



ciencia de dados

felipe coelho argolo

Página intencionalmente deixada em branco.

Ciência de dados

Filosofia e aplicações com software

Felipe Coelho Argolo felipe.c.argolo@protonmail.com

São Paulo, 10 de Fevereiro de 2019

Página oficial: <https://www.leanpub.com/fargolo>

Versão 1.0: Introdução; Capítulo 0; Capítulo 1; Capítulo 2; Capítulo 3; Capítulo 4; Capítulo 5 (em construção).

Para comentários, críticas, sugestões, ou simplesmente mandar um *oi*, entre em contato através do e-mail:
felipe.c.argolo@protonmail.com.

Prefácio

Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful
George Box & Norman R. Draper, *Empirical Model-Building and Response Surfaces*

Uma antiga aplicação da matemática é estimar quantidades em cenários desconhecidos usando observações anteriores. Os babilônios usavam interpolação linear para estimar informações: fazendo o censo populacional com intervalo (e.g. 4 em 4 anos), estimavam o valor dos anos não medidos, supondo que eles eram quantidades centrais em relação aos vizinhos.

Métodos iterativos também foram usados para aproximar a raiz quadrada de números naturais ($\sqrt{2}$) e números irracionais (π).

Essas técnicas deram fruto a abstrações mais gerais, aos campos da estatística e dos métodos numéricos. Em particular, o último século (XX) contou com a invenção do computador universal e dos processadores eletrônicos, impulsionando o poder de cálculos vertiginosamente.

O aperfeiçoamento teórico e instrumental trouxe ferramentas mais adequadas para cientistas e também algoritmos mais potentes para aplicações práticas.

Nos últimos anos, o campo ganhou forte notoriedade social e acadêmica em virtude dos resultados inéditos em problemas de predição com aplicação prática. Avanços em processamento de linguagem natural, visão computacional e algoritmos preditivos foram rapidamente aplicados pela indústria e por pesquisadores.

Uma descrição abrangente pode facilmente alcançar 1,000 páginas de texto sucinto, como o clássico ‘Deep Learning (Adaptive Computation and Machine Learning)’ de Goodfellow, Bengio and Courville. Outra obra de escopo e tamanho semelhante é a “Neural networks and learning machines”, de Simon Haykin. Inúmeros cursos online e videoaulas são produzidos por instituições de prestígio (e.g. Curso integral da Oxford em Deep Learning: <https://www.youtube.com/watch?v=PlhFWT7vAEw>).

Objetivos

Apesar da popularização da área e do enorme conteúdo disponível, ainda predominam duas características: (1) a formulação matemática é usada de forma intimidadora para credibilizar exposições; (2) os procedimentos e conclusões são executados ignorando premissas filosóficas.

Este texto percorre temas em diferentes profundidades. Os tópicos são escolhidos de maneira a amparar o estudo progressivo de estruturas matemáticas e preceitos epistemológicos em aprendizagem estatística (ciência de dados/machine learning).

Um guia para aplicação destas ferramentas para aqueles trabalhando na fronteira entre matemática aplicada e ciências naturais.

São exemplos de campos que fazem uso extenso das ferramentas descritas: neurociências (e.g. modelos lineares para sinal BOLD em *fmri*), psicometria (e.g. análise fatorial), ecologia, biologia molecular (e.g. testes estatísticos), ciências clínicas (e.g. meta-análises e inferência causal), economia (e.g. algo trading, modelos de mercado), marketing (e.g. mecanismos de recomendação).

O primeiro capítulo ilustra como o racional hipotético-dedutivo funciona para estudar teorias científicas. Aborda a relação entre ciências empíricas, o teorema do limite central, a distribuição normal e a distribuição *t*. O teste *t* de Student é aplicado para comparação de uma medida entre amostras.

O segundo capítulo amplia a perspectiva de comparações e testes. Entendemos o papel de descrever relações com os conceitos de tamanho de efeito (D de Cohen) e correlações lineares (ρ de Pearson). Também são introduzidas alternativas não-paramétricas aos procedimentos (ρ de Spearman e teste U de Mann-Whitney). Usando o gancho das relações lineares, vemos o conceito de regressão para fazer predições. Um framework ‘frequencista’ e linguagem R são usados para demonstrações de exemplos e exercícios.

O terceiro capítulo introduz o uso de muitas variáveis (análise multivariada). Grafos são a abstração base para relacionarmos múltiplos conceitos. Estudaremos regressão linear múltipla, colinearidade, mediação e

moderação. Conheceremos análise fatorial e sua generalização em equações estruturais: a implementação matemática do abrangente paradigma filosófico para modelos causais de Judea Pearl.

