados de localização e seu ajuste no uso de mecanismos diferencialmente privados.

### **3.8. Conclusão**

Este capítulo conclui que a preservação da privacidade de dados acerca de indivíduos é um problema desafiador. Técnicas de anonimização têm sido utilizadas para a disponibilização de dados sensíveis, procurando encontrar o melhor balanceamento entre privacidade e utilidade que atenda às diversas partes envolvidas no processo de disponibilização de dados. Diferentes tipos de ataques à privacidade têm sido empregados por usuários maliciosos com a intenção de violar informações sensíveis de bases de dados abertas. Para tal fim, os atacantes utilizam conhecimento que muitas vezes é imensurável, devido aos diversos cenários em que informações podem ser obtidas. No contexto de dados de localização, este risco se potencializa, em virtude das informações agregadas ao dado geográfico buscado quando de uma solicitação a um serviço de localização, que servem de munição para os agentes maliciosos. Este capítulo apresentou as principais técnicas no estado da arte em preservação de privacidade de dados de localização. Os modelos de anonimização buscam proteger de ataques de ligação ao registro, ou seja, prevenir a vinculação entre a identidade do usuário e sua localização, evitando a re-identificação de indivíduos, geralmente utilizando técnicas de supressão e generalização. Os modelos de ofuscação, por sua vez, buscam proteger a localização em si, garantindo que esta não seja revelada, mesmo no uso de serviços de localização. A Privacidade Diferencial se destaca por fornecer soluções de preservação de privacidade, onde um ruído aleatório controlado é adicionado a localização do usuário, garantindo que a localização real do usuário estará protegida independentemente do conhecimento do atacante.

Finalmente, entendemos que o problema da garantia de privacidade de dados de localização dos usuários de serviços de localização continua cientificamente relevante. A busca por um ponto ideal na curva de solução de compromisso entre privacidade do indivíduo e a utilidade do dado fornecido para esse tipo de serviço deve pautar os próximos passos da pesquisa. Este aspecto é particularmente importante no contexto de localizações pois a qualidade do serviço é dependente da precisão do dado de localização, portanto o envio de dado perturbado para o provedor de serviço tende a impactar negativamente na qualidade. Tanto o paradigma de anonimização sintática, quanto o modelo de Privacidade Diferencial apresentam aspectos de revisão que devem ser vistos como oportunidades de pesquisas e desenvolvimento. Avanços em ambos os paradigmas são necessários para garantir que o futuro ofereça cada vez mais proteção à privacidade de indivíduos e ao mesmo tempo haja dados úteis e disponíveis para pesquisadores, testadores e analistas de dados.

## Agradecimentos

Esta trabalho foi parcialmente financiada pela Lenovo, como parte do seu investimento em pesquisa e desenvolvimento de acordo com a Lei de Informática, pela CAPES (1836136), CNPq (122201/2018-3) e pelo LSBDD/UFC.

## Referências

- [1] Aggarwal, C. C. and Philip, S. Y. (2008). A framework for condensation-based anonymization of string data. *Data Mining and Knowledge Discovery*, 16(3):251–275.
- [2] Al-Dhubhani, R. and Cazalas, J. M. (2018). An adaptive geo-indistinguishability mechanism for continuous lbs queries. *Wireless Networks*, 24(8):3221–3239.
- [3] Andrés, M. E., Bordenabe, N. E., Chatzikokolakis, K., and Palamidessi, C. (2013). Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 901–914.
- [4] Ardagna, C. A., Cremonini, M., Damiani, E., De Capitani di Vimercati, S., and Samarati, P. (2007). Location privacy protection through obfuscation-based techniques. In Barker, S. and Ahn, G.-J., editors, *Data and Applications Security XXI*, pages 47–60.
- [5] Ardagna, C. A., Cremonini, M., De Capitani di Vimercati, S., and Samarati, P. (2011). An obfuscation-based approach for protecting location privacy. *IEEE Transactions on Dependable and Secure Computing*, 8(1):13–27.
- [6] Bayardo, R. J. and Agrawal, R. (2005). Data privacy through optimal k-anonymization. In *21st International conference on data engineering (ICDE'05)*, pages 217–228.
- [7] Beresford, A. R. and Stajano, F. (2003). Location privacy in pervasive computing. *IEEE Pervasive computing*, (1):46–55.
- [8] Blocki, J., Datta, A., and Bonneau, J. (2016). Differentially private password frequency lists. *IACR Cryptology ePrint Archive*, 2016:153.
- [9] BRITO, F. T. and MACHADO, J. C. (2017). Preservação de privacidade de dados: Fundamentos, técnicas e aplicações. *Jornadas de atualização em informática*, pages 91–130.
- [10] Brito, F. T. and Machado, J. C. (2017). Preservação de privacidade de dados: Fundamentos, técnicas e aplicações. In Delicato, F. C., Pires, P. F., and Silveira, I. F., editors, *Jornadas de A tualização em Informática 2017*. Sociedade Brasileira de Computação - SBC.
- [11] Chatzikokolakis, K., Palamidessi, C., and Stronati, M. (2015). Constructing elastic distinguishability metrics for location privacy. *arXiv preprint arXiv:1503.00756*.

- [12] Dewri, R., Ray, I., Ray, I., and Whitley, D. (2008). On the optimal selection of  $k$  in the  $k$ -anonymity problem. In *24th ICDE International Conference on Data Engineering*, pages 1364–1366, Cancun, Mexico.
- [13] Domingo-Ferrer, J., Sánchez, D., and Soria-Comas, J. (2016). Database anonymization: Privacy models, data utility, and microaggregation-based inter-model connections. *Synthesis Lectures on Information Security, Privacy, & Trust*, 8(1):1–136.
- [14] Domingo-Ferrer, J. and Soria-Comas, J. (2015). From  $t$ -closeness to differential privacy and vice versa in data anonymization. *Knowledge-Based Systems*, 74:151–158.
- [15] Domingo-Ferrer, J. and Torra, V. (2001). A quantitative comparison of disclosure control methods for microdata. *Confidentiality, disclosure and data access: theory and practical applications for statistical agencies*, pages 111–134.
- [16] Duarte Neto, E. R., Machado, J. C., and Mendonça, A. L. (2019). PrivLBS: Preserving privacy in location based services. *J. Inf. Data Manag.*, 10(2):81–96.
- [17] Dwork, C. (2006). Differential privacy. In *33rd International Colloquium*, pages 1–12.
- [18] Dwork, C. (2008). Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer.
- [19] Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407.
- [20] Fung, B. C., Wang, K., Fu, A. W.-C., and Yu, P. S. (2010a). *Introduction to Privacy-Preserving Data Publishing: Concepts and Techniques*. Chapman & Hall/CRC, 1st edition. ISBN 978-1-4200-9148-9.
- [21] Fung, B. C. M., Wang, K., Chen, R., and Yu, P. S. (2010b). Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):1–53.
- [22] Gedik, B. and Liu, L. (2007). Protecting location privacy with personalized  $k$ -anonymity: Architecture and algorithms. *IEEE Transactions on Mobile Computing*, 7(1):1–18.
- [23] Goldreich, O. (2003). Cryptography and cryptographic protocols. *Distributed Computing*, 16(2-3):177–199.
- [24] Gutscher, A. (2006). Coordinate transformation - a solution for the privacy problem of location based services? In *Proceedings 20th IEEE International Parallel Distributed Processing Symposium*, pages 7 pp.–.
- [25] Kido, H., Yanagisawa, Y., and Satoh, T. (2005). An anonymous communication technique using dummies for location-based services. In *Proceedings of the Int. Conf. on Pervasive Services, ICPS'05*, pages 88–97. IEEE.

- [26] Lee, J. and Clifton, C. (2011). *How Much Is Enough? Choosing  $\epsilon$  for Differential Privacy*, pages 325–340. Springer Berlin Heidelberg.
- [27] LeFevre, K., DeWitt, D. J., and Ramakrishnan, R. (2005). Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60. ACM.
- [28] Li, H., Sun, L., Zhu, H., Lu, X., and Cheng, X. (2014). Achieving privacy preservation in wifi fingerprint-based localization. In *INFOCOM, 2014 Proceedings IEEE*, pages 2337–2345. IEEE.
- [29] Li, N., Li, T., and Venkatasubramanian, S. (2007). t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115. IEEE.
- [30] Liu, B., Zhou, W., Zhu, T., Gao, L., and Xiang, Y. (2018). Location privacy and its applications: A systematic study. *IEEE Access*, 6:17606–17624.
- [31] Machado, J., Neto, E. D., and Bento Filho, M. (2019). Técnicas de privacidade de dados de localização. *Sociedade Brasileira de Computação*.
- [32] Machanavajjhala, A., Kifer, D., Abowd, J. M., Gehrke, J., and Vilhuber, L. (2008). Privacy: Theory meets practice on the map. In *Proceedings of the 24th International Conference on Data Engineering, ICDE 2008, April 7-12, 2008, Cancún, México*, pages 277–286.
- [33] Matejka, J. and Fitzmaurice, G. (2017). Same stats, different graphs: generating datasets with varied appearance and identical statistics through simulated annealing. In *Proceedings of the 2017 CHI conference on human factors in computing systems*, pages 1290–1294.
- [34] McSherry, F. (2010). Privacy integrated queries: an extensible platform for privacy-preserving data analysis. *Commun. ACM*, 53(9):89–97.
- [35] McSherry, F. and Talwar, K. (2007). Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE.
- [36] Meyerson, A. and Williams, R. (2004). On the complexity of optimal k-anonymity. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228. ACM.
- [37] Mir, D. J., Isaacman, S., Cáceres, R., Martonosi, M., and Wright, R. N. (2013). DP-WHERE: differentially private modeling of human mobility. In *Proceedings of the 2013 IEEE International Conference on Big Data, 2013, Santa Clara, CA, USA*, pages 580–588.
- [38] Neto, E. R. D., Mendonça, A. L. C., Brito, F. T., and Machado, J. C. (2018). Privlbs: uma abordagem para preservação de privacidade de dados em serviços baseados em localização. In *Brazilian Symposium on Databases SBBD*, Rio de Janeiro, Brazil.

- [39] Nguyen, H. H., Kim, J., and Kim, Y. (2013). Differential privacy in practice. *Journal of Computing Science and Engineering*, 7(3):177–186.
- [40] NIU, B., Chen, Y., Wang, Z., Wang, B., Li, H., et al. (2020). Eclipse: Preserving differential location privacy against long-term observation attacks. *IEEE Transactions on Mobile Computing*.
- [41] Portela, T. T., Vicenzi, F., and Bogorny, V. (2019). Trajectory data privacy: Research challenges and opportunities. In *GEOINFO*, pages 99–110.
- [42] Primault, V., Boutet, A., Mokhtar, S. B., and Brunie, L. (2018). The long road to computational location privacy: A survey. *IEEE Communications Surveys & Tutorials*.
- [43] Schiller, J. and Voisard, A. (2004). *Location-based services*. Elsevier.
- [44] Shokri, R., Theodorakopoulos, G., Le Boudec, J.-Y., and Hubaux, J.-P. (2011). Quantifying location privacy. In *2011 IEEE symposium on security and privacy*, pages 247–262. IEEE.
- [45] Sun, G., Chang, V., Ramachandran, M., Sun, Z., Li, G., Yu, H., and Liao, D. (2017). Efficient location privacy algorithm for internet of things (iot) services and applications. *Journal of Network and Computer Applications*, 89:3–13.
- [46] Sweeney, L. (2002). k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570.
- [47] Tan, V. Y. F. and Ng, S.-K. (2007). Generic probability density function reconstruction for randomization in privacy-preserving data mining. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 76–90. Springer.
- [48] Theodorakopoulos, G. (2015). The same-origin attack against location privacy. In *Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society*, pages 49–53.
- [49] Truta, T. M., Campan, A., and Meyer, P. (2007). Generating microdata with p-sensitive k-anonymity property. In *Workshop on Secure Data Management*, pages 124–141. Springer.
- [50] Wang, K., Fung, B. C., and Yu, P. S. (2005). Template-based privacy preservation in classification problems. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE.
- [51] Wong, R. C.-W. and Fu, A. W.-C. (2010). Privacy-preserving data publishing: An overview. *Synthesis Lectures on Data Management*, 2(1):1–138.
- [52] Yang, D., Fang, X., and Xue, G. (2013). Truthful incentive mechanisms for k-anonymity location privacy. In *2013 Proceedings IEEE INFOCOM*, pages 2994–3002. IEEE.

- [53] Zheng, Y., Zhang, L., Xie, X., and Ma, W.-Y. (2009). Mining correlation between locations using human location history. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 472–475.
- [54] Zhu, X., Chi, H., Niu, B., Zhang, W., Li, Z., and Li, H. (2013). Mobicache: When k-anonymity meets cache. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 820–825, Atlanta, GA, USA. IEEE.

## Capítulo

# 4

## Blockchain e Contratos Inteligentes para Aplicações em IoT, Uma Abordagem Prática

Jauberth Weyll Abijaude, Fabíola Greve, Péricles de Lima Sobreira

### *Abstract*

*The Internet of Things aggregates devices able to capture information and interfere in the environment, in such a way to obtain, generate and send data on a large scale to different domain application systems, such as agriculture, industry, trade, and governments. These systems need a security layer to guarantee, among other requirements, the irrefutability of transactions and the integrity of the manipulated data. In this sense, an integration with the blockchain technology, through smart contracts, would meet this need. Blockchain is a disruptive technology that offers a digital trust network for conducting transactions between peers, often unknown. This chapter presents a recent boarder research of IoT with blockchain; introduces a classification of layered blockchain technology and conducts a comprehensive study on consensus strategies and IoT applications with blockchain. In the end, it offers a guide with information that allows interested parties to design training in this area, including the realization of practical exercises.*

### *Resumo*

*A Internet das Coisas agrega dispositivos capazes de capturar informações e interferir no ambiente, de maneira a obter, gerar e enviar dados em larga escala para sistemas de domínios de aplicações diferentes, tais como agricultura, indústria, comércio e governos. Estes sistemas precisam de uma camada de segurança para garantir, dentre outras características, a irrefutabilidade das transações e a integridade dos dados manipulados. Neste sentido, a integração com a blockchain, através dos contratos inteligentes, atenderia a esta necessidade. A blockchain é uma tecnologia disruptiva que oferece uma rede de confiança digital para a realização de transações entre pares, muitas vezes desconhecidos. Este capítulo apresenta pesquisas recentes na fronteira da IoT com a blockchain; apresenta uma classificação da tecnologia de blockchain em camadas e realiza um estudo amplo sobre as estratégias de consenso e aplicações IoT com blockchain. Ao final, fornece um guia com informações que permitam aos interessados a concepção de treinamentos nesta área, contemplando, inclusive, a realização de exercícios práticos.*

## 4.1. Introdução

O avanço de inúmeras tecnologias, incluindo sensores; atuadores; computação embarcada, na nuvem e na névoa; e o surgimento de uma nova geração de dispositivos sem fio, faz com que muitos objetos ou coisas em nosso dia a dia se tornem interoperáveis. A Internet das Coisas, (do inglês *Internet of Things* - IoT), adota processamento, arquitetura de comunicação, tecnologias inteligentes e estratégias de gerenciamento para integrar um grande número de objetos inteligentes à Internet [Dorri et al. 2016].

As aplicações da IoT permitem a comunicação direta e a interação entre os dispositivos pela Internet. Os dispositivos IoT atuais incluem smartphones, eletrodomésticos inteligentes, veículos e sensores internos e externos. No entanto, com o grande número de dispositivos, os aplicativos IoT tradicionais estão enfrentando desafios em muitos aspectos, incluindo integridade de dados, segurança e sua robustez [Lao et al. 2020].

A blockchain e as tecnologias de registro distribuído (*Distributed Ledger Technologies*, DLTs) oferecem suporte distribuído e confiável para a realização de transações entre participantes, que não necessariamente têm confiança entre si, e que se encontram dispersos numa rede P2P. É considerada uma tecnologia disruptiva e com potencial para substituir entidades certificadoras e centralizadoras das transações de negócios, tais como bancos, governos, cartórios, etc., conferindo-lhes atributos de segurança como a auditabilidade, irrefutabilidade, não-repudição, rastreabilidade, consistência e (um certo nível de) anonimato [Greve et al. 2018].

Por outro lado, os Contratos Inteligentes (SC do inglês *Smart Contracts*) servem como um mecanismo que torna os sistemas blockchain flexíveis e escalonáveis para lidar com tarefas relacionadas a contratos entre partes [Christidis and Devetsikiotis 2016], permitindo o desenvolvimento de aplicações verdadeiramente descentralizadas. Em termos gerais, os SCs são vistos como procedimentos gerais para construir sistemas de automação e controle de dispositivos IoT, e atualmente, muitas empresas fornecem soluções de IoT usando SCs [Buterin et al. 2014].

Integrar a blockchain aos SCs e às aplicações de IoT consiste numa engrenagem que não é trivial, onde se observa situações peculiares, dignas de um olhar atencioso e margeado por algumas questões a serem aprimoradas [Bogner et al. 2016]. Diante de tal cenário, este capítulo explora, de forma teórica e prática, o casamento da blockchain, SCs e IoT, trazendo o estado da arte das pesquisas recentes na área.

O capítulo está estruturado em oito seções. A Seção 4.2 introduz conceitos de blockchain, SCs, IoT e blockchain para IoT. A Seção 4.3 aprofunda o tema de blockchain, dividindo-a em 4 camadas (dados, rede, consenso e aplicação) e discutindo sobre cada uma delas [Wu et al. 2019]. A Seção 4.4 apresenta os protocolos de consenso e a correlação dos mesmos com IoT [Wu et al. 2019, Lao et al. 2020]. A Seção 4.5 explora as aplicações da blockchain, inclusive para IoT [Mistry et al. 2020, Zhang and Chen 2020] e apresenta um exemplo prático. A Seção 4.6 discute e sugere como abordar estes temas em cursos relacionados. A Seção 4.7 apresenta os desafios de pesquisa e a última, Seção 4.8 conclui o trabalho.



## 4.2. Principais Conceitos

Esta seção possui caráter introdutório com o objetivo de revisar conceitos fundamentais para o acompanhamento do minicurso e nivelar os conhecimentos. São quatro subseções que exploram a blockchain, os contratos inteligentes, a internet das coisas e blockchain para IoT.

### 4.2.1. Blockchain

A primeira rede blockchain, apresentada em 2008, permitia transacionar valores digitais através de uma estrutura computacional distribuída [Nakamoto 2008]. Tal trabalho estabelecia as bases de um sistema econômico alternativo à base de uma moeda digital (ou criptomoeda), o Bitcoin. Em 2009, através da implementação de uma máquina de estados simplificada, foi lançada a primeira versão do software com um arranjo até então inédito, que proporcionou eliminar a terceira parte de confiança, necessária para as transações financeiras tradicionais.