O quarto capítulo introduz redes neurais. Começamos da inspiração biológica envolvida nas primeiras abstrações concebidas para um neurônio artificial. Conheceremos a primeiramente máquina inteligente da história: Mark I Perceptron. Codificaremos um Mark I virtual do zero (*from scratch*) e observaremos a aprendizagem. Entenderemos o algoritmo de Gradient Descent, usando derivativas para encontrar mínimos na função de erro.

Redes Neurais expandem o poder de um neurônio com múltiplos nodos para a construção de sistemas preditivos complexos. Redes profundas incluem camadas sucessivas, permitindo transformações em sequência para resolver classes mais gerais de problemas. Entenderemos como os neurônios podem propagar erros aos outros, otimizando gradientes em conjunto com o mecanismo de *backpropagation*. Também codificaremos uma rede neural *from scratch*, Mark II.

O quinto capítulo apresenta um racional diferente para análise. Partindo do conceito de holismo epistemológico (W. van Quine), reabordamos alguns exemplos anteriores usando inferência bayesiana. Fazemos perguntas diferentes para obter outras informações sobre nossos dados. Usaremos R, Stan e um framework bayesiano para modelos simples e hierárquicos. Exploramos o poder das simulações através de Markov Chain Monte Carlo para obter estimativas difíceis de tratar analiticamente.

Sumário

Capítulo 0 - Ferramentas : programação com estatística básica

- Computadores
- R : Curso rápido
 - Instalação, R e Rstudio
 - Tipos
 - Operadores úteis: <- , %>%
 - Matrizes e dataframes
 - Gramática dos gráficos e ggplot
 - Funções
 - Vetores, loops e recursões: calculando a variância

Capítulo 1 - Os pássaros de Darwin e o método hipotético dedutivo

- Pássaros em Galápagos
 - Distribuição normal
 - Ciência experimental e o Teorema do limite central
- Método hipotético-dedutivo e Testes de hipótese
 - Valor p
 - Distribuição t de Student e teste t

Capítulo 2 - Sobre a natureza das relações

- Prelúdio: Quem precisa do valor p?
- Tamanho de efeito: D de Cohen
- Correlações lineares
 - Coeficiente de correlação ρ de Pearson
 - Predições com regressão linear
- Correlações e testes não paramétricos
 - ρ de Spearman
 - Teste U de Mann Whitney

Capítulo 3 - Análise multivariada, grafos e inferência causal

- Regressão múltipla
 - Colinearidade
- Grafos e trajetórias causais
 - Mediação e moderação
 - Análise fatorial
 - Equações estruturais

Capítulo 4 - Neurônios

- Um neurônio artificial: O perceptron
 - História e implementação do zero : Mark I
- Redes Neurais e Deep learning (múltiplas camadas)
- Gradient Descent
- Backpropagation

Em construção: Capítulo 5 - Contexto e inferência Bayesiana

- Intuições sobre distribuições probabilísticas
- Inferência Bayesiana para teste de diferenças e correlação linear
- O número de Euler
- Regressão logística
- Modelos hierárquicos
- Flexibilidade Bayesiana

- Usando priors
- O estimador Markov Chain Monte Carlo

Em construção: Capítulo 6 - Programação probabilística para contextos gerais

- Inferência Bayesiana para cosmologia
- Prevendo halos de matéria escura (Kaggle top solution)
- Redes neurais probabilísticas com PyMC3

Capítulo 7 - Ambientes desconhecidos

- Aprendizagem não supervisionada
- Redução de dimensões
- Clustering
- Aprendizagem semi-supervisionada
- Reinforcement learning

Pré-requisitos

Para uma leitura fluida do texto, recomenda-se a compreensão de rudimentos em probabilidade, estatística e cálculo (análise real). Os exemplos com ferramentas computacionais (exceto gráficos) usam sintaxe semelhante à matemática apresentada no texto. Assim, baixa familiaridade com linguagens de programação não é uma barreira.

Todos os exemplos podem ser reproduzidos usando software livre.