Os elementos básicos e seminais desta tecnologia, combinados de forma engenhosa, sustentam de forma teórica/prática o desenvolvimento de aplicações descentralizadas, dentre elas as diversas criptomoedas. São eles:

- **Criptografia:** Satisfaz os requisitos de segurança do sistema e das aplicações. Dentre os recursos mais utilizados, destacam-se os resumos criptográficos (funções *hash*) e as assinaturas digitais;
- **Consenso distribuído:** Permite que participantes distribuídos coordenem as suas ações, de forma a alcançar decisões comuns, e assim garantir a manutenção da consistência dos seus estados (*safety*) e o progresso do sistema (*liveness*), apesar da existência de falhas [Greve 2005];
- **Livro razão distribuído:** O livro-razão (*ledger*) é uma estrutura de dados imutável, em que transações são registradas e o estado global do sistema é mantido replicado em todos os nós da rede P2P.

Como consequência desta composição tecnológica, a blockchain (ou BC) garante algumas propriedades que contribuem de forma inovadora para o desenvolvimento de aplicações descentralizadas e sistemas, como por exemplo [Greve et al. 2018]:

- **Descentralização:** Sistemas e aplicações que usam a BC não precisam de uma entidade central para coordenar as ações, as tarefas são executadas de forma distribuída;
- **Disponibilidade e integridade:** Os dados e as transações são replicados para todos os participantes da BC, mantendo o sistema seguro e consistente;
- **Transparência e auditabilidade:** A cadeia de blocos que registra as transações é pública e pode ser auditada e verificada;
- **Imutabilidade e Irrefutabilidade:** os registros são imutáveis e a correção só pode ser feita a partir de novos registros. O uso de recursos criptográficos garante que os lançamentos não podem ser refutados;

- Privacidade e Anonimidade: As transações são anônimas, com base nos endereços dos usuários. Os servidores armazenam apenas fragmentos criptografados dos dados do usuário;
- Desintermediação: A BC consegue eliminar terceiros em suas transações, atuando como um conector de sistemas de forma confiável e segura;
- Cooperação e incentivos: Oferta de um modelo de negócios à base de incentivos, à luz da teoria dos jogos. O consenso sob demanda passa a ser oferecido como serviço em diversos níveis e escopos.

Em 2013, surge uma nova plataforma, a *Ethereum*, que evolui para além das transações de uma criptomoeda. Implementada sob um modelo de máquina de *turing* completa, com uma nova criptomoeda, e ancorada sob alguns conceitos de seu antecessor, esta nova plataforma inova ao permitir que programas de computador possam ser armazenados e executados nas cadeias de blocos. Tais programas, conhecidos como contratos inteligentes não são em si uma novidade, pois já haviam sido propostos e definidos como um conjunto de cláusulas contratuais, especificado em formato digital, incluindo protocolos nos quais as partes cumprem estas cláusulas [Szabo 1997].

A rede *Ethereum* oferece uma máquina de estados determinística completa, que consiste em um estado único acessível globalmente, e uma máquina virtual que aplica mudanças a esse estado. Sob uma perspectiva mais prática, a *Ethereum* é uma infraestrutura de computação globalmente descentralizada e de código aberto que executa programas chamados contratos inteligentes. Ela usa a blockchain para sincronizar e armazenar as mudanças de estado do sistema, incorpora a criptomoeda *ether* e *gas*, sendo esta última para medir e restringir os custos dos recursos de execução [Antonopoulos 2017].

As transações, compostas de mensagem com remetente, destinatário, valor e carga útil de dados, dentre outros, são processadas pela Máquina Virtual Ethereum (EVM, do inglês *Ethereum Virtual Machine*). Esta máquina virtual é baseada em uma pilha que executa os *bytecodes* (instruções em linguagem de máquina), resultantes do processo de compilação dos contratos inteligentes [EVM 2020]. O estado da rede *Ethereum* é armazenado localmente em cada nó como um banco de dados, que contém as transações e o estado do sistema em uma estrutura de dados em *hash* serializada chamada de *Merkle Patricia Tree* [Merkle-Patricia-Tree 2020].

O modelo de consenso utilizado pela *Ethereum* é o mesmo do Bitcoin. Assim, ele emprega blocos sequenciais, com uma impressão digital de dados (*hash*) único, ponderados em importância pelo protocolo de consenso *Ethash* [Etash 2020], baseado em prova de trabalho (PoW, do inglês *proof of work*), para determinar a cadeia mais longa e, portanto, o estado atual. No entanto, por diversas razões, a *Ethereum 2.0* usa o *Casper*, baseado em prova de posse ou participação (PoS, do inglês *Proof of Stake*).

Estas duas blockchains, a Bitcoin e a *Ethereum*, são de caráter público. Elas são também conhecidas como não permissionadas ou de acesso aberto, com acesso anônimo, sem nenhum controle sobre a entrada e saída de nós na rede e sem confiança mútua entre si. Existem também as blockchains permissionadas ou federadas, onde os nós são conhecidos e precisam ser autenticados. São redes voltadas normalmente para ambientes

corporativos, onde cada participante tem um papel definido. A *BC Hyperledger Fabric* é um exemplo de BC privada.

#### 4.2.2. Contratos Inteligentes

O termo contrato inteligente foi mencionado pela primeira vez em 1997 por Nick Szabo [Szabo 1997]. Apesar de estar definido teoricamente, não havia, à época, uma infraestrutura computacional com custos e recursos equilibrados para a sua implementação.

Com o surgimento da plataforma *Ethereum*, a ideia de implementar tais contratos foi então consolidada, hospedando-os na blockchain. Desta forma, os contratos inteligentes passam a ser sistemas que movem ativos digitais automaticamente, de acordo com regras pré-especificadas [Buterin et al. 2014].

Os Contratos inteligentes são normalmente escritos em uma linguagem de alto nível, como a Solidity, suportado na rede *Ethereum*. Para serem executados, eles são compilados, e como resultado são geradas duas saídas: os *bytecodes* e a ABI (do inglês *Application Binary Interface*). Enquanto os *bytecodes* precisam ser enviados para a rede *Ethereum* usando uma transação específica de criação de contrato, a ABI constitui uma interface pela qual as aplicações podem acessar as funções e dados dos contratos. Tal acesso é feito em conjunto com o endereço *Ethereum* que identifica o contrato.

É importante ressaltar que os contratos precisam ser chamados por uma transação para serem executados. Isto pode ser feito por uma transação iniciada a partir de uma conta *Ethereum* ou por um contrato que pode chamar outro contrato e assim por diante, ressaltando que, em tal encadeamento, a primeira execução sempre será originada por uma conta *Ethereum*. Estas transações são atômicas e o estado só será modificado se todas as transações forem executadas com sucesso. Em caso de falha, todas as operações serão desfeitas e o estado da rede refeito como se nenhuma transação tivesse sido executada. Portanto, os contratos nunca funcionarão "por conta própria" ou "em segundo plano". Eles ficam dormentes até que uma transação acione-os através da ABI, de forma direta ou indireta.

Como os contratos são imutáveis, conseqüentemente não conseguimos alterar o seu código. Para excluí-lo, é necessário, antecipadamente, programar uma função com a opção de autodestruição. Isto remove o código e seu estado de seu endereço, deixando a conta em branco. Quaisquer transações enviadas para esse endereço de conta após a exclusão do contrato não resultam em nenhuma execução de código, porque não há mais nenhum código para executar. O uso desta função implica em reembolso de taxas, como forma de incentivar a liberação de recursos na rede. É necessário esclarecer que as transações já realizadas pelo contrato não são apagadas, pois a blockchain é imutável.

Segundo [Bartoletti and Pompianu 2017], os contratos podem ser classificados em cinco categorias: financeiro, notário, jogo, carteira e biblioteca.

Na categoria financeira, os contratos gerenciam, reúnem ou distribuem dinheiro como característica principal. Alguns contratos certificam a propriedade de um ativo do mundo real, endossam seu valor e controlam as negociações. O projeto *Ethereum DAO* foi o mais representativo desta classe, até o seu colapso devido a um problema grave de segurança em junho de 2016, obrigando um *hard fork* para reembolso de valores pagos

indevidamente. Alguns contratos fornecem um seguro que cobre contratemplos provados digitalmente (por exemplo, *Etherisc* vende apólices de seguro para voos; se um voo atrasar ou for cancelado, obtém-se um reembolso). Outros contratos publicam mensagens publicitárias (por exemplo, *PixelMap* é inspirado na *Million Dollar Homepage*).

Os contratos classificados como Notariais exploram a imutabilidade da blockchain para armazenar alguns dados persistentemente e, em alguns casos, para certificar sua propriedade e procedência. Alguns contratos permitem que os usuários gravem o *hash* de um documento na blockchain, para que possam provar a existência e integridade do documento. Outros permitem declarar direitos autorais sobre arquivos de artes digitais, como fotos ou música.

A categoria jogo reúne contratos que implementam jogos de azar (por exemplo, loterias, dados, roleta, etc.) e *games*. A categoria Carteira (*wallet*) trata de chaves, envia transações, gerencia dinheiro, implanta e monitora contratos, a fim de simplificar a interação com o blockchain. Por último, na categoria Biblioteca os contratos implementam operações de propósito geral (como, por exemplo, transformações matemáticas e manipulação de *strings*), para serem usadas por outros contratos.

### 4.2.3. Internet das Coisas (IoT)

A Internet das Coisas é composta por dispositivos físicos com funções de rede, componentes micro-computadorizados e itens incorporados com funções de conectividade [Atzori 2016]. Junto com os rápidos avanços em software e hardware, as tecnologias relacionadas à IoT são indispensáveis na sociedade moderna.

Tais dispositivos permitem o monitoramento de ambientes industriais, domésticos e públicos, através de câmeras de vigilância, sensores, atuadores e/ou monitores. Isto permite o desenvolvimento de novas aplicações que exploram uma quantidade de dados muito grande gerados por esses dispositivos [Díaz et al. 2016, Mehmood et al. 2017].

Estas aplicações estão pulverizadas em áreas com diferentes propósitos. Por exemplo, um grande número de empresas de eletrônicos está projetando dispositivos para casas inteligentes [Stojkoska and Trivodaliev 2017]. As cidades inteligentes são uma realidade, oferecendo mais conforto e conveniência ao público [Mehmood et al. 2017]. De modo geral, para implementar estas soluções, a utilização de serviços de middleware abstraem as dificuldades de acesso aos dispositivos devido a heterogeneidade de protocolos e interfaces. A Figura 4.1 ilustra uma organização geral de elementos para Internet das Coisas [Sztajnberg et al. 2018]. Os sensores e atuadores são agrupados em dispositivos e serviços, que por sua vez precisam das interfaces de comunicação para enviar/receber dados das camadas superiores.

Existe uma variedade de protocolos e padrões de comunicação que podem ser empregados, entre eles, destacam-se o MQTT [Mqtt 2017], CoAP [Shelby et al. 2014], AMQP [Vinoski 2006], XMPP [Saint-Andre et al. 2004], WebSocket, REST e Lorawan [Sornin et al. 2015], etc.

As práticas deste minicurso utilizam o protocolo REST. Ele permite o uso da infraestrutura do HTTP para acionar ou obter recursos, apenas dando uma interpretação diferente para os métodos GET, PUT, POST e DELETE, e valendo-se da possibilidade de



Figura 4.1. Organização de elementos na IoT. Fonte [Sztajnberg et al. 2018]

enviar informações adicionais numa mensagem HTTP [Sztajnberg et al. 2018].

#### 4.2.4. Blockchain para IoT

Blockchains para IoT são sistemas de blockchain personalizados e otimizados para aplicações de IoT. Estas aplicações são desenvolvidas em muitos campos. No entanto, boa parte desses aplicativos possui problemas como vazamento e confiabilidade de dados. Para mitigar esses efeitos problemáticos, a blockchain pode ser usada para fornecer maior segurança e estabilidade nos aplicativos IoT tradicionais [Lao et al. 2020].

Os dispositivos de IoT possuem uma série de limitações relativas à memória, processamento, potência e comunicação que precisam ser ponderados antes de se aplicar uma camada de blockchain. Nos últimos anos, muitas pesquisas foram publicadas com tentativas de abordar este casamento de forma que seja possível adotá-lo no mundo real [Bahga and Madisetti 2016, Sharma et al. 2017, Huh et al. 2017].

Segundo [Lao et al. 2020], as aplicações de blockchain e IoT podem ser categorizadas em 3 grupos: (a) pagamento digital, (b) serviço de contratos inteligentes, e, (c) armazenamento.

A categoria de pagamentos digitais foi a primeira e mais ampla utilização no campo da blockchain. Atualmente blockchains como Bitcoin e *Ethereum* podem ser utilizadas em *smartphones*, os quais funcionam como dispositivos capazes de processar (e armazenar localmente) uma parte da cadeia de blocos, o que ajudou a popularizar a manipulação de criptomoedas em dispositivos móveis.

A categoria dos contratos inteligentes emprega esta tecnologia para construir sistemas de automação e controle com base na IoT, eliminando a terceira parte de confiança [Christidis and Devetsikiotis 2016]. Há muitas empresas que fornecem este serviço, como por exemplo, a LeewayHertz <sup>1</sup> é uma empresa que fornece soluções para *startups* de IoT

<sup>1</sup>[www.leewayhertz.com](http://www.leewayhertz.com)

e empresas usando o contrato inteligente *Ethereum*. A Ecotrace <sup>2</sup>, empresa brasileira que atua na área de rastreabilidade de alimentos emprega blockchain, inteligência artificial e IoT para unir os elos da cadeia produtiva usando a blockchain *Hyperledger*.

A última categoria, a de armazenamento, aplica-se a aplicativos de armazenamento de dados que vêem a blockchain como um banco de dados seguro e distribuído. A Factom <sup>3</sup> é um desses exemplos, que utiliza APIs (*Application Programming Interface*) bem definidas para persistir informação de plataformas Web tradicionais na blockchain.

Apesar dos evidentes benefícios proporcionados pela integração tecnológica entre blockchain e IoT, há muito ainda a ser feito. Uma das iniciativas promissoras é a blockchain IOTA <sup>4</sup>, que promete alta escalabilidade, ausência de protocolos de consenso baseado em provas e promessa de realização de transferências quase instantâneas a custo zero [Silvano and Marcelino 2020].

### 4.3. Arquitetura da Blockchain e de Aplicações Blockchain-IoT

A tecnologia blockchain envolve muitos elementos além de simplesmente conectar blocos em uma cadeia. Ao adicionar elementos de IoT, esta complexidade ganha novos contornos que precisam ser desvendados. Esta composição blockchain-IoT é discutida em [Lin and Liao 2017, Zheng et al. 2017, Wu et al. 2019, Lao et al. 2020], que servem de base para esta seção, subdividida em duas partes: 4.3.1. Arquitetura blockchain, 4.3.2. Arquitetura para aplicações blockchain-IoT.

#### 4.3.1. Arquitetura Blockchain

A blockchain, segundo [Wu et al. 2019], tem uma arquitetura dividida em quatro camadas: (i) Dados, onde se encontram os blocos, o armazenamento de dados e a estrutura de árvore utilizada; (ii) Rede, onde se encontra a rede P2P e os mecanismos de comunicação; (iii) Consenso, onde naturalmente estão os protocolos de consenso; e, (iv) Aplicação, onde estão os contratos inteligentes, as criptomoedas e as *sidechains*.

A Figura 4.2 ilustra esta divisão. Na camada de dados estão a estrutura, organização e armazenamento de dados. Fatores como desempenho e acesso são cruciais para a rede blockchain. Cada uma destas redes utiliza estruturas diferentes. De um modo geral, o banco de dados para armazenamento é o Google LevelDb. A rede Bitcoin usa a árvore de *Merkle* como forma de organizar e armazenar as informações, enquanto a *Ethereum* usa a *Merkle Patricia Tree*.

A *Merkle Patricia tree* fornece uma estrutura de dados autenticada criptograficamente que pode ser usada para armazenar as ligações (chave, valor). Elas são totalmente determinísticas, o que significa que uma *tree Patricia* com as mesmas ligações (chave, valor) tem a garantia de ser exatamente a mesma até o último byte e, portanto, ter o mesmo *hash* de raiz com eficiência  $O(\log(n))$  para inserções, pesquisas e exclusões.

A Camada de rede da blockchain é autonomamente mantida e gerenciada por uma rede P2P composta por mineradores e usuários. É uma estrutura descentralizada e sem

---

<sup>2</sup><https://ecotrace.info/>

<sup>3</sup>[www.factom.com](http://www.factom.com)

<sup>4</sup>[www.iota.org](http://www.iota.org)

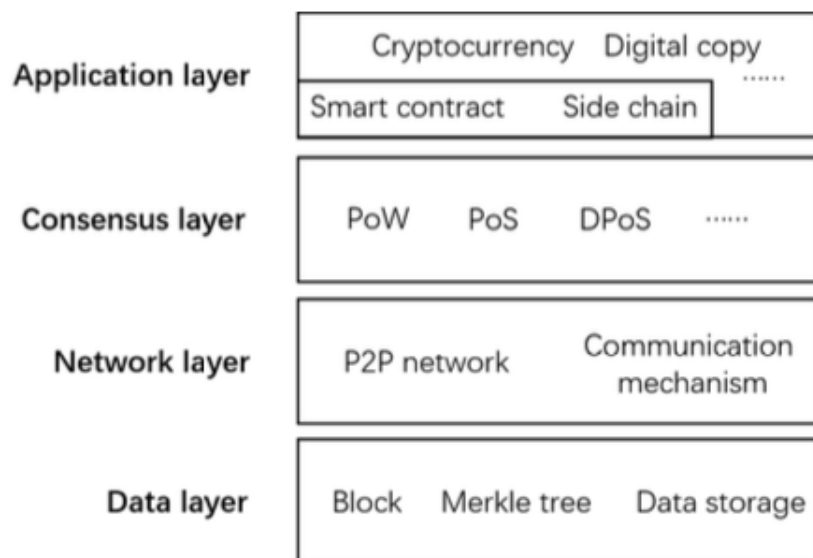


Figura 4.2. Arquitetura blockchain em quatro camadas. Fonte [Wu et al. 2019]

a necessidade de controle de entrada e saída, com tolerância a falhas. A comunicação e a autenticação devem ser protegidas em caso de ataques.

Na plataforma *Ethereum*, existe uma variedade de redes baseadas em conformidade com a especificação formal definida em [Buterin et al. 2014], mas que podem ou não interoperar umas com as outras. Entre elas, estão a própria *Ethereum*, *Ethereum Classic*, *Ella*, *Expanse*, *Ubiq*, *Musicooin*, etc. Embora haja compatibilidade a nível de protocolo, estas redes costumam ter recursos ou atributos que exigem que os mantenedores do software cliente *Ethereum* façam pequenas alterações para dar suporte a cada rede. Por causa disso, nem todas as versões do software cliente *Ethereum* executam todas as blockchains baseadas em *Ethereum* [Antonopoulos 2017].

Atualmente, existem seis implementações principais do protocolo *Ethereum*: (1) *Parity*, escrito em *Rust*; (2) *Geth*, escrito em *Go*; (3) *cpp-ethereum*, escrito em C++; (4) *pyethereum*, escrito em Python; (5) *Mantis*, escrito em *Scala*; (6) *Harmony*, escrito em Java.

Os nós completos da *Ethereum* podem ajudar outros novos nós a obterem os dados do bloco para inicializar sua operação, além de oferecer ao operador uma verificação autorizada e independente de todas as transações e contratos. Para isto exige-se largura de banda e hardware especializados.