Leitura recomendada:

Neurociências

- Principles of neural science - Eric Kandel

Matemática pura e programação

- Better Explained (<https://betterexplained.com/>)
- What is mathematics - Courant & Robbins
- Fundamentos da matemática elementar - Iezzi (Vol. 5)
- MOOCs sobre estatística básica usando R (e.g.: <https://www.coursera.org/specializations/statistics>)
- Cálculo Diferencial e Integral - Piskounov =)
- <http://material.curso-r.com/>
- R Graphics Cookbook
- R Inferno
- Learn you a Haskell for Great Good
- Layered Grammar of Graphic - Hadley Wickham.
- The art of computer programming
- Algorithms unlocked
- Portais: statsexchange, stackoverflow, mathexchange, cross-validated.

Machine Learning

- An Introduction to Statistical Learning: with Applications in R
- Neural Networks and Learning Machines - Simon Haykin
- Stanford course on computer vision: <http://cs231n.stanford.edu/>
- Deep learning at Oxford 2015: (<https://www.youtube.com/watch?v=dV80NA1Eins&list=PLE6Wd9FR--EfW8dtjAuPoTuPcqmOV53Fu>)

Filosofia

- A lógica da pesquisa científica - K. Popper
- A estrutura das revoluções científicas - Thomas Kuhn
- Contra o Método - Paul Feyerabend
- Dois dogmas do empiricismo - Willard van Quine



Capítulo 0 : Ferramentas

Programação com estatística básica

Master Foo and the Shell Tools¹

Um aprendiz do caminho Unix veio ao Mestre Foo e disse: “Estou confuso. Não é o caminh Unix que cada programa deve se concentrar em uma coisa e fazê-la bem?

Mestre Foo assentiu.

O aprendiz continuou: “Também não é do caminho Unix que a roda não deve ser reinventada?

Mestre Foo assentiu novamente.

“Então, por que existem diversas ferramentas com capacidades similares em processamento de texto: sed, awk e Perl? Com qual delas posso praticar melhor o caminho Unix?”

Mestre Foo perguntou ao aprendiz: “Se você tem um arquivo de texto, qual ferramenta usaria para produzir uma cópia com algumas palavras trocadas por uma string de sua escolha?”

O aprendiz torceu o nariz e disse: “As expressões regulares de Perl seriam um excesso para tarefa tão simples. Eu não conheço awk, e venho escrevendo scripts sed nas últimas semanas. Como tenho experiência com sed, eu preferiria ele no momento. Mas se o trabalho precisa ser feito apenas uma vez, um editor de textos funcionaria.”

Mestre Foo assentiu e respondeu: “Quando você estiver com fome, coma; quando estiver com sede, beba; quando estiver cansado, durma.”

E, ao ouvir isso, o aprendiz foi iluminado.

¹<http://catb.org/esr/writings/unix-koans/shell-tools.html>

Computadores

Ao longo do texto, usaremos exemplos com software. Computadores são úteis para acelerar os cálculos necessárias para nossos objetivos.

Há milênios, o homem usa instrumentos, como ábacos e tabelas, para fazer operações extensas e precisas envolvendo grandes números. Dado um problema ou dado a ser computado, esses instrumentos mecanismos automatizam partes do processo devido à maneira como foram construídos. A principal diferença destas ferramentas para os computadores de hoje é que nossas máquinas podem ser programadas para fazer computações arbitrárias.

Ada Lovelace (*10 December 1815 – 27 November 1852*) foi a primeira a descobrir essa possibilidade. Estudando a Máquina Analítica de Charles Babbage, Ada concebeu uma maneira de realizar computações para as quais a máquina não havia sido desenhada originalmente. O programa concebido calculava os Números de Bernoulli. Discutivelmente, alterar a estrutura de máquinas mais simples também consiste em reprogramá-las.

Máquinas desse tempo pesavam toneladas e eram muito mais lentas. O avançar dos anos tornou a tecnologia mais acessível, ao ponto de possibilitar computadores pessoais de alta potência e baixo-custo. Além disso, ao invés de operações mecânicas complexas, podemos usar linguagens de programação que traduzem comandos baseados no inglês para instruções de máquina.

Os programas aqui apresentados são escritos em R, Stan e Python. As três têm código aberto, podendo ser obtidas, instaladas e usadas sem pagamentos. Sendo um texto didático, as implementações com software priorizam legibilidade. Os três frameworks usam libs em C/C++/Fortran para otimizar computações e interface com GPU (graphic processing unit).

R será mais usada. É uma linguagem interpretada voltada à computação estatística, possuindo ferramentas úteis em sua biblioteca de base. Entre estas, funções para gerar e manipular distribuições probabilísticas.