Os clientes remotos *Ethereum* não armazenam uma cópia local da blockchain nem validam blocos ou transações. Eles oferecem a funcionalidade de uma carteira eletrônica, podem criar e também transmitir transações. Os clientes remotos mais comuns são o *MetaMask*, *Emerald Wallet*, *MyEtherWallet* ou *MyCrypto*.

Há diferenças entre o cliente remoto e a carteira. Normalmente, o cliente remoto oferece a funcionalidade de transação de uma carteira e uma API (como *web3*, descrita em 4.5.4. Outro conceito que merece atenção é o de uma carteira remota no *Ethereum*

como de um cliente leve ( análogo a um cliente de Verificação de Pagamento Simplificado em Bitcoin). Os clientes leves validam cabeçalhos de bloco e usam provas *Merkle* para validar a inclusão de transações no blockchain e determinar seus efeitos, dando-lhes um nível de segurança semelhante a um nó completo. Por outro lado, os clientes remotos *Ethereum* não validam cabeçalhos de bloco ou transações. Eles confiam inteiramente em um cliente completo para lhes dar acesso ao blockchain e, portanto, perdem garantias significativas de segurança e anonimato [Antonopoulos 2017].

A rede *Ethereum* é composta da rede principal e de redes de teste, estas últimas utilizadas como ambientes de estudos e pesquisa para desenvolvimento. A rede principal, endereçável na porta TCP 30303 trabalha com *ethers* que precisam ser comprados com dólares e as transações sofrem consequências reais. As redes de teste trabalham com *ethers* que não possuem valor real e que podem ser adquiridos em geradores de *ethers* na Internet sem custos financeiros. A rede *Ropsten* é uma rede de teste pública de blockchain. A rede de teste *Kovan* é uma rede de teste pública que usa o protocolo de consenso Aura com prova de autoridade (Esta é uma rede permissionada). A rede de teste *Rinkeby* utiliza o protocolo de consenso "*Clique*" com prova de autoridade (Esta também é uma rede permissionada).

Além disto, existe a opção de uma rede *localhost* 8545 que se conecta a um nó em execução no mesmo computador que o navegador, usando uma blockchain privada local como a *Ganache*, descrita em 4.5.4. A opção *Custom RPC* permite conexão a qualquer nó com uma interface de Chamada de Procedimento Remoto (RPC) compatível com *Geth*.

A camada de consenso é fundamental em uma rede blockchain. Chegar a um consenso não é uma tarefa trivial e muitos algoritmos de consenso foram propostos para atingir esse objetivo. Esses algoritmos ou mecanismos podem ser classificados em: PoW, PoS e suas variantes; BFT (do inglês *Byzantine Fault Tolerance*) e suas variantes. Esta camada é tratada com detalhes em 4.4

Por fim, a camada de Aplicação estende a capacidade do blockchain e torna mais fácil para os desenvolvedores construir aplicativos blockchain através dos contratos inteligentes, explicados anteriormente, das *sidechains* e do emprego de BaaS (do inglês *Blockchain as a Service*) [Samaniego et al. 2016], por exemplo. Nesta camada também estão as criptomoedas e uma série de aplicações incluindo *Fintechs*, seguros, pagamentos, governo, etc. Existe inclusive a possibilidade de ser combinada com inteligência artificial, big data, Computação quântica, IoT etc. As aplicações na plataforma *Ethereum* serão detalhadas em 4.5.

### **4.3.2. Arquitetura para Aplicações Blockchain-IoT**

A arquitetura de aplicações para blockchain-IoT é composta de 5 camadas: Física, Rede, Blockchain, Middleware e Aplicação [Lao et al. 2020]. A Figura 4.3 ilustra estas camadas.

A camada física da arquitetura blockchain-IoT é a mesma que a camada física da IoT [Lee and Lee 2015]. Inclui os sensores, atuadores, dispositivos inteligentes, etiquetas RFID, telefones celulares, câmeras de monitoramento e quaisquer outros dispositivos de IoT relacionados à aplicações blockchain-IoT. Os protocolos empregados também são os



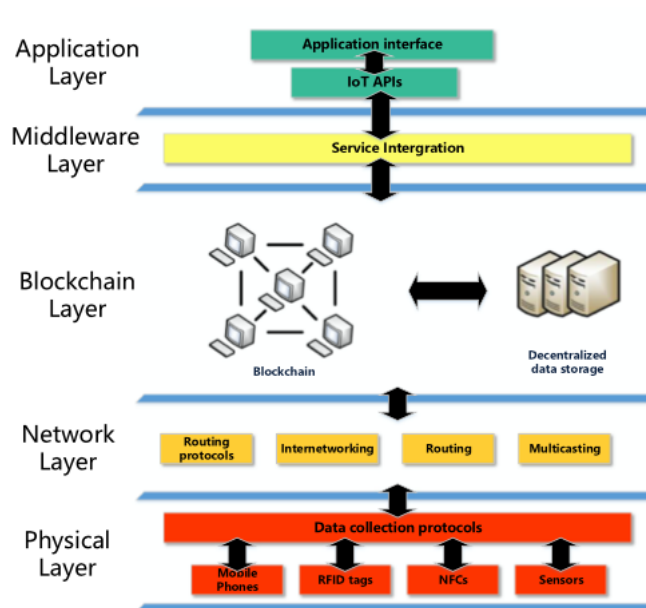


Figura 4.3. Arquitetura blockchain-IoT em 5 camadas. Fonte [Lao et al. 2020]

mesmos usados em IoT, como MQTT, COAP, REST, etc.

A camada de rede agrega funções de roteamento, interconexão de redes e *multicasting* [Jiang et al. 2016]. Esta camada é muito similar à camada de rede tradicional da blockchain. A camada blockchain representa as funções de consenso, armazenamento de dados e compartilhamento de dados, podendo inclusive ser uma plataforma de blockchain customizada [Nakamoto 2008, Underwood 2016, Ethereum 2014].

A camada de middleware é responsável por gerenciar a integração de IoT com blockchain e o fornecimento de serviço de segurança adicional [Alphand et al. 2018]. A camada de aplicativo é semelhante ao sistema IoT e às arquiteturas blockchain tradicionais. Para isto são fornecidas interações, serviços de abstração e APIs para os usuários [Dorri et al. 2017].

Para este tipos de aplicação, a blockchain é vista como um banco de dados seguro e distribuído, protegido contra violação de dados e ataques maliciosos, suportando, quando possível, contratos inteligentes que podem recepcionar os dados gerados pela IoT, mantendo um registro confiável e sem a necessidade de elementos certificadores. Isto permite uma variedade de aplicações e de soluções, nas mais diversas áreas do conhecimento, antes inimagináveis.

A Tabela 4.1 compara algumas arquiteturas blockchain-IoT, informando as características na camada de aplicação, middleware, blockchain, rede e física. A última coluna da tabela classifica as arquiteturas nos grupos de aplicação específica e blockchain como serviço.

As arquiteturas de aplicações específicas são relativas a softwares ou sistemas comerciais que usam IoT e blockchain como partes essenciais em suas operações. O *Smart Home* possui na camada física diversos dispositivos de IoT registrados na rede

**Tabela 4.1. Arquiteturas de Blockchain-IoT. Fonte [Lao et al. 2020]**

Blockchain-IoT	Aplicação	Middleware	Blockchain	Rede	Física	Classe
Smart Home	Smart Home App	Gerenciamento	Blockchain comercial	P2P	Dispositivos inteligentes	Aplicação específica
LO3 Energy	Energy Shopping	Energy Token	Blockchain pública	Rede de baixa latencia	Painéis solares	Aplicação específica
Slock.it	DApp	Não usa	Ethereum	Rede comercial	Travas eletrônicas	Aplicativo específica
Hybrid-IoT	Aplicação IoT	Plataforma Hybrid-IoT	POW blockchain, BFT blockchain	P2P	Sensores	Aplicativo como serviço
BP2IoT	DApp	C	Blockchain	P2P	Dispositivos de IoT	Aplicativo como serviço
JD.COM	JD.com	Blockchain Gateway	BFT blockchain	P2P	Dispositivos de IoT	Aplicativo como serviço
IoT Data Service Framework	Aplicação para Usuários	Framework	Ethereum	P2P	Dispositivos de IoT	Aplicativo como serviço
IoT Chain	Acesso com Autorização	Framework	Ethereum	Rede Comercial	Dispositivos de IoT	Aplicativo como serviço

blockchain através de uma estrutura LSB (*Lightweight Scalable Blockchain*) que garante a segurança e privacidade [Dorri et al. 2019].

LO3 Energy <sup>5</sup> oferece energia solar usando uma rede P2P. A camada física é composta por painéis solares. Eles capturam a geração de energia e remetem para a blockchain usando uma *token* própria (*Exergy Token*) e uma rede de baixa latência. Os clientes compram a energia por meio de aplicativos.

O “Slock.it” <sup>6</sup>, adquirido pela Blockchains Company, controla fechaduras eletrônicas que são desbloqueadas através de um *token*. A arquitetura é simples e consiste em aplicativos distribuídos, blockchain *Ethereum*, rede comercial e bloqueios eletrônicos.

Os aplicativos como serviço são softwares de suporte que se conectam em um sistema blockchain-IoT. Tendo como camada central um middleware, estes aplicativos integram dispositivos IoT e blockchain com uma plataforma de gerenciamento simples para desenvolvedores. Rotinas como verificação de contratos e de informações precisam estar disponíveis e acessíveis nas camadas de blockchain e rede.

O “Hybrid-IoT” [Sagirlar et al. 2018] é uma plataforma classificada nesta categoria que implementa consenso baseado nos algoritmos PoW e BFT. A plataforma para a IoT aplicada à indústria BP2IoT [Bahga and Madiseti 2016] permite a criação de DApps. Os dispositivos IoT precisam ser registrados na rede blockchain.

A plataforma de blockchain “*JD Blockchain Open Platform*” <sup>7</sup> fornece serviços de *gateway*, de nó e de consenso na blockchain. Baseada em consenso BFT, possui protocolo de autenticação para controlar o número de acessos à rede blockchain.

Uma estrutura para implementar integridade e segurança de dados de IoT com base na plataforma *Ethereum*, onde o dispositivo de IoT é responsável por gerar e gravar dados na blockchain, dispensando uma autoridade certificadora, e posteriormente, proporcionando ao usuário a verificação da integridade dos dados por meio de um aplicativo

<sup>5</sup><https://lo3energy.com/>

<sup>6</sup><https://www.blockchains.com/>

<sup>7</sup><http://ledger.jd.com/>

de usuário de dados é definida em [Liu et al. 2017].

A IotChain combina a arquitetura OSCAR [Vučinić et al. 2015] à estrutura de autorização ACE [Seitz et al. 2017]. Cada usuário registrado tem um *token* autorizado que identifica um conjunto de recursos. O dispositivo IoT é responsável pela geração de dados. O proprietário dos dados é responsável por enviar os dados para a blockchain, a arquitetura OSCAR e a estrutura de autorização ACE responsáveis por garantir a segurança dos dados do usuário [Alphand et al. 2018].

#### 4.4. Consenso

O consenso é um problema fundamental em computação distribuída e permite com que um conjunto de participantes (ou nós) numa rede chegue a um acordo sobre um conjunto de transações, ou sobre um determinado estado do sistema, apesar da ocorrência de falhas ou da presença de nós maliciosos, que podem subverter o sistema [Greve et al. 2018]. O consenso, portanto, mantém o estado consistente das réplicas e a disponibilidade do sistema. No contexto da IoT, o consenso da blockchain precisa ser bem elaborado para atender aos requisitos de falta de recursos (computacional, espaço, etc.). Desta forma, as exigências de aplicativos de IoT, com altos custos de manutenção e fraco suporte a usos de tempo crítico, podem ser resolvidas com a introdução de um mecanismo de consenso distribuído adequado.

Esta seção apresenta os protocolos de consenso Prova de Trabalho - PoW, PoS, Variantes do PoW e PoS, Consenso Tolerante à Falhas Bizantinas - BFT e Grafo Direcionado Acíclico - DAG (*Directed Acyclic Graph*). Em seguida faremos um comparativo entre os protocolos de consenso, ilustrados na Figura 4.4 e abordamos as características de protocolos de consenso para IoT. O termo PoX (*Proof of Somethings*) é uma forma de referir-se genericamente aos protocolos que necessitam de alguma prova para alcançar o consenso [Lao et al. 2020].

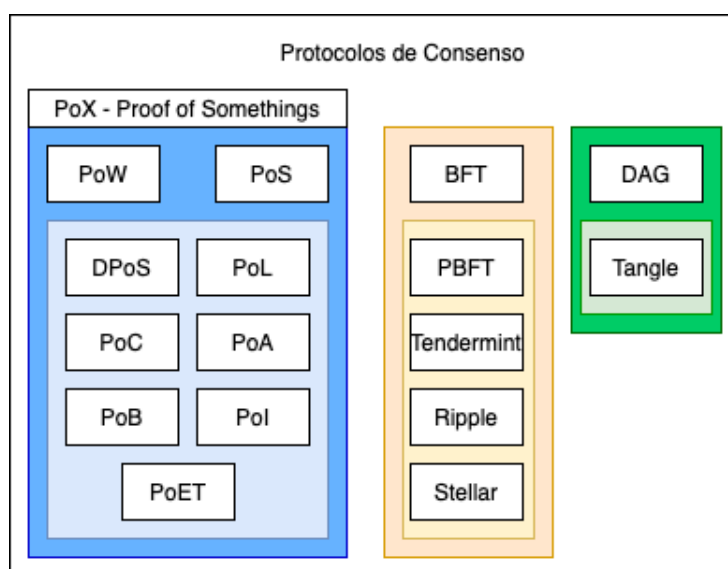


Figura 4.4. Diagrama com alguns protocolos de consenso.

#### 4.4.1. Prova de Trabalho - Proof of Work (PoW)

O protocolo de consenso PoW surge com a blockchain do Bitcoin, no famoso artigo [Nakamoto 2008]. Desde então, diversas variações apareceram. No geral, o PoW adota a seguinte estratégia: cada nó da rede precisa resolver um desafio computacional (um *puzzle* criptográfico) para poder propor à rede um bloco de transações. Assim, através de um mecanismo de competição, em um processo exaustivo, o nó que resolver o quebra-cabeça matemático obterá uma recompensa na forma de criptomoeda, o bitcoin. Após a formação do bloco, o nó irá encaminhá-lo à rede, e todos os nós irão agregá-lo a uma "blockchain", estrutura de dados contendo toda a cadeia de blocos até então acordada pelos nós, de tal forma que o bloco recentemente transmitido aponta para o anterior.

Desta forma, observa-se dois princípios básicos que fazem o consenso PoW funcionar [Lao et al. 2020]: (i) *A regra da cadeia mais longa*: o nó considera a cadeia mais longa como a cadeia certa. Isso porque, como mais de um nó pode resolver o puzzle ao mesmo tempo, mais de um bloco é transmitido na rede para estender a cadeia. Por princípio, os nós irão sempre estender a cadeia mais longa, e portanto, após um tempo, todos estarão com a mesma estrutura de blockchain, obtendo-se assim o acordo. (ii) *A regra de incentivo*: um nó será recompensado ao encontrar um bloco adequado. Desta forma, os nós estarão motivados a despendere recursos computacionais participando da competição (ou mineração de blocos). Essa estratégia mantém a rede disponível e operante.

Estas premissas são a base de funcionamento da rede Bitcoin. Elas garantem a exatidão e exclusividade da cadeia de blocos, evitando o duplo gasto e a manipulação da cadeia de blocos (ou livro razão) por um nó malicioso. Ainda, elementos criptográficos são adicionados a esse processo para garantir a segurança, privacidade e integridade dos dados. Evidentemente, há outros desafios a serem tratados em um sistema complexo que movimenta ativos digitais tão valiosos, como as propriedades da criptomoeda, onde se emprega criptografia de chave assimétrica. Para uma melhor base sobre esses elementos de segurança, recomendamos o livro [Narayanan et al. 2016].

Sagirlar et al. [Sagirlar et al. 2018] propõem o uso da blockchain na IoT com a intenção de alcançar um sistema IoT baseado em consenso distribuído adequado que supere as desvantagens. No sistema, de nome Hybrid-IoT, os subgrupos de dispositivos IoT formam blocos de blocos PoW, chamados de sub-blocos de PoW, conforme ilustrado na Figura 4.5. Em seguida, a conexão entre as sub-blockchains PoW emprega uma estrutura interconectora de consenso bizantino BFT, como a Polkadot<sup>8</sup> ou Cosmos<sup>9</sup>. Os autores propõem a formação de sub-blockchains PoW guiadas por um conjunto de diretrizes baseadas em parâmetros de dimensões, métricas e limites. Para comprovar a validade da abordagem, realizaram uma avaliação de desempenho e de segurança.

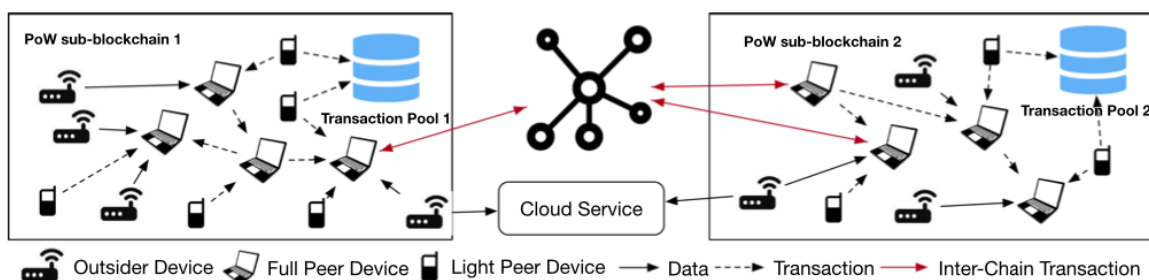
#### 4.4.2. Prova de Participação - Proof of Stake (PoS)

O PoS [Kiayias et al. 2017] é um dos algoritmos em ascensão para muitas aplicações de blockchain. Após anos de uso do PoW, algumas desvantagens ficaram evidentes, como segurança (ataques de duplo gasto possíveis), alto consumo energético e desperdício de recursos (no processo de competição/mineração) ou baixa vazão - *throughput* (pouca quan-

---

<sup>8</sup><https://polkadot.network>

<sup>9</sup><https://cosmos.network/>



**Figura 4.5. Arquitetura do Hybrid-IoT com emprego de PoW e o conceito de sub-blockchains. Fonte [Sagirlar et al. 2018]**

tidade de transações acordadas no tempo).

De forma simplista, o PoS baseia-se na hipótese de que os usuários com a posse de mais moedas (ou recursos computacionais) são mais propensos a garantir a confiabilidade do sistema e têm menos probabilidade de se comportar como nós maliciosos. Assim, o algoritmo PoS considera a porcentagem do número total de moedas (ou recursos) que um nó envolvido na competição detém, e eventualmente considera o tempo que o nó leva com o montante de moedas (ou recursos) para estabelecer uma porcentagem de direito de participação no consenso. Quanto mais moedas (ou recursos), mais probabilidade o nó terá de participar do consenso para decidir sobre os blocos.