Sendo uma linguagem de ‘alto nível’, não temos sobrecarga cognitiva no programador com manejo de memória e hardware no código. A abstração de detalhes físicos, como registradores da CPU, são feitas automaticamente pelo interpretador. O ecossistema para visualização de dados possui poder e flexibilidade. A comunidade R cresce rápido e fluência nessa linguagem dá acesso a ferramentas muito diversas com bases grandes de suporte. Há suporte para estilo funcional e orientado a objetos.

Stan é uma linguagem/plataforma de domínio específico bastante popular entre estatísticos bayesianos. Possui ferramentas poderosas (e.g: Variational inference, MCMC com NUTS e HMC) para lidar com distribuições probabilísticas e inferência nesse contexto.

Python é uma linguagem de propósito geral. Bastante popular e dotada de uma base de usuários imensa. Linguagem de primeira escolha como interface de alto nível (wrapper) para a maioria das tecnologias de aplicação industrial (e.g: PyTorch, Pyro, Tensorflow). Com dois “dialetos” incompatíveis (2.X e 3.X) diferindo em mínimos detalhes, possui também uma variedade de opções que pode confundir iniciantes (e.g: pip vs. conda). Vamos precisar de Python (PyMC/Pyro) para combinar inferência Bayesiana e redes neurais.

R: Curso rápido

Programas de computador são importantes ao longo dos próximos capítulos para realizar cálculos, gerar dados e visualizações.

Felizmente, os programas que escreveremos são simples, de forma que não precisamos conhecer todos os recursos e características da linguagem R. Neste capítulo, entenderemos os instrumentos básicos para caminharmos.

Veremos diversas maneiras de escrever um programa para calcular a variância σ^2 de um conjunto de medidas.

Instalação

R

Instruções para download e instalação podem ser encontradas em:

<https://cloud.r-project.org/>

Em Windows, o processo costuma consistir em clicar no executável de instalação e concordar com os prompts. Para Linux, envolve adicionar o CRAN à lista de repositórios e baixar o pacote *r-base* ou o código-fonte/tarball diretamente do website. Há inúmeros tutoriais explicando a instalação.

Rstudio

Com o R instalado, recomendo o uso do ambiente de desenvolvimento RStudio (<https://www.rstudio.com/>) para obter algumas facilidades. Entre elas: atalhos *vim*, editor com highlight de sintaxe, autocompletar, renderização em tempo real de animações e plots, visualização de datasets, ambiente de desenvolvimento, logs, suporte a markup languages, como Markdown, RMarkdown e Latex.²

Tipos

Primeiro, vamos conhecer as entidades básicas do R. Lidamos rotineiramente com vetores, que são células contíguas contendo dados. Os dados podem ser de tipos: lógico (verdadeiro/falso), caracteres, números inteiros, reais e complexos:

“logical”: a vector containing logical values (TRUE/FALSE) “integer”: a vector containing integer values (1,2,3,4...,23,26...)
“double”: a vector containing real values (3.14...)
“complex”: a vector containing complex values (2 +2i)
“character”: a vector containing character values (“string”)

Para saber o tipo de um objeto em R, use `typeof(objeto)`. Podemos acessar elementos de um vetor pelo seu índice, independente do tipo. Declaramos dois vetores, character e double.

```
>a <- c("banana", "terracota", "pie")
>b <- c(2.2, 4.4, 5.5)
> typeof(a)
[1] "character"
> typeof(b)
[1] "double"
```

A função *combine*: `c(arg1,arg1,...)` combina argumentos em um vetor. Para nossas aplicações, vamos usar números reais (double) na maioria dos casos. Os tipos integer, double e complex fazem parte da classe dos números (*numeric*)

```
>class(b)
[1] "numeric"
```

²Este texto é escrito em Markdown e o código-fonte pode ser encontrado em <https://github.com/fargolo/stat-learn>

Operadores

Além dos operadores clássicos (+,-,/,-, ...), usamos constantemente dois operadores pouco comuns: O operador “`<-`” atribui o valor da expressão à sua direita ao objeto à sua esquerda. É preferível ao operador “`=`” para evitar confusão ao passar argumentos de funções e fazer comparações lógicas.

```
>a <- 3  
>a  
[1] 3
```

O operador “`%>%`” da biblioteca **magrittr** fornece o resultado da expressão à sua esquerda como argumento para a expressão à sua direita. Evita aninhamento de expressões, tornando fluxos de computações mais legíveis.