A fim de permitir que cada bloco seja gerado mais rapidamente, o mecanismo PoS elimina o processo exaustivo de resolução do *puzzle* criptográfico. Mas, há problemas que também tonaram-se ou podem se tornar evidentes, como aconteceu com o PoW. Por exemplo, os usuários com mais moedas por um longo período têm maior possibilidade de serem selecionados pelo sistema para gerar o próximo bloco, ocasionando um elitismo e centralização das decisões. Existem muitas plataformas de blockchain que adotam o PoS, como a Cardano <sup>10</sup>, Algorand <sup>11</sup> e a *Ethereum 2.0* <sup>12</sup>, que passa a adotar o PoS, devendo herdar boa parte das aplicações com IoT que usam contratos inteligentes dessa rede, num futuro próximo.

#### 4.4.3. Variantes do PoW e PoS

Apesar de eficientes, os protocolos baseados em PoW e PoS tem alguns pontos negativos, como por exemplo, a alta concentração do controle dos nós que possuem mais recursos, o alto consumo energético e as constantes atualizações de hardware. Desta maneira, alguns protocolos surgem em resposta a estes desafios. Alguns deles são apresentados a seguir.

#### Prova de Participação Delegada - DPoS (*Delegated Proof of Stake*)

O DPoS [Larimer 2014] resolve o problema de centralização através da introdução do mecanismo de delegação. Os nós da rede elegem nós especiais, os super nós ou delegados, que passam a gerar e assinar os blocos. Com esta estratégia, o tempo de se confirmar uma transação melhora, pois o protocolo DPoS elimina a necessidade de se aguardar a confirmação de nós não confiáveis. Parece um paradoxo tentar centralizar de-

<sup>10</sup><https://cardano.org/>

<sup>11</sup><https://www.algorand.com/>

<sup>12</sup><https://ethereum.org/en/eth2/>

cisões em um sistema descentralizado, no entanto qualquer nó pode ser alçado à condição de delegado. Quando um deles viola quaisquer regras do protocolo, seus direitos são negados e outro delegado será eleito.

Como exemplo de uso do DPoS temos as plataformas *BitShares*<sup>13</sup>, *Steem*<sup>14</sup> e *EoS*<sup>15</sup>, que apresentam 101, 21 e 21 delegados, respectivamente. Comparado com o PoW, o DPoS é mais rápido e eficiente. Além disso, é mais democrático e flexível do que o PoS. Os delegados podem ser selecionados de forma justa e os delegados ruins podem ser expulsos rapidamente, caso haja uma boa governança. As desvantagens são que o DPoS não é tão descentralizado quanto o esperado para os delegados confiáveis existentes. Além disso, as pessoas relutam em votar para indicar delegados, a menos que haja incentivos e, caso não haja votos suficientes, grandes interessados podem facilmente dominar a votação para obter benefícios [Wu et al. 2019].

### **Prova de Sorte - PoL (*Proof of Luck*)**

Este protocolo emprega funções TEE(*Trustworthy Execution Environment*) para fornecer justiça de mineração, segurança de tempo e certeza da identidade do nó. Surge como uma alternativa ao PoW, que exige cada vez mais poder de processamento e consumo de energia. Através da geração de um número aleatório executado em um TEE, estas funções bloqueiam as plataformas que podem ser usadas para processamento das operações como forma de limitar o poder desigual da computação. Após a geração do número, o PoL escolhe um líder para o consenso, e, com isto, obtém-se economia no consumo de energia, baixa latência para confirmação de transações e equidade na mineração [Milutinovic et al. 2016].

### **Prova de Capacidade ou Prova de Espaço - PoC (*Proof of Capacity* ou *Proof of Space*)**

O PoC [Dziembowski et al. 2015] usa o espaço disponível no disco rígido para definir privilégios ao invés do poder computacional dos nós concorrentes. A probabilidade de propor um bloco é proporcional ao espaço de armazenamento cedido à rede por um nó minerador. Quanto maior a capacidade de armazenamento em disco, maior a probabilidade de decidir sobre um bloco no consenso.

### **Prova de Atividade - PoA (*Proof of Activity*)**

Os algoritmos de PoA contam com um conjunto de N nós confiáveis chamados de autoridades. Cada autoridade é identificada por um único id e a maioria delas é considerada honesta, ou seja, pelo menos  $N / 2 + 1$ . As autoridades chegam a um consenso para ordenar as transações emitidas pelos clientes. O consenso em algoritmos de PoA depende de um esquema de rotação de mineração, uma abordagem amplamente usada para distribuir de forma justa a responsabilidade da criação de blocos entre as autoridades. O tempo é dividido em etapas, cada uma das quais tem uma autoridade eleita como líder de mineração [De Angelis et al. 2018].

As principais implementações de PoA são o *Clique*<sup>16</sup> e o *Aura*<sup>17</sup>. Ambas têm um

---

<sup>13</sup><https://wallet.bitshares.org/#/>

<sup>14</sup><https://steem.com/>

<sup>15</sup><https://eos.io/>

<sup>16</sup><https://eips.ethereum.org/EIPS/eip-225>

<sup>17</sup><https://openethereum.github.io/Aura>

primeiro turno onde o novo bloco é proposto pelo líder atual (proposta de bloco); então o *Aura* requer uma nova rodada (aceitação do bloco), enquanto o *Clique* não.

### **Prova de Queima - PoB (*Proof of Burn*)**

Neste protocolo de consenso, os mineradores devem comprovar que queimaram algumas moedas, enviando-as para alguns endereços onde não podem ser gastos. A quantidade dessas moedas destruídas determina a probabilidade de um minerador emitir um novo bloco. O PoB funciona como uma espécie de mineração virtual, queimando moedas virtuais [Frankenfield 2018].

### **Prova de Importância - PoI (*Proof of Importance*)**

Empregando o conceito de importância, este protocolo consegue medir a capacidade de uma conta de minerar um bloco. Para isto, a quantidade de moedas que possui e o número de transações realizadas são considerados. Há uma certa similaridade com o PoS, sob o ponto de vista do saldo em criptomoedas quando se observa o uso do saldo como critério decisivo para eleger nó, no entanto há de se observar que no PoI também se considera o volume de transação realizada. Para minerar um bloco os nós devem realizar transações ativamente. Ao aplicar PoI, a blockchain ganha vantagens de eficiência energética e alta taxa de transação [Bach et al. 2018]. Um exemplo de aplicação deste protocolo é o NEM <sup>18</sup>.

### **Prova de Tempo Decorrido - PoET (*Proof of Elapsed Time*)**

O algoritmo de consenso PoET [PoET 2018] faz com que os nós eleitos estocasticamente aguardem um tempo de espera aleatório criado pelo sistema. O nó que primeiro esgotar o tempo será eleito o líder para a criação do novo bloco. O PoET é um algoritmo semelhante a uma loteria que atende à justiça, ao investimento e à verificação. Para evitar trapaças, dois requisitos precisam ser verificados: O primeiro é que o líder realmente espera por um tempo aleatório em vez de um curto período de tempo para vencer. O segundo é que o líder realmente espera pelo tempo de espera determinado pelo protocolo.

#### **4.4.4. Tolerância a Falhas Bizantinas - BFT(*Byzantine Fault Tolerance*)**

Os protocolos baseados em BFT pertencem a uma classe que conseguem obter um acordo em um sistema, onde os processadores podem falhar de forma arbitrária, denominado Problema Geral Bizantino [Lamport et al. 1982]. A seguir, define-se os seguintes protocolos baseados em BFT: PBFT, *Tandermint*, *Ripple* e *Stellar*.

#### **PBFT (*Practical Byzantine Fault Tolerance*)**

O PBFT [Castro et al. 1999] foi o primeiro algoritmo prático a tolerar falhas bizantinas e adaptou-se para ser usado em ambientes assíncronos. O *BFT-Smart* é um outro projeto promissor em bom estágio de maturidade [Bessani et al. 2014]. O PBFT é oferecido pelo Hyperledger Fabric como camada de acordo (ordenação de transações). Além disso, o *BFT-Smart* [Sousa et al. 2018] também foi recentemente incorporado ao projeto [Greve et al. 2018].

---

<sup>18</sup><https://nem.io/>

### ***Tendermint***

O *Tendermint* [Kwon 2014] é um protocolo de consenso BFT quase assíncrono, baseado em validadores, propondo blocos de transações e votando neles. Ele requer apenas duas rodadas de votação para chegar a um consenso. Em cada rodada, há três etapas (ou seja, propor, prevenir, pré-comprometer). Quando mais de 2/3 dos votos pré-comprometidos forem recebidos para alcançar o consenso em uma rodada, o consenso para a próxima rodada começará.

### ***Ripple***

O *Ripple Protocol Consensus Algorithm* (RPCA) [Todd 2015] utiliza sub-redes confiáveis coletivamente dentro da rede maior para chegar a um consenso para o Problema Geral Bizantino. No Ripple, a Unique Node List (UNL) é um conjunto de outros servidores mantidos por cada servidor, que desempenha um papel importante quando um servidor faz consultas para determinar o consenso. Apenas os votos dos servidores na UNL são considerados na determinação do consenso. Esta é uma diferença óbvia de muitos algoritmos de consenso. A UNL representa um subconjunto da rede que exige sabedoria coletiva para chegar a um consenso. A premissa da RPCA é que cada servidor confia nos outros servidores da UNL e acredita que eles não entrarão em conluio. A RPCA procede em várias rodadas para chegar a um consenso. Em cada rodada, cada servidor primeiro coleta o máximo de transações para se preparar para o consenso e torná-las públicas na forma de “conjunto de candidatos”. Em seguida, cada servidor faz uma união dos conjuntos candidatos dos servidores em seu UNL e vota em cada transação. De acordo com o resultado da votação, as transações que obtiverem votos abaixo de um percentual mínimo serão descartadas ou colocadas em candidatos definidos no próximo consenso para o próximo bloco do livro-razão, enquanto aqueles que obtiverem votos suficientes irão para o próximo turno [Wu et al. 2019]

### ***Stellar***

O *Stellar Consensus Protocol* (SCP) [Mazieres 2015] é um protocolo do acordo bizantino federado (FBA). Ele é considerado o primeiro mecanismo de consenso comprovadamente seguro a desfrutar simultaneamente de quatro propriedades principais: controle descentralizado, baixa latência, confiança flexível e segurança assintótica. Segurança assintótica significa que a segurança do SCP depende de assinaturas digitais e famílias de *hash* cujos parâmetros podem ser ajustados de forma realista para proteger contra adversários com um poder de computação inimaginavelmente vasto.

#### **4.4.5. Grafo Direcionado Acíclico - DAG (*Directed Acyclic Graph*)**

Existe uma categoria de protocolos, a exemplo do Dagcoin [Lerner 2015] e do Tangle [Popov 2018], que apresentam uma estratégia visando explorar o paralelismo do sistema tradicional de blockchain de cadeia única, e empregam uma estrutura de dados de grafo direcionado acíclico para conectar blocos. O mecanismo de consenso, distinto dos demais abordados anteriormente, consiste em que cada transação fique vinculada aos dois registros de transações anteriores através do grafo. Desta forma, a conformidade da transação atual pode ser comprovada referenciando-se às transações anteriores. Se comparado com outros protocolos de consenso que estabelecem algum tipo de prova, observa-se que o



DAG preocupa-se apenas com as transações vinculadas, constituindo um modo bem mais simples do que as diferentes provas a que se submetem os nós. O IOTA é uma plataforma de blockchain que através do Tangle utiliza este conceito.

#### **4.4.6. Comparativo entre os Protocolos de Consenso**

Nas seções anteriores, descreveu-se alguns algoritmos de consenso mais populares e atuais em blockchain. Aqui é feita uma comparação desses algoritmos, ilustrada na Tabela 4.2. A linha Tipo de Blockchain, classificada em permissionada e não permissionada, pode também ser classificada em pública, consórcio e privada. A blockchain de consórcio é gerenciada por algumas organizações ou institutos e permitem que as pessoas participem. Ao custo de alguma descentralização, a blockchain de consórcio pode fornecer a eficiência e a segurança da blockchain pública, mantendo algum controle central, monitoramento e proteção. O Hyperledger é um dos aplicativos de blockchain de consórcio mais famosos. Na blockchain privada, as permissões são até mesmo limitadas e concentradas em um indivíduo ou em uma organização. Portanto, em alguns casos, algumas regras ou mesmo dados podem ser modificados ou adulterados pela autoridade. Comparado com a blockchain pública e a blockchain de consórcio, a blockchain privada ganha as vantagens em alta velocidade e taxa de transferência. A blockchain privada é frequentemente aplicada a empresas individuais [Wu et al. 2019].

Pode-se observar, que de modo geral, os protocolos baseados em BFT tem o desempenho prejudicado à medida que a rede aumenta a quantidade de nós. Isto ocorre devido ao custo adicional de comunicação entre os nós novos e os existentes. No entanto há vantagens de extensibilidade, já que se pode combinar com vários tipos de algoritmos de melhoria para atender a necessidades específicas [Lao et al. 2020].

As blockchains baseadas em protocolos do tipo PoX são indicadas para projetos públicos, porque a exigência de poder computacional ou moeda é útil no sentido de prevenir ataques de negação de serviço, abuso de serviço e tornar a cadeia mais segura e confiável, mas por outro lado, este conjunto de protocolo também sofre de ineficiência e alta sobrecarga computacional.

O DAG é uma nova tendência em sistemas de blockchain. Ele consegue atingir alto rendimento, típico de protocolos BFT e características de flexibilidade dos protocolos PoX. Isto implica em um mecanismo de consenso com pouca exigência computacional e altos rendimentos de eficiência. No entanto, a aplicação do DAG não está madura. Por ser um protocolo novo, ainda são necessárias mais pesquisas para que se possa de fato comprovar sua aplicação e descobrir mais sobre possíveis vulnerabilidades. O IOTA tem se mostrado um ambiente adequado para este processo de validação.

Como pode-se observar, cada um destes protocolos possui vantagens e desvantagens, que permeiam critérios como desempenho, escalabilidade, segurança, poder computacional, eficiência energética, etc. Cabe ao pesquisador ou engenheiro definir o escopo de sua aplicação, e com base nisto, escolher aquele que ofereça o melhor custo x benefício.

**Tabela 4.2. Comparação dos principais protocolos de consenso. Fonte:[Wu et al. 2019, Lao et al. 2020]**

Ano	PoW 1999	PoS 2012	DPoS 2014	PoC	PoA	Pol	PoB	BFT	PBFT 1999	Ripple 2012	Stellar 2015	Tendermint 2014	DAG 2015
Tipo de blockchain	Não permissionada	Não permissionada	Permissionada	Não permissionada	Não permissionada	Não permissionada	Não permissionada	Não permissionada	Permissionada	Permissionada	Permissionada	Permissionada	Privada
Conceito Chave	Poder Computacional	Participação, idade da moeda	Delegados, Participação					Réplica	Baixa	UNL	Quórum	Validadores	
Latência	Alta	Baixa	Baixa					Alta	Baixa	Baixa	Baixa	Baixa	Baixa
Troughput	Baixo 7 tps	Baixo 100 tps	Alto 100.000 tps	Baixa	Baixo	Baixo	Baixo		100s tps	Alta 1500 tps	Alta 3000 tps	Alta 10.000 tps	Alta 800 tps
Finalização da transação	Probabilística	Probabilística	Probabilística	Probabilística	Probabilística	Probabilística	Probabilística	Determinístico	Imediata	Determinística	Imediata	Imediata	Probabilística
Mineradores?	Sim	Sim	Sim						Não	Não	Não	Não	Não
Token	Sim	Sim	Sim						Não	Não	Não	Não	Não
Eficiência energética	Não	Sim	Sim	Sim	Sim				Sim	Sim	Sim	Sim	Sim
Tolerância a invasão	50% do poder computacional	50% de participação	<50%	50% de espaço	50% participação online	50% Participação	50% de moedas	33% de falha das réplicas	33% de falha das réplicas	20% de falhas dos nós		33% falhas	33% poder computacional
Escalabilidade	Baixa	Baixa	-	Baixa	Alta	Baixo	Baixo	Baixa	Baixo	Alta	Alta		Alta
Vantagem	Livre adesão ao consenso	Eficiência em Energia	Incremento de Transações	Eficiência em Energia	Eficiência em Energia	Menos chance de acumulação	Incentivo a longo prazo	Baixo custo de transação	Finalização de bloco de alto rendimento	Rápida finalização do Bloco		baixo custo de transação	Alto Throughput
Desvantagem	Baixa vazão Consumo energia Alta taxa forks	Overhead na Comunicação	-	Desperdício de espaço em disco	Cenários de aplicativos limitados	Requisitos de Confiança	Baixa latência de confirmação	Centralização de despesas gerais	Centralização de despesas gerais	Requisitos de confiança		Centralização de despesas gerais	Overhead
Vulnerabilidade	Mineração egoísta	Ataque de Longo Alcance	-	Mineração Egoísta	Simple ponto de falha	Simple ponto de falha	Ataque de negação de gastos	33% de ataque	33% de ataque	Simple ponto de falhas		33% de ataque	Ataque Sybil
Aplicações	Bitcoin Ethereum Peercoin Litecoin Dogecoin	Ethereum Percoin NXT, NVC Cosmos coin	Bitshares Ark	IPFS	Decred	NEM	XCP	Tendermint	Hyperledger	Ripple	Stellar	Cosmos Network, Tendermint	

#### 4.4.7. Características de Protocolos de Consenso para IoT

Os dispositivos de IoT possuem limitações computacionais, energéticas e restrições de armazenamento de dados. Os protocolos de consenso precisam, além destas características, de um ambiente distribuído para garantir validade e consistência. Atingir este equilíbrio é o desafio para tornar este casamento duradouro e estável. A alta eficiência energética e um processo de consenso leve podem atenuar estes problemas.

Aplicações que envolvem blockchain-IoT precisam então driblar tais limitações e herdar os benefícios da blockchain. Se as restrições dos dispositivos de IoT forem premissas, nem os clientes leves nem os mineradores são indicados.

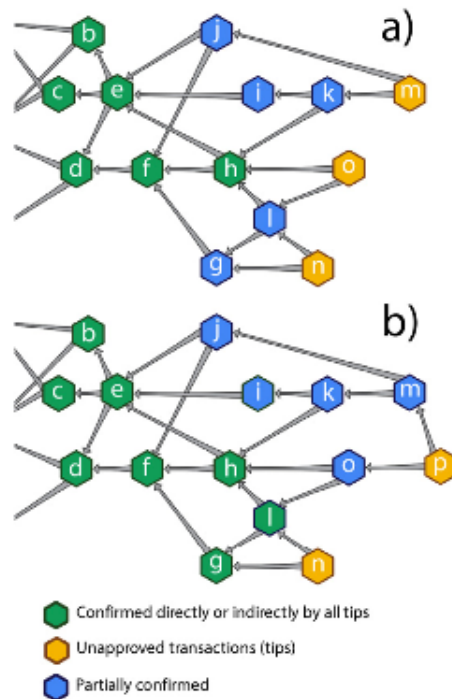
A blockchain IOTA é um exemplo de plataforma desenvolvida para IoT. Ela adota o consenso Tangle, baseado em DAG. Esta rede não possui mineradores, portanto o consumo energético é mais eficiente. Cada nó participante desta blockchain que necessita criar/enviar transações, primeiramente deve participar ativamente do processo de consenso aprovando duas transações anteriores.

A rede no grafo Tangle é composta por nós que são entidades que emitem e validam transações, e cada nó também representa uma transação [Popov et al. 2020]. Para que um nó adicione uma transação à rede, primeiro ele deve escolher duas transações para que possa aprová-las, conforme o algoritmo *Markov Chain Monte Carlo* (MCMC); Em seguida, o nó verifica se as transações escolhidas estão em conflito. Se isto ocorre, o nó deve desaprovar as transações conflitantes, e assim prevenir o gasto duplo; o próximo passo é resolver uma espécie de prova de trabalho (PoW), encontrando um valor *nonce*, tal que, seu *hash* seja concatenado com alguns dados da transação aprovado. É necessário destacar que este esforço computacional é uma versão muito mais leve do que a realizada pelos protocolos PoW tradicionais; Após conseguir este valor, o usuário envia sua transação para a rede, e ela se torna um *tip* (transação não aprovada); Por fim, o *tip* aguarda a confirmação por meio de aprovação direta ou indireta até que seu peso acumulado atinja o limite predefinido.

As transações são assíncronas e os nós não veem o conjunto de todas as transações. A Figura 4.6 ilustra um processo de confirmação. Em (a), o nó "m" não conhece o histórico dos ramos que contêm as transações "o", "n", "l" e "g", mas ainda validou as transações "k" e "j".

Ser assíncrono implica na possibilidade da existência de transações conflitantes. É possível que a transação "k" entre em conflito com "g" porque ambas não foram validadas em conjunto.

Quando a transação "p" valida "m" e "o", em 4.6(b), cada transação vinculada a elas é verificada, direta ou indiretamente, incluindo "k" e "g". Neste momento, se houver o conflito, ele é identificado e é necessário escolher um ramo. Segundo Popov [Popov et al. 2020], a principal regra usada para decidir entre transações conflitantes é executar o algoritmo de seleção várias vezes, identificando qual das duas transações tem maior probabilidade de ser indiretamente aprovada pelo *tip* escolhido, assim garante-se a escolha do ramo com maior probabilidade [Silvano and Marcelino 2020], enquanto o outro é abandonado.



**Figura 4.6.** Grafo ilustrando um estado do Tangle. Em (a) um estado aleatório. Em (b) um estado após uma transação *p*. Fonte: [Silvano and Marcelino 2020]

A rede *Ethereum*, embora empregue o consenso PoW, e em breve deva migrar para o PoS é usada por Atonomi [Atonomi 2019] e consegue manter um ecossistema saudável que permite a desenvolvedores e fabricantes de IoT um ambiente de confiança universal. Através da blockchain, a aplicação valida a identidade imutável dos dispositivos de forma interoperável. O sistema *Ethereum* é detalhado na Seção 4.5.

## 4.5. Aplicações

A blockchain é amplamente utilizada em campos como indústria, governo, saúde, logística, comércio, etc. Esta seção descreve a evolução da blockchain desde o seu lançamento até os dias atuais, classificando as aplicações em sidechains e contratos inteligentes. Ao final, será demonstrada uma aplicação de IoT com a blockchain *Ethereum*, como forma de ilustrar os conhecimentos aqui abordados.

### 4.5.1. Evolução da Blockchain

Esta subseção analisa a evolução das redes blockchain desde a sua versão 1.0, originalmente concebida por Nakamoto para dar suporte à criptomoeda Bitcoin. Aqui será mostrada a evolução da blockchain bitcoin e da criação de novas redes como a *Ethereum*, *Hyper Ledger*, *Dash*, *Litecoin*, etc. Em seguida, analisa-se a Blockchain 2.0 cujo marco principal são os contratos inteligentes. A rede *Ethereum* aceita tais contratos através da EVM (*Ethereum Virtual Machine*). Ela funciona como uma Máquina de Turing completa, executando os contratos de forma determinística. Na sequência, a Blockchain 3.0 e as DApps (*Decentralized Applications*), que tem causado grandes impactos tanto na

indústria quanto na academia.

A Figura 4.7 ilustra a terminologia usada no ambiente computacional para identificar a evolução da blockchain e seus principais marcos. A primeira versão da blockchain tem como marco principal o lançamento do Bitcoin, em 2009. Com ela, a comunidade tomou conhecimento de um engenhosa combinação de técnicas como o uso de uma rede P2P, elementos criptográficos, livro-razão público e a eliminação da terceira parte de confiança sustentando as transações de uma criptomoeda, até então desconhecida e sem representatividade financeira.



**Figura 4.7. A evolução da blockchain. Fonte: [Wu et al. 2019]**

A rede P2P é composta por nós que disputam o direito de publicar um bloco com as transações feitas pelos usuários e premia o vencedor com novas moedas recém emitidas. Através deste mecanismo de incentivo, do livro razão distribuído e da prova de trabalho, era possível então manter o estado da rede e evitar possíveis fraudes. Isto serviu de inspiração para pesquisas e novas aplicações, inclusive o lançamento de novas moedas digitais.

O *Litecoin* apareceu como uma das primeiras alternativas ao Bitcoin. Em 2011, a mineração de Bitcoin exigia um hardware cada vez mais especializado e caro, tornando difícil para as pessoas comuns minerarem Bitcoin. O *Litecoin* é basicamente o mesmo que o Bitcoin em tecnologia, mas é mais leve. O algoritmo Litecoin tenta permitir que qualquer pessoa com um computador comum participe do processo de mineração.

Outras moedas digitais também ganham seus próprios recursos. Por exemplo, *Dash* é uma moeda digital que suporta transações instantâneas e protege a privacidade do usuário. Ela é baseada em Bitcoin e possui uma rede única de camada dupla que inclui mineradores e nós mestres. Ela melhora o Bitcoin em dois aspectos principais: velocidade de transação e anonimato. Esta tecnologia de pagamento permite que as transações sejam concluídas quase que instantaneamente e usa técnicas de combinação de moedas para garantir a privacidade das transações [Wu et al. 2019].

A rede *Ethereum*, lançada em 2015, implementou os contratos inteligentes, permitindo transacionar, em suas redes, programas de computador, além do *ether*, que é a sua criptomoeda. Tais contratos são os representantes da Blockchain 2.0, ampliando a funcionalidade da rede. Na blockchain *Ethereum*, as pessoas também podem minerar para obterem criptomoedas recém emitidas, pois o mecanismo base foi importado do Bitcoin.

O núcleo desta plataforma é a Máquina Virtual Ethereum, que pode realizar a codificação de algoritmos complexos. O *Ethereum* é adequado para a construção de aplicativos que interagem automaticamente e diretamente entre pares ou que facilitam as atividades de coordenação de aplicativos P2P. Em teoria, qualquer atividade ou transação financeira complexa pode ser codificada de forma automática e confiável no *Ethereum*.

Além dos aplicativos financeiros, a Internet das coisas será fortemente beneficiada e influenciada, pois possui cenários que requerem uma camada de segurança, principalmente nos dados gerados pelos dispositivos.

O projeto *Hyperledger* também é uma blockchain representante do ciclo 2.0. Composta por núcleos técnicos, Capítulos e apoio da *Linux Foundation*, esta blockchain é composta de uma biblioteca de código-fonte aberta, permitindo que os interessados criem soluções personalizadas. É uma comunidade crescente, com implementações relevantes, entre elas a *Hyperledger Sawtooth* e a *Hyperledger Fabric*.

A terceira versão da blockchain é marcada pelo avanço das DApps. Estas aplicações descentralizadas podem ser definidas como um aplicativo que é maioritariamente ou totalmente descentralizado [Antonopoulos 2017]. Esta possibilidade de desenvolver aplicativos que se apóiam em um sistema distribuído, em especial a blockchain e os contratos inteligentes, causam um enorme impacto na economia e na academia. Enquanto que no primeiro eles modificam boa parte das relações contratuais vigentes, entregando a um sistema computacional a decisão e execução de determinadas ações, no segundo é uma fonte de pesquisa e estudos para criar novos conhecimentos, melhorar o desempenho, segurança e ampliar os horizontes para os mais diversos domínios. A literatura está repleta de publicações que abrangem a tecnologia, finanças econômicas, contabilidade, impostos e regulamentação, saúde, etc.

A blockchain X.0 sugere a continuidade desta evolução. A próxima sessão irá detalhar as categorias de aplicação para a Blockchain 3.0, com ênfase em aplicações que envolvem IoT.

#### 4.5.2. Categorias de Aplicações para Blockchain

As aplicações que envolvem blockchain podem ser agrupadas em *sidechains*, que permitem ao desenvolvedor anexar recursos à rede principal através de uma cadeia separada, e os contratos inteligentes, que são programas de computador representando interesses contratuais distintos executados na blockchain.

As *sidechains*, diferente dos *forks* que atualizam a blockchain, foram propostas em 2014 para que fosse possível transferir bitcoins e outros ativos entre várias blockchains [Back et al. 2014]. Ainda existem desafios técnicos para a implementação de *sidechains* [Croman et al. 2016] como por exemplo a coordenação do poder de mineração entre as cadeias, a pressão adicionada na cadeia principal e os efeitos na escalabilidade e alta latência.

Embora as *sidechains* sejam conectadas à blockchain, elas permanecem isoladas, e caso algum problema ocorra na *sidechain*, apenas ela fica comprometida, não interrompendo a cadeia principal. Alguns exemplos são: O Lisk Restaurante <sup>19</sup> permite que os clientes peçam comida online através de uma *sidechain* exclusiva com tipos específicos de transações personalizadas para restaurantes [Alves 2021]. O *Loom* <sup>20</sup> é uma plataforma para rodar DApps e jogos em *sidechains* conectados à *Ethereum*. O *POA Network* <sup>21</sup> é

---

<sup>19</sup>[www.liskrestaurant.com:3000/History](http://www.liskrestaurant.com:3000/History)

<sup>20</sup><https://loomx.io/>

<sup>21</sup><https://www.poa.network/>

uma *sidechain* *Ethereum* pública de código aberto para o desenvolvimento de contratos inteligentes que usa PoA. O *Liquid*<sup>22</sup> é uma *sidechain* comercial que permite a movimentação instantânea de fundos entre as *exchanges*. Por último, o *Root Stock-RSK*<sup>23</sup> é uma *sidechain* de código aberto atrelada à rede principal do Bitcoin para a execução de contratos inteligentes.

Os contratos inteligentes são sistemas que movem ativos digitais automaticamente de acordo com regras pré-especificadas. Normalmente são escritos em uma linguagem de alto nível, como Solidity. Os contratos só são executados se forem chamados por uma transação. Um contrato pode chamar outro contrato que pode chamar outro contrato e assim por diante, mas o primeiro contrato nesta cadeia de execução sempre terá sido chamado por uma transação de uma conta de usuário.

As transações dos contratos inteligentes, independente da cadeia de execução, são sempre atômicas. Quaisquer mudanças no estado global (contratos, contas, etc.) são efetivadas apenas se toda a execução for encerrada com sucesso. Se a execução falhar devido a um erro, todos os seus efeitos (mudanças no estado) são “revertidos” como se a transação nunca tivesse sido executada. Uma transação com falha ainda é registrada como tentativa e terá de pagar as taxas de execução [Antonopoulos 2017].

Como as aplicações blockchain-IoT passam por contratos inteligentes, mais detalhes e exemplos encontram-se descritos na seção seguinte.

#### 4.5.3. Aplicações que Utilizam Blockchain

Esta subseção apresenta aplicações que usam a blockchain. O objetivo é exemplificar a variedade de aplicações e suas relações com a blockchain. Segundo [Wu et al. 2019], é possível agrupar os contratos inteligentes conforme ilustrado na Figura 4.8, com grifo nosso, para o grupo de IoT. Em cada um dos grupos, cita-se ao menos uma referência da área.

**Governança Corporativa:** A Governança corporativa, as preocupações atuais e a evolução com relação à adoção da tecnologia blockchain nas áreas de serviços financeiros e governança corporativa e pública são discutidas em [Almatarneh 2020]. O autor avalia os riscos e benefícios da utilização de contratos inteligentes e avalia sua adequação (em termos de transparência, prestação de contas, responsabilidade e justiça), evidenciando as questões regulatórias. Este ensaio avalia as implicações potenciais dessas mudanças para administradores, investidores institucionais, pequenos acionistas, auditores e outras partes envolvidas na governança corporativa. Em [Yermack 2017], o custo mais baixo, a maior liquidez, a manutenção de registros mais precisa, a transparência de propriedade oferecidos por blockchains e suas consequências são demonstradas.

**Bancos:** A referência [Dashkevich et al. 2020] apresenta os tipos de casos de uso por Bancos Centrais considerados para adaptação com a blockchain, listando pesquisas em moeda Digital emitida por Bancos Centrais; Conformidade Regulatória; Sistemas de Compensação e Liquidação de Pagamentos operados por bancos centrais; transferência de propriedade de ativos e auditoria. Em [Guo and Liang 2016] os autores afirmaram que a

---

<sup>22</sup><https://blockstream.com/liquid/>

<sup>23</sup>[www.rsk.co](http://www.rsk.co)

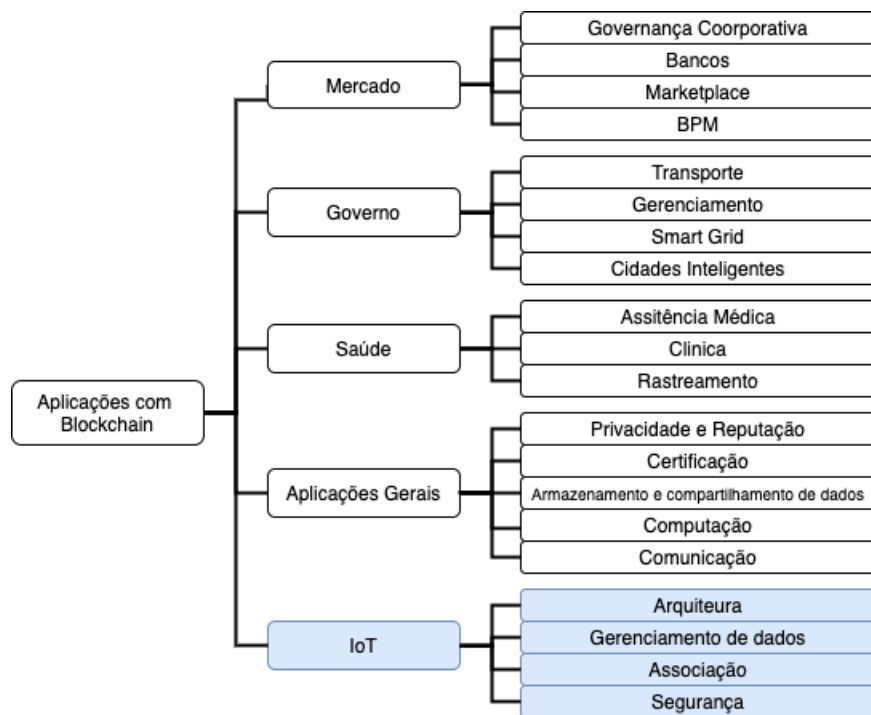


Figura 4.8. Classificação de aplicações que usam blockchain. Fonte:[Wu et al. 2019]

blockchain pode revolucionar as técnicas de pagamento e os sistemas de informação de crédito nos bancos. Em [Cocco et al. 2017] aponta-se os desafios e oportunidades de implementação da tecnologia blockchain em todos os bancos e como blockchain pode lidar com os processos financeiros como o Bitcoin.

**Marketplace:** O *Silk Road*, o primeiro mercado da *darknet*, pode ser um exemplo típico de aplicação de blockchain em *marketplace*. Em [Christin 2013], analisa-se e exemplifica-se como os itens eram vendidos e como a população de vendedores evoluiu ao longo do tempo. Uma proposta para facilitar a recuperação eficaz de dados e transações automáticas entre vários provedores e consumidores de dados em um sistema *marketplace* está em [Yoo and Ko 2020].

**BPM (Gerenciamento de Processos de negócios, do inglês *Business Process Management*):** Mendling et al [Mendling et al. 2018] descreveu os desafios e oportunidades de blockchain para gerenciamento de processos de negócios (BPM) para simular pesquisas utilizando blockchain. O trabalho deles refletiu que não apenas a blockchain poderia ser usada no BPM estabelecido, mas também além dele. Eles também discutiram a tecnologia de blockchain e recursos de BPM em áreas como estratégia, governança, tecnologia da informação, pessoas e cultura

**Transporte:** Sistemas de transporte inteligente aliam a blockchain para tornar os dados imutáveis e tornarem os dados uma prova digital, empregando contratos inteligentes vinculados às seguradoras para coletar dados [Balasubramaniam et al. 2020].

**Gerenciamento:** O artigo [Hou 2017] considera a realidade na China e discute a aplicação da tecnologia blockchain no governo eletrônico chinês, descobrindo que a tecno-



logia blockchain pode trazer melhorias nos serviços governamentais, maior transparência e acessibilidade de informações do governo, desenvolvimento de compartilhamento de informações entre diferentes organizações e assistência na construção de um sistema de crédito individual, ressaltando as integrações com sistemas de gestão.

**Smart Grids:** A referência [Xu et al. 2017] propõe uma estrutura de gerenciamento de recursos baseada em blockchain para economizar o consumo de energia em *datacenters*. Alladi, em [Alladi et al. 2019] apresenta uma revisão de diversas aplicações para *smart grids* com uso de blockchain, entre elas o uso de energia pré-paga e o controle remoto de consumo de energia.

**Cidades Inteligentes:** O termo cidade inteligente envolve diversas aplicações em muitas áreas diferentes, em [Kuperberg et al. 2019] tem-se uma análise completa das oportunidades e do estado da arte da tecnologia blockchain e documentos de identidade eletrônica (eIDs) emitidos pelo governo de diversos países, incluindo implementações e provas de conceito existentes.

**Assistência Médica:** A assistência médica remota baseada em blockchains para sistemas de informação e assistência médica de tele-medicina é abordada em [Ji et al. 2018]. Uma aplicação baseada em blockchain que permite aos pacientes possuírem, controlar e compartilharem de forma fácil e segura seus próprios dados de saúde sem violar a privacidade para assistência médica é detalhada em [Yue et al. 2016]. Azaria et al. [Azaria et al. 2016] propôs o MedRec, um sistema distribuído de gerenciamento de registros para processamento de registros médicos eletrônicos usando a tecnologia blockchain. A referência [Ahmad et al. 2021] explora as oportunidades e os desafios de adaptabilidade para a tecnologia de blockchain no setor de teles-saúde e tele-medicina mostrando como a blockchain pode desempenhar para fornecer a segurança e privacidade das informações necessárias, transparência operacional, imutabilidade de registros de saúde e rastreabilidade para detectar fraudes relacionadas a pedidos de seguro de pacientes e credenciais médicas.

**Clínica:** Em [Shae and Tsai 2017] propõe-se uma arquitetura de plataforma de blockchain para ensaios clínicos e medicina de precisão de forma a permitir um ecossistema de dados médicos confiável para pesquisa colaborativa. A referência [Liu 2016] discute as vantagens e desvantagens das tecnologias de blockchain e big data para registros médicos.

**Rastreamento:** Um exemplo de rastreamento de dados médicos com o uso de blockchain integrado no rastreamento de um paciente desde a primeira visita a um médico de atenção primária, rastreando seus dados até o diagnóstico final, pode ser encontrado em [Angraal et al. 2017].

**Privacidade e Reputação:** Dennis e Owen [Dennis and Owen 2015] propuseram o primeiro sistema de reputação generalizado que poderia ser aplicado a várias redes baseadas em blockchain. O uso de blockchain para gerenciar reputação para identificar usuários mal-intencionados e proteger a privacidade dos usuários simultaneamente no cenário *crowdsourcing* com dispositivos móveis é explorada em [Zhang et al. 2020].

**Certificação:** Um esquema desenhado para construir um sistema de certificação descentralizado baseado em blockchain e contratos inteligentes com o objetivo de fornecer serviços de certificado blockchain para a competição de inovação e empreendedorismo de

estudantes universitários é explicado em [Xie et al. 2020]. A proposição de blockchain e certificado digital, junto com um protocolo de autenticação seguro para dados de privacidade em blockchains sem verificar a assinatura de identidade criptografada do terceiro participante é detalhada em [Liu et al. 2020].

**Armazenamento e compartilhamento de dados:** Chowdhury et al. detalha uma estrutura de compartilhamento de dados que garante a autenticidade dos dados compartilhados em tempo real e fornece privacidade transacional em uma rede blockchain, melhorando o processo de tomada de decisão e reduzindo o custo geral [Chowdhury et al. 2018].

**Computação:** BeCome usa a tecnologia blockchain na computação de ponta para garantir a integridade dos dados [Xu et al. 2019]. O artigo [Ikeda 2018] explica um novo sistema de informação que acomoda estados quânticos de forma ponto a ponto com o apoio da blockchain.

**Comunicação:** A referência [Chaer et al. 2019] discute oportunidades e desafios da rede 5G com a camada de blockchain empregada para segurança.

**Arquitetura:** Ta-Shma et al. [Ta-Shma et al. 2017] propõem uma arquitetura usando componentes *open source* otimizados para aplicativos de Big Data empregados em cidades inteligentes.

**Gerenciamento de dados:** Em [Park et al. 2018] está descrito um sistema para gerenciar dados no mercado de P2P baseado em contratos inteligentes *Ethereum*.

**Segurança de dados:** A plataforma *Ethereum* junto com um contrato inteligente é comparada com uma comunicação usando MQTT quanto ao seu nível de segurança e simulando ataques em [Fakhri and Mutijarsa 2018]. O uso de blockchain para segurança na comunicação é também aplicado na Indústria 4.0, dados marítimos e cidades inteligentes [Rathee et al. 2021, Rahimi et al. 2020, Biswas and Muthukkumarasamy 2016].

**Combinações de técnicas:** Uma nova Arquitetura Orientada a Serviços (SOA) baseada em uma blockchain semântica para registro, descoberta, seleção e pagamento é apresentado em [Ruta et al. 2017].

#### 4.5.4. Prática

Esta prática auxilia na criação de um ambiente de desenvolvimento e ilustra a interação entre um dispositivo de IoT ou um emulador que envia dados de temperatura e umidade e um contrato inteligente implementado na blockchain *Ethereum*. O material completo e um tutorial pode ser encontrado na página com informações complementares deste capítulo em [Abijaude et al. 2021].

Ainda não há um Ambiente de Desenvolvimento Integrado (IDE, do inglês *Integrated Development Environment*) ou um ambiente amigável para o desenvolvimento dos contratos e de DApps. Pode-se usar editores on-line, como por exemplo o Remix ou preparar um ambiente de desenvolvimento local. Esta última solução é a utilizada pelos autores e já testada em cursos na Universidade Estadual de Santa Cruz (UESC), Universidade Estadual do Sudoeste da Bahia (UESB) e Universidade Federal da Bahia (UFBA).

Para tanto, usa-se o o Node.js e um editor de código de sua preferência instalados

e configurados no computador. Após isto, deve-se instalar uma carteira para ter acesso à rede *Ethereum*. Aqui usamos a extensão Metamask.

Os seguintes pacotes adicionais para o Node.js precisam ser instalados: a) `solc`: compilador *Solidity*; b) `mocha`: *framework* para testar os contratos antes de implementá-los em uma rede BC; c) `web3`: coleção de bibliotecas que permite interagir com um nó *Ethereum* local ou remoto usando HTTP; d) `ganache-cli`: é uma BC pessoal para desenvolvimento rápido de aplicativos distribuídos *Ethereum* e *Corda* em um ambiente seguro e determinístico; e) `truffle-hdwallet-provider`: para realizar as assinaturas usando as palavras mnemônicas.

Estas palavras são informadas ao usuário no momento da instalação do metamask e devem ser guardadas, pois através delas conseguiremos autorizar as transações.

Com o ambiente pronto (na página do curso existe um tutorial completo para isto), deve-se criar uma pasta, fazer o download do projeto e descompactá-lo.

A pasta `deploySensor` possui o contrato inteligente (`Sensor.sol`, ilustrado na Figura 4.9) e os arquivos necessários para compilação (`compile.sol`) e implementação (`deploy.js`). O processo de compilação gera duas saídas: os bytecodes que precisam ser enviados para a rede blockchain e a ABI (do inglês *Application Binary Interface*) que precisa ser fornecida à aplicação, para que se tenha acesso aos contratos.

```
27     constructor(  
28         string memory _name,  
29         MeasureSetting[] memory _settings  
30     ) {  
31         require(_settings.length > 0, "Settings empty");  
32         owner = msg.sender;  
33         name = _name;  
34         for(uint8 i; i<_settings.length;i++){  
35             settings.push(_settings[i]);  
36         }  
37     }  
38     // Cadastra as medições  
39     function insertMeasure(Measure[] memory newMeasure) public {  
40         for(uint i; i < newMeasure.length; i++){  
41             require(newMeasure[i].value.length == settings.length,  
42                 "SettingsSize is different of new measure value");  
43             measures.push(newMeasure[i]);  
44         }  
45     }  
46     // Envia todas medições  
47     function getAllMeasure() public view returns (Measure[] memory measure) |  
48         return measures;
```

Figura 4.9. Parte do contrato `Sensor.sol`

O contrato `Sensor.sol` é responsável por receber e registrar medidas de temperatura e umidade conforme as Linhas 38 e 43. Ele armazena em uma matriz os valores de temperatura, umidade e timestamp enviados pelo sensor. A Linha 47 envia os valores que foram armazenados na matriz.

A pasta `simple-api` é composta de duas outras pastas: (a) `api` e (b) `frontend`. Em (a) temos o arquivo `server.js`, ilustrado na Figura 4.10, que cria uma interface

```

19 // Rota de inserção de dados
20 app.post("/insert-data", function (req, res) {
21     // Captura as variáveis no corpo da requisição
22     var { hash, temperature, humidity } = req.body;
23     // Captura o timestamp (data e horário) da requisição
24     const timestamp = new Date().getTime();
25
26     try {
27         // Compara o hash do sensor
28         if (hashSensor !== hash) throw { message: "Inválid Hash" };
29         // Verifica se os dados foram enviado
30         // Se for vazio retorna erro
31         if (!temperature || !humidity)
32             throw { message: "Empty temperature or humidity" };
33     }

```

**Figura 4.10.** Parte do arquivo `server.js` onde se implementa a URL para cadastro de dados pelo sensor

REST, simulando o middleware. Através dela conseguiremos enviar os dados para serem recepcionados, e em seguida enviados para o contrato inteligente.

A Linha 20 cria a rota `/insert_data` para receber os dados enviados pelo sensor. Na Linha 27 comparamos se o hash enviado pelo sensor é previamente conhecido para que os dados sejam aceitos.

Em (b) temos a pasta `src` onde estão os arquivos `sensor.js`, `web3.js`, `App.js` e `index.js`. O primeiro arquivo, `sensor.js` informa o endereço do contrato inteligente implementado na rede blockchain e a ABI gerada no momento da compilação do contrato. O arquivo `web3.js` cria uma instância da `web3` e a injeta na aplicação. Este passo é fundamental para que o arquivo `App.js` possa utilizar funções de manipulação do contrato.

```

3 // Endereço do contrato gerado no deploy
4 const address = "0xfc272950a29c2f2307174e82c07566c1b231c951";
5 // Abi gerada no deploy do contrato
6 const abi = [
7
8     inputs: [
9         {
10             internalType: "string",
11             name: "_name",
12             type: "string",
13         },

```

**Figura 4.11.** Parte do arquivo `sensor.js` responsável por informar o endereço do contrato e a ABI.

Na Figura 4.11, temos uma parte do código do arquivo `sensor.js`. Na Linha 4 é responsável por informar à aplicação qual o endereço do contrato para onde serão enviadas as requisições. O valor da variável `address` deve ser atualizado para o novo valor obtido após o processo de implementação de um novo contrato. O conteúdo da variável `abi` também deve ser atualizada pela nova ABI gerada no processo de compilação.

A Figura 4.12 mostra a parte do código onde a aplicação interage com o contrato inteligente. Na Linha 60 as contas ativas no metatask são recuperadas para fins de

identificação e cobrança. A Linha 63 aciona o método `insertMeasure` no contrato inteligente para registrar as medidas de temperatura, umidade e *timestamp*.

```
60 // Recupera as contas do metamask
61 const contas = await web3.eth.getAccounts();
62 // Insere medidas no contrato passando a temperatura, humidade e o timestamp
63 const leitura = await sensor.methods
64   .insertMeasure([
65     [
66       [data.temperature.toFixed(0), data.humidity.toFixed(0)],
67       data.timestamp,
68     ],
69   ])
70   .send({ from: contas[0] });
```

Figura 4.12. Código que demonstra a interação do sistema com um contrato inteligente.

É possível encontrar os tutoriais para compilação e implementação dos contratos; instalação e configuração da API que representa o middleware; e, instalação e configuração do front-end com a página da aplicação ilustrada na Figura 4.13.

Esta tela exibe os dados recebidos pelo sensor e os dados armazenados no contrato inteligente. Na parte superior são listadas as medidas recebidas pelos sensores. Ao clicar no botão **Sim!**, os dados são automaticamente enviados para a blockchain. Os endereços exibidos na tela representam a conta *Ethereum* de quem enviou a transação e o endereço da conta do contrato. O link "Veja a Transação no etherscan" permite ao usuário encontrar o registro de sua transação em um site especializado em manter o registro público de todas as transações da rede.

← → ↻ 🏠 ⓘ localhost:3000

📱 Apps 🔍 Buscador de artig... 📁 omnet 📁 latex 📁 Adm&Seg

## Contrato de sensor

id	Temperatura	Umidade	Data
1	23.40	93.32	14/05/2021 22:32:46
2	19.20	90.32	14/05/2021 22:33:09

Deseja enviar para o contrato a última visita de id: 2?

Endereço de envio: 0x1fa69f3ee1378b2159b0a852f6f29cb53e11d733

Endereço do Contrato: 0xfc272950a29c2f2307174e82c07566c1b231c951

[Veja a transação no ethersan](#)

## Medições do contrato

id	Temperatura	Umidade	Data
1	33.40	91.32	14/05/2021 9:29:35
2	38.40	89.32	14/05/2021 9:36:10
3	32.40	79.32	14/05/2021 9:43:39
4	23.40	93.32	14/05/2021 10:41:37
5	19.20	90.32	14/05/2021 22:33:09

Figura 4.13. Tela da DApp que envia/recebe dados do contrato.

Existe uma diferença entre as aplicações web tradicionais e as aplicações que envolvem blockchain.

Nas aplicações web tradicionais, conforme ilustrado na Figura 4.14, geralmente o usuário usa um servidor web para acessar uma página ou um serviço web, o qual eventualmente pode até consultar outros servidores em cadeia. Comumente, um usuário envia um POST para o servidor web que em seguida envia de volta os recursos solicitados.

Em contrapartida, com o uso de uma DApp e da blockchain este cenário muda um pouco. A Figura 4.15 exemplifica a prática, a qual é uma das situações possíveis. Em (1), um sensor envia para o middleware uma coleta de dados. O middleware, por sua vez, envia estes dados para a blockchain (2) para que os mesmos sejam armazenados em um contrato e aguarda a confirmação da transação em (3). Isto possibilita armazenar informações provenientes de IoT na blockchain, o que por conseguinte, permite a herança das propriedades de segurança a nível de aplicação.

O usuário pode acessar estes dados quando em (4) envia uma solicitação para o servidor web resgatar estes valores lidos pelos sensores e já armazenados na blockchain. Ao receber esta solicitação, o servidor web, através da ABI e das funções programadas solicita à blockchain que envie estes dados (5), os quais são exibidos para o usuário.

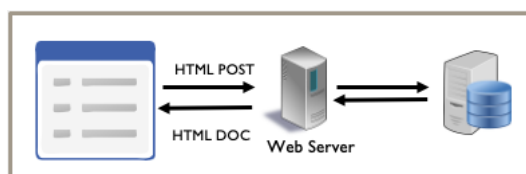


Figura 4.14. Arquitetura web.

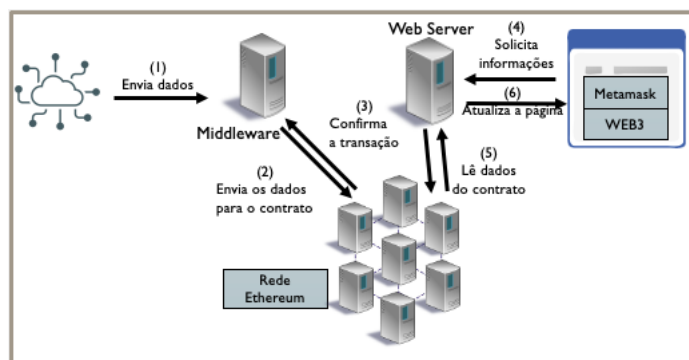


Figura 4.15. Arquitetura que exemplifica uma das formas de uma DApp trocar informações com a IoT.

Ao acessar a página do curso, teremos um tutorial completo para a execução da prática. Esta prática pode ser feita de duas formas: usando um dispositivo de IoT com interface de rede ou através de um programa que envie mensagens como o Postman ou Insomnia.

#### 4.6. Como Montar um Curso de Blockchain e IoT

A programação de aplicações para blockchain e IoT não é uma tarefa trivial. Envolve conceitos, propriedades e conhecimentos que vão além da blockchain, dos contratos inteligentes e da linguagem de programação *Solidity*.

Existe uma escassez de material teórico e prático para o ensino de tecnologias emergentes como blockchain, CIs e desenvolvimento de DApps. Uma alternativa para isto são as plataformas MOOC (*Massive Open Online Course*), como Udemy, Coursera, edX. Algumas Universidades promovem cursos de extensão ou treinamentos para seus alunos [Rao and Dave 2019, Dettling 2018, Araujo et al. 2019].

Os autores deste minicurso elaboraram e ministraram um curso em três universidades na Bahia: A Universidade Estadual de Santa Cruz (UESC), a Universidade Estadual do Sudoeste da Bahia (UESB) e a Universidade Federal da Bahia (UFBA), e com base nesta experiência, propõem um roteiro.

#### **4.6.1. Infraestrutura**

O hardware empregado para o desenvolvimento dos laboratórios é bastante simples, uma vez que não há plataformas que necessitem de muitos recursos computacionais. Nos 3 treinamentos ministrados havia computadores equipados com processadores que vão desde o Core 2 Duo com 4 Gb de RAM, até o Core i7 com 32 Gb de RAM. O espaço em disco também não é um fator limitante, uma vez que a maioria das máquinas possui espaço de armazenamento suficiente para os experimentos realizados.

O conjunto de softwares necessários à realização de todas as atividades práticas do treinamento é composto por navegadores, extensões para navegadores, ferramentas, pacotes e aplicativos hospedados em sites. Os softwares são compatíveis com praticamente todos os sistemas operacionais, como por exemplo Windows, Linux, Unix e MacOs.

#### **4.6.2. Conteúdo a ser Abordado**

Os cursos são compostos de três módulos. O primeiro, básico, aborda conceitos de BC, CIs e DApps, criando um contrato simples e uma DApp. O segundo, explora mais profundamente as funções da linguagem *Solidity*, construindo um contrato bem mais complexo, utilizando técnicas de engenharia de software, e criando uma DApp multipágina. O terceiro módulo serviu de base para a criação deste minicurso e prevê a integração com a Internet das Coisas, coletando dados dos sensores, armazenando-os em CIs e disparando ações quando determinadas situações forem alcançadas.

O curso básico foi ministrado com carga horária de 16h. Em 2 dias de aula, com 4 horas no período da manhã e 4 horas no período da tarde. No primeiro dia, durante o período da manhã foi abordada toda a parte teórica e conceitual da blockchain com ênfase na plataforma Ethereum, além da apresentação do editor de contratos on-line *Remix*. Durante o período da tarde, configuramos as máquinas locais e realizamos rotinas de testes.

No segundo dia, durante o período da manhã apresentamos mais um pouco de teoria com foco na interação com as redes *Ethereum*. Aprendemos mais um pouco sobre a linguagem de programação *Solidity* e escrevemos, compilamos, testamos e implementamos um contrato na rede *Rinkeby*. No período da tarde desenvolvemos uma DApp que interagia com o contrato implementado.

É possível encontrar mais recursos na internet, como por exemplo:

a) *CryptoZombies*: uma plataforma online onde o intuito é ensinar sobre contratos

inteligentes de forma interativa. O usuário desenvolve um jogo com foco em zumbis onde a logística é administrada por um contrato e com interação visual através de html, css e javascript. Disponível em <https://cryptozombies.io/pt/>.

b) *Ethernaut*: uma plataforma online que apresenta diversos tutoriais voltados para jogos. Ela foca puramente na criação de contratos, sem implementação visual. Alguns exemplos possibilitam a interação através da ferramenta do desenvolvedor do navegador. Disponível em <https://ethernaut.openzeppelin.com/>.

c) *Vyper Tutorials*: semelhante à ideia do CryptoZombies, esta plataforma propõe a criação de um jogo de *pokémon*. Atualmente está em fase de desenvolvimento, mas já é possível aprender como funciona a criação de contratos. Futuramente serão adicionadas interações através de interface assim como *CryptoZombies*. Disponível em <https://vyper.fun/#/>.

d) *Ethereum Studio*: uma ferramenta para desenvolvedores que desejam aprender sobre como construir aplicações na rede *Ethereum*. Os modelos ensinam como escrever um contrato inteligente, implementá-lo e interagir com os CIs por meio de um aplicativo baseado na web. Disponível em <https://studio.ethereum.org/>.

### 4.6.3. Práticas Propostas

Propõe-se dividir as práticas na exploração de um editor web e suas funcionalidades. Em seguida, a montagem de um ambiente local de desenvolvimento e explorando suas vantagens e desvantagens. Após isto, introduz-se o conceitos de testes para em seguida, construir contratos inteligentes mais sofisticados, integrados a uma DApp com apenas uma página web. Na sequência, aprofunda-se mais o desenvolvimento de contratos, usando-se por exemplo técnicas de engenharia de software e a integração com DApps multipáginas, para então finalizar com contratos interagindo com dispositivos IoT.

A primeira prática apresenta o Remix, o ambiente de desenvolvimento e o primeiro contrato. Este é um exemplo didático e serve para o aluno se familiarizar com o ambiente e começar a entender os conceitos de compilação, implementação, rede de testes, etc.

A segunda prática, recomenda-se que seja a montagem de ambiente local de desenvolvimento, ressaltando as vantagens e desvantagens desta abordagem. Nessa prática, o aluno vai lidar com scripts de compilação e implementação, compreender conceitos de ABI, bytewords, instalar uma carteira Ethereum e abastecê-la com ethers sem valor comercial.

O conceitos de testes de contratos e seus benefícios devem ser introduzidos como o terceiro momento da prática. É importante deixar claro que os contratos são imutáveis, e uma vez implementados não é possível modificá-los e nem mesmo apagá-los. Aqui o aluno aprende a configurar o ambiente de testes, montar uma rede blockchain local em sua máquina e executar testes básicos.

Na sequência, as práticas evoluem para contratos mais sofisticados, que representam exercícios mais elaborados e envolvem a transferência de moedas entre as contas. Novos testes também são propostos aqui e, ao final, a integração com um sistema Web simples, com apenas uma página, criando a primeira DApp.



Após a criação deste contrato, sugere-se criar contratos mais complicados, que envolvem conceitos de engenharia de software, como padrão *fabric*. Questões de quem vai pagar por eventuais operações nos contratos e recursos mais avançados da linguagem solidity serão tratados nesta prática. Depois de novas rotinas de teste, constrói-se uma DApp multipágina, que representa um sistema mais elaborado e com grau de dificuldade maior.

A última e mais desafiadora prática é construir um pequeno sistema que seja capaz de enviar dados de dispositivos IoT para um contrato e exibí-los em um sistema, empregando, por exemplo o estilo arquitetural REST. Caso não seja possível ter os dispositivos de IoT, pode-se optar pela geração de dados que simule tais equipamentos.

## **4.7. Desafios de Pesquisa**

Blockchain, Internet das coisas e contratos inteligentes são uma área em evolução e com muitos desafios de pesquisa em aberto, inclusive relacionados à segurança das aplicações. Estes desafios foram categorizados e discutidos nas próximas subseções.

### **4.7.1. Atividades Ilegais**

Uma blockchain pública é mantida por uma comunidade ao invés de autoridades. Sua natureza descentralizada também a torna uma plataforma ideal para atividades ilegais e sem censura. A economia baseada em criptomoedas facilita a lavagem de dinheiro e outras atividades ilegítimas, como a compra de bens ilícitos, compartilhamento de pornografia infantil e jogos de azar na Internet. Por exemplo, o site *Silk Road* era um mercado negro online com uma plataforma para vendedores e compradores fazerem, entre outras coisas, o tráfico ilegal de drogas [Matthews 2015]. Meiklejohn et al. [Meiklejohn et al. 2013] explora a heurística para agrupar carteiras de Bitcoin com base em evidências de autoridade compartilhada e, em seguida, usando ataques de re-identificação para classificar os operadores e quem busca usar Bitcoin para fins criminosos ou fraudulentos em grande escala.

### **4.7.2. Throughput de Transações**

Os principais fatores que afetam o *throughput* das transações na blockchain são a comunicação por *broadcast*, o mecanismo de consenso e a verificação da transação. A parte mais sensível entre eles é o mecanismo de consenso. Devido às características de confiança da blockchain, a garantia da exatidão e unicidade, exige-se que cada nó comprove trabalho suficiente para que possa expressar sua confiabilidade e autenticidade de sua própria mensagem. Embora o algoritmo PoW tenha resolvido o problema do gasto duplo, ele desperdiça muitos recursos e leva muito tempo para verificar as transações. Considerando que o número de aplicativos blockchain-IoT cresce a cada ano, novos mecanismos de consenso para aprimorar o desempenho da rede blockchain são necessários. Além disso, as soluções para resolver problemas de escalabilidade, capacidade de processamento ou armazenamento do dispositivo IoT na rede blockchain é um tópico que se mostra importante. Outra direção interessante de pesquisa futura é a análise do modelo de tráfego blockchain-IoT. No entanto, pode exigir uma grande escala de simulação, coleta de dados e análise de dados [Lao et al. 2020].

**Tabela 4.3. Tamanho esperado dos dados em transações com o aumento do throughput. Fonte [Wu et al. 2019]**

TPS	Tamanho do bloco	Taxas	Tamanho anual
1	0.3MB	0,12 BTC	15 GB
3	0.9MB	0,36 BTC	47 GB
10	3 MB	1.2 BTC	150 GB
100	30 MB	12 BTC	1.5 TB
1.000	300 MB	120 BTC	15 TB
10.000	3 GB	1.200 BTC	150 TB
100.000	30 GB	12.000 BTC	1.500 TB

#### **4.7.3. Problemas Relativos ao Armazenamento**

Na blockchain, todas as transações são registradas em blocos, e diferentes nós no sistema distribuído possuem uma cópia de toda esta informação. Isto pode impedir efetivamente que o livro-razão seja violado. Porém, com o acúmulo de transações e o crescimento dos dados, o espaço restante em cada bloco está ficando cada vez menor.

O volume de dados originados das transações pode impactar severamente no armazenamento e na capacidade da blockchain. Segundo [Wu et al. 2019], ao fazer uma estimativa simples, supondo que cada transação tenha 512 bytes e a unidade da taxa seja 0,0004 / KB, o tamanho dos dados da transação anual ficará muito caro para armazenar, conforme ilustrado na Tabela 4.3. De acordo com o registro da VISA em 2015, um total de 92.064 milhões de transações de pagamento foram geradas ao longo do ano. Se convertermos esse volume de dados de transações da rede em Bitcoins, o tamanho anual poderia ser 47 TB, o que está muito além da capacidade da máquina/banco de dados comum. Se aumentarmos o tamanho do bloco para 30 MB, o volume de dados de transação anual também pode ser de 1,5 TB, o que também é um número expressivo. Portanto, também é um problema muito desafiador expandir a capacidade da blockchain.

#### **4.7.4. Privacidade e Segurança**

Em blockchains públicas, cada participante pode obter um backup completo dos dados transacionados, que são abertos e transparentes, garantindo, desta forma, a confiabilidade da blockchain de uma certa maneira. No entanto, para muitos aplicativos de blockchain, esse recurso pode ser fatal. Em [Berdik et al. 2021] encontra-se um relatório abrangente sobre diferentes instâncias de estudos e aplicativos blockchain propostos pela comunidade de pesquisa e seus respectivos impactos na blockchain e seu uso em outros aplicativos ou cenários. O artigo [Leng et al. 2020] sob a perspectiva dos sistemas de informação aponta a nível de processo, de dados e de infraestrutura questões de segurança na blockchain, sugerindo direções futuras de pesquisa.

#### **4.7.5. Problemas Relativos à Combinação com IoT**

A integração das tecnologias necessárias à realização de aplicações envolvendo blockchain e IoT não é trivial. Os desafios mencionados nas subseções anteriores (atividades ilegais, *throughout* de transações, problemas de armazenamento e privacidade e segu-

rança) também podem se aplicar à integração com IoT, e alguns desafios ainda se tornam mais proeminentes neste contexto. Os principais problemas de segurança em relação à arquitetura em camadas da IoT, além dos protocolos usados para rede, comunicação e gerenciamento são catalogados e mostrados como a blockchain pode ser um habilitador-chave para resolvê-los [Khan and Salah 2018].

#### **4.7.6. Problemas Relativos à Combinação com Outras Tecnologias**

A blockchain possui potencial para ser aplicado em diversos campos, podendo assim ser combinada a várias outras tecnologias. Como exemplo, podemos citar a computação quântica, computação em nuvem e névoa, *Big Data*, inteligência artificial e realidade virtual. Para cada uma destas tecnologias há desafios de pesquisa em aberto.

Em relação à computação quântica, os protocolos criptográficos usados pela blockchain são suscetíveis a ataques pelo desenvolvimento de um computador quântico. Por exemplo, a segurança do Bitcoin será quebrada pelo enorme poder de computação dos computadores quânticos dentro de 10 anos. Tecnicamente, o esquema de assinatura da curva elíptica usado pelo Bitcoin poderia ser completamente quebrado por um computador quântico já em 2027 [Wu et al. 2019]. Em [Chen 2020] discute-se direções promissoras de criptografia, viabilidade da distribuição de chaves quânticas, gargalos encontrados pela rede de distribuição de chave quântica, conflitos simultâneos de links de retransmissão quântica em grande escala, atraso de retransmissão e acesso inconveniente a aplicativos que ainda não foram resolvidos completamente.

O uso de blockchain com Inteligência Artificial têm recebido grande parte da atenção das pesquisas nos últimos anos. Em [Ekramifard et al. 2020] são listados 23 artigos que demonstram desafios e aplicações inspiradoras que aumentam a segurança, a eficiência e a produtividade dos aplicativos.

Blockchain com computação em nuvem/névoa tem como uma grande dificuldade o estabelecimento de um ambiente seguro. Os dados na nuvem precisam ser transferidos pela Internet, então os desafios e comparação de vários problemas no ambiente de nuvem e problemas de segurança usando blockchain são relatados em [Pavithra et al. 2019].

O uso de blockchain com Realidade Virtual também possui desafios de pesquisa em aberto. Uma visão geral das oportunidades investigadas pelas soluções atuais combinando realidade virtual/realidade aumentada e blockchain é discutida, evidenciando oportunidades que poderiam promover a convergência dessas tecnologias e impulsioná-las em [Cannavo and Lamberti 2020].

Combinar *Big Data* com blockchain representa um grande potencial para melhorar os serviços e aplicativos de *big data*. Oportunidades de pesquisa e desafios em aberto para aplicativos de *big data* em diferentes domínios verticais, como cidade inteligente, saúde inteligente, transporte inteligente e rede inteligente são apresentados e discutidos para conduzir pesquisas adicionais nesta área promissora [Deepa et al. 2020].

#### **4.7.7. Padrões de Blockchain**

Algumas organizações ou institutos internacionais têm se dedicado ativamente ao estabelecimento de padrões de blockchain para regular e facilitar seu desenvolvimento. A

Tabela 4.4 mostra os padrões em desenvolvimento da ISO/TC (um comitê técnico internacional estabelecido em 2016, com o objetivo de padronizar blockchains e DLT's, do inglês *Distributed Ledger Technology*).

**Tabela 4.4. Padrões em desenvolvimento da ISO/TC 307**

Padrão	Objetivo	Estágio
ISO/AWI 22739	Terminologia	20.00
ISO/NP TR 23244	Visão geral da privacidade e proteção das informações de identificação pessoal	10.99
ISO/NP TR 23245	Risco de segurança e vulnerabilidades	10.99
ISO/NP TR 23246	Visão geral do gerenciamento de identidade usando tecnologias de blockchain e ledger distribuída	10.99
ISO/AWI 23257	Arquitetura	20.00
ISO/AWI TS 23258	Taxonomia e Ontologia	20.00
ISO/AWI TS 23259	Contratos inteligentes legalmente vinculativos	20.00
ISO/NP TR 23455	Visão geral e interações entre contratos inteligentes em blockchain e sistemas DLT	10.99
ISO/NP TR 23576	Segurança de ativos digitais	10.99
ISO/NP TR 23578	Problemas de descoberta relacionados à interoperabilidade	10.99

Além destes padrões, existem também:

- *Blockchain Reference Architecture*: Lançado pela China Blockchain Technology (CBT). Esta arquitetura divide a blockchain nas camadas de usuário, serviços, core e básica [Technology 2017b].
- *Blockchain–Data format specification*: Também lançado pela CBT. Este padrão organiza o formato dos dados, incluindo a estrutura e relacionamento, classificando os dados em seis categorias (dados da conta, do bloco, da transação, de entidades, dos contratos inteligentes e de configuração) [Technology 2017a]
- IEEE P2418: Padrão em desenvolvimento pela CTB para um framework de blockchain e IoT [Technology 2017c].
- *The Web Ledger Protocol 1.0*: Publicado pelo grupo de blockchain da W3C com o objetivo principal de proporcionar a flexibilidade e conectividade entre algoritmos [Sporny and D. 2017].

#### 4.8. Conclusão

Este capítulo conduziu um levantamento das principais pesquisas na área de Internet das Coisas, blockchain e contratos inteligentes, desenvolvendo conceitos iniciais para nivelamento, descrevendo a arquitetura básica, os protocolos de consenso, as aplicações de blockchain e IoT, a montagem de um curso para estas tecnologias, os principais desafios e uma prática.

A Seção 4.2 trouxe conceitos de blockchain, contratos inteligentes, Internet das coisas e blockchain para IoT como objetivo de nivelar os conhecimentos.

A Sessão 4.3 apresentou as arquiteturas de blockchain de aplicações blockchain-IoT, e apresentou uma tabela comparativa entre as aplicações que usam blockchain e IoT, contemplando a camada de middleware, plataforma de blockchain, camada de rede, camada física e a classe de aplicação.

A Seção 4.4 apresentou os protocolos de consenso, agrupando-os em PoX, BFT e DAG. Os protocolos PoX são divididos em dois grandes grupos (PoW e PoS) e suas variantes. Logo após, os protocolos da família BFT e DAG foram descritos. Esta sessão também apresentou uma tabela comparativa entre estes protocolos, salientando as suas respectivas vantagens, desvantagens e aplicações.

A Seção 4.5 apresentou um histórico da evolução da blockchain e seus principais marcos, seguidas de uma análise das aplicações possíveis para a blockchain, agrupadas em : Mercado, Governo, saúde, Aplicações Gerais e IoT. Para encerrar esta seção, os autores apresentaram a prática e a página do minicurso com informações complementares, além de um tutorial para a execução.

Direcionamentos de como implementar um curso que contemple o conteúdo deste minicurso foram abordados na Seção 4.6. Por fim, a Seção 4.7 apresentou os desafios de pesquisa atuais nesta área do conhecimento.

## Referências

- [Abijaude et al. 2021] Abijaude, J., Serra, H., Sobreira, P., and Greve, F. (2021). Mini-curso blockchain e contratos inteligentes para aplicações em iot, uma abordagem prática. <https://github.com/lifuesc/jai2021>. Acessado em 14/05/2021.
- [Ahmad et al. 2021] Ahmad, R. W., Salah, K., Jayaraman, R., Yaqoob, I., Ellahham, S., and Omar, M. (2021). The role of blockchain technology in telehealth and telemedicine. *International Journal of Medical Informatics*, 148:104399.
- [Alladi et al. 2019] Alladi, T., Chamola, V., Rodrigues, J. J., and Kozlov, S. A. (2019). Blockchain in smart grids: A review on different use cases. *Sensors*, 19(22):4862.
- [Almatarneh 2020] Almatarneh, A. (2020). Blockchain technology and corporate governance: The issue of smart contracts—current perspectives and evolving concerns.
- [Alphand et al. 2018] Alphand, O., Amoretti, M., Claeys, T., Dall’Asta, S., Duda, A., Ferrari, G., Rousseau, F., Tourancheau, B., Veltri, L., and Zanichelli, F. (2018). Iot-chain: A blockchain security architecture for the internet of things. In *2018 IEEE wireless communications and networking conference (WCNC)*, pages 1–6. IEEE.
- [Alves 2021] Alves, D. (2021). Proof-of-concept (poc) of restaurant’s food requests in the lisk blockchain/sidechain. In *Journal of Physics: Conference Series*, volume 1828, page 012110. IOP Publishing.

- [Angraal et al. 2017] Angraal, S., Krumholz, H. M., and Schulz, W. L. (2017). Blockchain technology: applications in health care. *Circulation: Cardiovascular quality and outcomes*, 10(9):e003800.
- [Antonopoulos 2017] Antonopoulos, A. (2017). *Mastering Bitcoin: Programming the Open Blockchain*. O’reilly, 2nd edition.
- [Araujo et al. 2019] Araujo, P., Viana, W., Veras, N., Farias, E. J., and de Castro Filho, J. A. (2019). Exploring students perceptions and performance in flipped classroom designed with adaptive learning techniques: A study in distributed systems courses. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 219.
- [Atonomi 2019] Atonomi (2019). Atonomi—bringing trust and security to iot. <https://atonomi.io/>. Acessado em 04/05/2021.
- [Atzori 2016] Atzori, M. (2016). Blockchain-based architectures for the internet of things: A survey (2016). *Available at SSRN 2846810 eLibrary*.
- [Azaria et al. 2016] Azaria, A., Ekblaw, A., Vieira, T., and Lippman, A. (2016). Medrec: Using blockchain for medical data access and permission management. In *2016 2nd International Conference on Open and Big Data (OBD)*, pages 25–30. IEEE.
- [Bach et al. 2018] Bach, L. M., Mihaljevic, B., and Zagar, M. (2018). Comparative analysis of blockchain consensus algorithms. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MI-PRO)*, pages 1545–1550. IEEE.
- [Back et al. 2014] Back, A., Corallo, M., Dashjr, L., Friedenbach, M., Maxwell, G., Miller, A., Poelstra, A., Timón, J., and Wuille, P. (2014). Enabling blockchain innovations with pegged sidechains. *URL: <http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains>, 72*.
- [Bahga and Madiseti 2016] Bahga, A. and Madiseti, V. K. (2016). Blockchain platform for industrial internet of things. *Journal of Software Engineering and Applications*, 9(10):533–546.
- [Balasubramaniam et al. 2020] Balasubramaniam, A., Gul, M. J. J., Menon, V. G., and Paul, A. (2020). Blockchain for intelligent transport system. *IETE Technical Review*, pages 1–12.
- [Bartoletti and Pompianu 2017] Bartoletti, M. and Pompianu, L. (2017). An empirical analysis of smart contracts: platforms, applications, and design patterns. In *International conference on financial cryptography and data security*, pages 494–509. Springer.
- [Berdik et al. 2021] Berdik, D., Otoum, S., Schmidt, N., Porter, D., and Jararweh, Y. (2021). A survey on blockchain for information systems management and security. *Information Processing & Management*, 58(1):102397.

- [Bessani et al. 2014] Bessani, A., Sousa, J., and Alchieri, E. E. (2014). State machine replication for the masses with bft-smart. In *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 355–362. IEEE.
- [Biswas and Muthukkumarasamy 2016] Biswas, K. and Muthukkumarasamy, V. (2016). Securing smart cities using blockchain technology. In *2016 IEEE 18th international conference on high performance computing and communications; IEEE 14th international conference on smart city; IEEE 2nd international conference on data science and systems (HPCC/SmartCity/DSS)*, pages 1392–1393. IEEE.
- [Bogner et al. 2016] Bogner, A., Chanson, M., and Meeuw, A. (2016). A decentralised sharing app running a smart contract on the ethereum blockchain. In *Proceedings of the 6th International Conference on the Internet of Things*, pages 177–178. ACM.
- [Buterin et al. 2014] Buterin, V. et al. (2014). A next-generation smart contract and decentralized application platform. *white paper*.
- [Cannavo and Lamberti 2020] Cannavo, A. and Lamberti, F. (2020). How blockchain, virtual reality and augmented reality are converging, and why. *IEEE Consumer Electronics Magazine*.
- [Castro et al. 1999] Castro, M., Liskov, B., et al. (1999). Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186.
- [Chaer et al. 2019] Chaer, A., Salah, K., Lima, C., Ray, P. P., and Sheltami, T. (2019). Blockchain for 5g: opportunities and challenges. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE.
- [Chen 2020] Chen, H. (2020). Quantum relay blockchain and its applications in key service. In *Proceedings of the 2020 4th International Conference on Cryptography, Security and Privacy*, pages 95–99.
- [Chowdhury et al. 2018] Chowdhury, M. J. M., Colman, A., Kabir, M. A., Han, J., and Sarda, P. (2018). Blockchain as a notarization service for data sharing with personal data store. In *2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, pages 1330–1335. IEEE.
- [Christidis and Devetsikiotis 2016] Christidis, K. and Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *Ieee Access*, 4:2292–2303.
- [Christin 2013] Christin, N. (2013). Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22nd international conference on World Wide Web*, pages 213–224.
- [Cocco et al. 2017] Cocco, L., Pinna, A., and Marchesi, M. (2017). Banking on blockchain: Costs savings thanks to the blockchain technology. *Future internet*, 9(3):25.

- [Croman et al. 2016] Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Sirer, E. G., et al. (2016). On scaling decentralized blockchains. In *International conference on financial cryptography and data security*, pages 106–125. Springer.
- [Dashkevich et al. 2020] Dashkevich, N., Counsell, S., and Destefanis, G. (2020). Blockchain application for central banks: A systematic mapping study. *IEEE Access*, 8:139918–139952.
- [De Angelis et al. 2018] De Angelis, S., Aniello, L., Baldoni, R., Lombardi, F., Margheri, A., and Sassone, V. (2018). Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain.
- [Deepa et al. 2020] Deepa, N., Pham, Q.-V., Nguyen, D. C., Bhattacharya, S., Prabadevi, B., Gadekallu, T. R., Maddikunta, P. K. R., Fang, F., and Pathirana, P. N. (2020). A survey on blockchain for big data: Approaches, opportunities, and future directions. *arXiv preprint arXiv:2009.00858*.
- [Dennis and Owen 2015] Dennis, R. and Owen, G. (2015). Rep on the block: A next generation reputation system based on the blockchain. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 131–138. IEEE.
- [Dettling 2018] Dettling, W. (2018). How to teach blockchain in a business school. In *Business Information Systems and Technology 4.0*, pages 213–225. Springer.
- [Díaz et al. 2016] Díaz, M., Martín, C., and Rubio, B. (2016). State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing. *Journal of Network and Computer applications*, 67:99–117.
- [Dorri et al. 2016] Dorri, A., Kanhere, S. S., and Jurdak, R. (2016). Blockchain in internet of things: Challenges and solutions. *CoRR*, abs/1608.05187.
- [Dorri et al. 2017] Dorri, A., Kanhere, S. S., Jurdak, R., and Gauravaram, P. (2017). Blockchain for iot security and privacy: The case study of a smart home. In *2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops)*, pages 618–623. IEEE.
- [Dorri et al. 2019] Dorri, A., Kanhere, S. S., Jurdak, R., and Gauravaram, P. (2019). Lsb: A lightweight scalable blockchain for iot security and anonymity. *Journal of Parallel and Distributed Computing*, 134:180–197.
- [Dziembowski et al. 2015] Dziembowski, S., Faust, S., Kolmogorov, V., and Pietrzak, K. (2015). Proofs of space. In *Annual Cryptology Conference*, pages 585–605. Springer.
- [Ekramifard et al. 2020] Ekramifard, A., Amintoosi, H., Seno, A. H., Dehghantanha, A., and Parizi, R. M. (2020). A systematic literature review of integration of blockchain and artificial intelligence. *Blockchain cybersecurity, trust and privacy*, pages 147–160.



- [Etash 2020] Etash (2020). Etash protocol. <https://eth.wiki/en/concepts/ethash/ethash>. Acessado em 04/05/2021.
- [Ethereum 2014] Ethereum, W. (2014). A secure decentralised generalised transaction ledger [j]. *Ethereum project yellow paper*, 151:1–32.
- [EVM 2020] EVM (2020). Ethereum virtual machine. [https://eth.wiki/en/concepts/evm/ethereum-virtual-machine-\(evm\)-awesome-list](https://eth.wiki/en/concepts/evm/ethereum-virtual-machine-(evm)-awesome-list). Acessado em 04/05/2021.
- [Fakhri and Mutijarsa 2018] Fakhri, D. and Mutijarsa, K. (2018). Secure iot communication using blockchain technology. In *2018 International Symposium on Electronics and Smart Devices (ISESD)*, pages 1–6. IEEE.
- [Frankenfield 2018] Frankenfield, J. (2018). Proof of burn. <https://www.investopedia.com/terms/p/proof-burn-cryptocurrency>. Acessado em 04/05/2021.
- [Greve et al. 2018] Greve, F., Sampaio, L., Abijaude, J., Coutinho, A., Valcy, Í., and Queiroz, S. (2018). Blockchain e a revolução do consenso sob demanda. *Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (Minicursos\_SBRC)*, 36.
- [Greve 2005] Greve, F. G. P. (2005). Protocolos fundamentais para o desenvolvimento de aplicações robustas. In *Minicursos SBRC 2005: Brazilian Symposium on Computer Networks*, pages 330–398.
- [Guo and Liang 2016] Guo, Y. and Liang, C. (2016). Blockchain application and outlook in the banking industry. *Financial Innovation*, 2(1):1–12.
- [Hou 2017] Hou, H. (2017). The application of blockchain technology in e-government in china. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–4.
- [Huh et al. 2017] Huh, S., Cho, S., and Kim, S. (2017). Managing iot devices using blockchain platform. In *2017 19th international conference on advanced communication technology (ICACT)*, pages 464–467. IEEE.
- [Ikeda 2018] Ikeda, K. (2018). Security and privacy of blockchain and quantum computation. In *Advances in Computers*, volume 111, pages 199–228. Elsevier.
- [Ji et al. 2018] Ji, Y., Zhang, J., Ma, J., Yang, C., and Yao, X. (2018). Bmpls: blockchain-based multi-level privacy-preserving location sharing scheme for telecare medical information systems. *Journal of medical systems*, 42(8):1–13.
- [Jiang et al. 2016] Jiang, J., Wen, S., Yu, S., Xiang, Y., and Zhou, W. (2016). Identifying propagation sources in networks: State-of-the-art and comparative studies. *IEEE Communications Surveys & Tutorials*, 19(1):465–481.

- [Khan and Salah 2018] Khan, M. A. and Salah, K. (2018). Iot security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems*, 82:395–411.
- [Kiayias et al. 2017] Kiayias, A., Russell, A., David, B., and Oliynykov, R. (2017). Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Annual International Cryptology Conference*, pages 357–388. Springer.
- [Kuperberg et al. 2019] Kuperberg, M., Kemper, S., and Durak, C. (2019). Blockchain usage for government-issued electronic ids: A survey. In *International Conference on Advanced Information Systems Engineering*, pages 155–167. Springer.
- [Kwon 2014] Kwon, J. (2014). Tendermint: Consensus without mining. *Draft v. 0.6, fall*, 1(11).
- [Lamport et al. 1982] Lamport, L., Shostak, R., and Pease, M. (1982). The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401.
- [Lao et al. 2020] Lao, L., Li, Z., Hou, S., Xiao, B., Guo, S., and Yang, Y. (2020). A survey of iot applications in blockchain systems: Architecture, consensus, and traffic modeling. *ACM Computing Surveys (CSUR)*, 53(1):1–32.
- [Larimer 2014] Larimer, D. (2014). Delegated proof-of-stake (dpos). *Bitshare whitepaper*, 81:85.
- [Lee and Lee 2015] Lee, I. and Lee, K. (2015). The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431–440.
- [Leng et al. 2020] Leng, J., Zhou, M., Zhao, L. J., Huang, Y., and Bian, Y. (2020). Blockchain security: A survey of techniques and research directions. *IEEE Transactions on Services Computing*.
- [Lerner 2015] Lerner, S. D. (2015). Dagcoin: a cryptocurrency without blocks. *White paper*.
- [Lin and Liao 2017] Lin, I.-C. and Liao, T.-C. (2017). A survey of blockchain security issues and challenges. *IJ Network Security*, 19(5):653–659.
- [Liu et al. 2020] Liu, B., Xiao, L., Long, J., Tang, M., and Hosam, O. (2020). Secure digital certificate-based data access control scheme in blockchain. *IEEE Access*, 8:91751–91760.
- [Liu et al. 2017] Liu, B., Yu, X. L., Chen, S., Xu, X., and Zhu, L. (2017). Blockchain based data integrity service framework for iot data. In *2017 IEEE International Conference on Web Services (ICWS)*, pages 468–475. IEEE.
- [Liu 2016] Liu, P. T. S. (2016). Medical record system using blockchain, big data and tokenization. In *International conference on information and communications security*, pages 254–261. Springer.

- [Matthews 2015] Matthews, C. M. (2015). Silk road creator found guilty of cybercrimes,. <https://www.wsj.com/articles/silk-road-creator-found-guilty-of-cybercrimes-1423083107>. Acessado em 04/05/2021.
- [Mazieres 2015] Mazieres, D. (2015). The stellar consensus protocol: A federated model for internet-level consensus. *Stellar Development Foundation*, 32.
- [Mehmood et al. 2017] Mehmood, Y., Ahmad, F., Yaqoob, I., Adnane, A., Imran, M., and Guizani, S. (2017). Internet-of-things-based smart cities: Recent advances and challenges. *IEEE Communications Magazine*, 55(9):16–24.
- [Meiklejohn et al. 2013] Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G. M., and Savage, S. (2013). A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140.
- [Mendling et al. 2018] Mendling, J., Weber, I., Aalst, W. V. D., Brocke, J. V., Cabanillas, C., Daniel, F., Debois, S., Ciccio, C. D., Dumas, M., Dustdar, S., et al. (2018). Blockchains for business process management-challenges and opportunities. *ACM Transactions on Management Information Systems (TMIS)*, 9(1):1–16.
- [Merkle-Patricia-Tree 2020] Merkle-Patricia-Tree (2020). Modified merkle patricia trie specification. <https://eth.wiki/en/fundamentals/patricia-tree>. Acessado em 04/05/2021.
- [Milutinovic et al. 2016] Milutinovic, M., He, W., Wu, H., and Kanwal, M. (2016). Proof of luck: An efficient blockchain consensus protocol. In *proceedings of the 1st Workshop on System Software for Trusted Execution*, pages 1–6.
- [Mistry et al. 2020] Mistry, I., Tanwar, S., Tyagi, S., and Kumar, N. (2020). Blockchain for 5g-enabled iot for industrial automation: A systematic review, solutions, and challenges. *Mechanical Systems and Signal Processing*, 135:106382.
- [Mqtt 2017] Mqtt (2017). Message queuing telemetry transport. <http://mqtt.org>. Acessado em 04/05/2021.
- [Nakamoto 2008] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Technical report, Bitcoin Org.
- [Narayanan et al. 2016] Narayanan, A., Bonneau, J., Felten, E., Miller, A., and Goldfeder, S. (2016). *Bitcoin and cryptocurrency technologies*. Princeton University Press.
- [Park et al. 2018] Park, J.-S., Youn, T.-Y., Kim, H.-B., Rhee, K.-H., and Shin, S.-U. (2018). Smart contract-based review system for an iot data marketplace. *Sensors*, 18(10).
- [Pavithra et al. 2019] Pavithra, S., Ramya, S., and Prathibha, S. (2019). A survey on cloud security issues and blockchain. In *2019 3rd International Conference on Computing and Communications Technologies (ICCCCT)*, pages 136–140. IEEE.

- [PoET 2018] PoET (2018). Poet 1.0 specification. <https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/poet.html>. Acessado em 04/05/2021.
- [Popov 2018] Popov, S. (2018). The tangle. *White paper*, 1:3.
- [Popov et al. 2020] Popov, S., Moog, H., and et al. (2020). The coordicide. *white paper Iota Foundation*.
- [Rahimi et al. 2020] Rahimi, P., Khan, N. D., Chrysostomou, C., Vassiliou, V., and Nazir, B. (2020). A secure communication for maritime iot applications using blockchain technology. In *2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 244–251. IEEE.
- [Rao and Dave 2019] Rao, A. R. and Dave, R. (2019). Developing hands-on laboratory exercises for teaching stem students the internet-of-things, cloud computing and blockchain applications. In *2019 IEEE Integrated STEM Education Conference (ISEC)*, pages 191–198. IEEE.
- [Rathee et al. 2021] Rathee, G., Balasaraswathi, M., Chandran, K. P., Gupta, S. D., and Boopathi, C. (2021). A secure iot sensors communication in industry 4.0 using blockchain technology. *Journal of Ambient Intelligence and Humanized Computing*, 12(1):533–545.
- [Ruta et al. 2017] Ruta, M., Scioscia, F., Ieva, S., Capurso, G., and Di Sciascio, E. (2017). Semantic blockchain to improve scalability in the internet of things. *Open Journal of Internet Of Things (OJIOT)*, 3(1):46–61.
- [Sagirlar et al. 2018] Sagirlar, G., Carminati, B., Ferrari, E., Sheehan, J. D., and Ragnoli, E. (2018). Hybrid-iot: Hybrid blockchain architecture for internet of things-pow sub-blockchains. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1007–1016. IEEE.
- [Saint-Andre et al. 2004] Saint-Andre, P. et al. (2004). Extensible messaging and presence protocol (xmpp): Core.
- [Samaniego et al. 2016] Samaniego, M., Jamsrandorj, U., and Deters, R. (2016). Blockchain as a service for iot. In *2016 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*, pages 433–436. IEEE.
- [Seitz et al. 2017] Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and Tschofenig, H. (2017). Authentication and authorization for constrained environments (ace). *Internet Engineering Task Force, Internet-Draft draft-ietf-aceoauth-authz-07*.

- [Shae and Tsai 2017] Shae, Z. and Tsai, J. J. (2017). On the design of a blockchain platform for clinical trial and precision medicine. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*, pages 1972–1980. IEEE.
- [Sharma et al. 2017] Sharma, P. K., Singh, S., Jeong, Y.-S., and Park, J. H. (2017). Dist-blocknet: A distributed blockchains-based secure sdn architecture for iot networks. *IEEE Communications Magazine*, 55(9):78–85.
- [Shelby et al. 2014] Shelby, Z., Hartke, K., and Bormann, C. (2014). The constrained application protocol (coap).
- [Silvano and Marcelino 2020] Silvano, W. F. and Marcelino, R. (2020). Iota tangle: A cryptocurrency to communicate internet-of-things data. *Future Generation Computer Systems*, 112:307–319.
- [Sornin et al. 2015] Sornin, N., Luis, M., Eirich, T., Kramp, T., and Hersent, O. (2015). Lorawan specification. *LoRa alliance*.
- [Sousa et al. 2018] Sousa, J., Bessani, A., and Vukolic, M. (2018). A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pages 51–58. IEEE.
- [Sporny and D. 2017] Sporny, M. and D., L. (2017). The web ledger protocol 1.0. <http://standards.ieee.org/develop/project/2418.html>. Acessado em 04/05/2021.
- [Stojkoska and Trivodaliev 2017] Stojkoska, B. L. R. and Trivodaliev, K. V. (2017). A review of internet of things for smart home: Challenges and solutions. *Journal of Cleaner Production*, 140:1454–1464.
- [Szabo 1997] Szabo, N. (1997). Formalizing and securing relationships on public networks. *First Monday*, 2(9).
- [Sztajnberg et al. 2018] Sztajnberg, A., da Silva Macedo, R., and Stutzel, M. (2018). Protocolos de aplicação para a internet das coisas: conceitos e aspectos práticos. *Sociedade Brasileira de Computação*.
- [Ta-Shma et al. 2017] Ta-Shma, P., Akbar, A., Gerson-Golan, G., Hadash, G., Carrez, F., and Moessner, K. (2017). An ingestion and analytics architecture for iot applied to smart city use cases. *IEEE Internet of Things Journal*, 5(2):765–774.
- [Tecnology 2017a] Tecnology, C. B. (2017a). Blockchain data format specification. <http://www.cbdforum.cn/bcweb/index/bz/1-0.html>. Acessado em 04/05/2021.
- [Tecnology 2017b] Tecnology, C. B. (2017b). Blockchain reference architecture. <http://www.cbdforum.cn/bcweb/index/article/bzwrr-1.html>. Acessado em 04/05/2021.

- [Technology 2017c] Technology, C. B. (2017c). Standard for the framework of blockchain use in internet of things (iot). <http://standards.ieee.org/develop/project/2418.htm>. Acessado em 04/05/2021.
- [Todd 2015] Todd, P. (2015). Ripple protocol consensus algorithm review. *Ripple Labs Inc White Paper (May, 2015)* <https://raw.githubusercontent.com/petertodd/rippleconsensus-analysis-paper/master/paper.pdf>.
- [Underwood 2016] Underwood, S. (2016). Blockchain beyond bitcoin. *Communications of the ACM*, 59(11):15–17.
- [Vinoski 2006] Vinoski, S. (2006). Advanced message queuing protocol. *IEEE Internet Computing*, 10(6):87–89.
- [Vučinić et al. 2015] Vučinić, M., Tourancheau, B., Rousseau, F., Duda, A., Damon, L., and Guizzetti, R. (2015). Oscar: Object security architecture for the internet of things. *Ad Hoc Networks*, 32:3–16.
- [Wu et al. 2019] Wu, M., Wang, K., Cai, X., Guo, S., Guo, M., and Rong, C. (2019). A comprehensive survey of blockchain: From theory to iot applications and beyond. *IEEE Internet of Things Journal*, 6(5):8114–8154.
- [Xie et al. 2020] Xie, R., Wang, Y., Tan, M., Zhu, W., Yang, Z., Wu, J., and Jeon, G. (2020). Ethereum-blockchain-based technology of decentralized smart contract certificate system. *IEEE Internet of Things Magazine*, 3(2):44–50.
- [Xu et al. 2017] Xu, C., Wang, K., and Guo, M. (2017). Intelligent resource management in blockchain-based cloud datacenters. *IEEE Cloud Computing*, 4(6):50–59.
- [Xu et al. 2019] Xu, X., Zhang, X., Gao, H., Xue, Y., Qi, L., and Dou, W. (2019). Become: blockchain-enabled computation offloading for iot in mobile edge computing. *IEEE Transactions on Industrial Informatics*, 16(6):4187–4195.
- [Yermack 2017] Yermack, D. (2017). Corporate governance and blockchains. *Review of Finance*, 21(1):7–31.
- [Yoo and Ko 2020] Yoo, H. and Ko, N. (2020). Blockchain based data marketplace system. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1255–1257. IEEE.
- [Yue et al. 2016] Yue, X., Wang, H., Jin, D., Li, M., and Jiang, W. (2016). Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control. *Journal of medical systems*, 40(10):1–8.
- [Zhang and Chen 2020] Zhang, C. and Chen, Y. (2020). A review of research relevant to the emerging industry trends: Industry 4.0, iot, blockchain, and business analytics. *Journal of Industrial Integration and Management*, 5(01):165–180.

- [Zhang et al. 2020] Zhang, W., Luo, Y., Fu, S., and Xie, T. (2020). Privacy-preserving reputation management for blockchain-based mobile crowdsensing. In *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–9. IEEE.
- [Zheng et al. 2017] Zheng, Z., Xie, S., Dai, H., Chen, X., and Wang, H. (2017). An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE international congress on big data (BigData congress)*, pages 557–564. IEEE.