As expressões a seguir são equivalentes.

```
>library(magrittr)  
>result <- 3 %>% exp %>% exp  
>result  
[1] 528491311  
>result == exp(exp(3))  
[1] TRUE
```

Onde $\exp(a) = e^a$, $e \sim 2.72\dots$. A expressão “`3 %>% exp %>% exp`” equivale a “ $\exp(\exp(3))$ ”, ou e^{e^3} . Usando parênteses, partimos da última computação. Usando o pipe (`%>%`), começamos com a primeira operação. Para a maioria das abstrações, é uma boa maneira de encadear funções.

Notem que para usar um recurso da biblioteca **magrittr**, carregamos usando o comando `library(magrittr)`. Para instalar uma biblioteca do repositório oficial (CRAN), usamos o comando `install.packages("magrittr")`.

Matrizes e data frames

R possui estruturas que ajudam a manipulação de dados estruturados como os que vemos comumente em ciências.

A mais simples é a lista. Uma lista é um conjunto de objeto de quaisquer tipos. Assim, podemos ter uma lista contendo vetores, doubles, matrizes e gráficos! Tudo em uma estrutura.

```
>mlist <- list(a = c(1,5,6,7), b = c("a","b","c","d"))  
>mlist  
$a  
[1] 1 5 6 7  
$b  
[1] "a" "b" "c" "d"  
>class(mlist)  
[1] "list"
```

Podemos acessar estruturas internas da maioria dos objetos em R pelo nome usando o operador `$`:

```
>typeof(mlist$a)  
[1] "double"  
>typeof(mlist$b)  
[1] "character"
```

Outro tipo útil é composto pelas matrizes, que correspondem às matrizes da matemática, podendo também conter caracteres em suas células.

```
>matrix(data=c(mlist$a, mlist$b), ncol=2)  
[,1] [,2]  
[1,] "1" "a"
```

```
[2,] "5"  "b"
[3,] "6"  "c"
[4,] "7"  "d"
```

Podemos conduzir multiplicação de matrizes facilmente.

```
>mat_example <- matrix(c(.5,.25,.25,.5,0,.5,.25,.25,.5), nrow=3, byrow=TRUE)
>mat_example
 [1,] [,1] [,2] [,3]
[1,] 0.50 0.25 0.25
[2,] 0.50 0.00 0.50
[3,] 0.25 0.25 0.50
>mat_example %*% c(1,0,1)
 [,1]
[1,] 0.75
[2,] 1.00
[3,] 0.75
```

Por fim, data.frames são extensões das matrizes:

```
>mat_example %>% data.frame
      X1    X2    X3
1 0.50 0.25 0.25
2 0.50 0.00 0.50
3 0.25 0.25 0.50
```

Data frames são os objetos mais comumente tratados em R e seguem o formato tidy.

Cada variável corresponde a uma coluna.

Cada observação corresponde a uma linha.

Cada tipo de unidade observacional forma uma tabela.

Um exemplo visual torna as coisas mais fáceis. A seguir, temos uma variável categórica (País) e duas numéricas (Número de médicos por 1.000 habitantes em 2011 e Expectativa de vida ao nascer) em formato tidy:

Note que cada linha corresponde a apenas um país (observação) e cada coluna representa uma variável. Se queremos ver a observação 9, vamos à linha correspondente e podemos encontrar os valores: “Armenia” (País), “2.845” (Médicos/1.000 hab. em 2011) e “71” (Expectativa de vida ao nascer).

Para acessar o valor correspondente, usamos índices separados por vírgula. O primeiro espaço é reservado às linhas selecionadas e deve ser um vetor de números (linhas selecionadas) ou vetor com valores lógico do tamanho do dataset (valores com índices TRUE serão incluídos). O segundo espaço corresponde às colunas e deve conter índices numéricos ou nomes das variáveis.

```
# primeiras 5 linhas com variaveis species e Sepal.Length'
>iris[1:5,c("Species",'Sepal.Length')]
  Species Sepal.Length
1  setosa      5.1
2  setosa      4.9
3  setosa      4.7
4  setosa      4.6
5  setosa      5.0
```

Country	Doctors 2011	Life Expectancy at Birth
Aruba	NA	NA
Andorra	NA	83
Afghanistan	0.23400000	61
Angola	NA	52
Albania	1.11300000	74
Arab World	1.52685042	NA
United Arab Emirates	NA	77
Argentina	NA	76
Armenia	2.84500000	71

Figure 1: País, Número de médicos a cada 1000 habitantes em 2011 e Expectativa de vida ao nascer. “NA” corresponde a dados faltantes no R. Layout do RStudio Fonte: WHO

Gramática dos gráficos e ggplot

Uma das ferramentas de destaque no ecossistema R é a **ggplot**. Ela provê uma sintaxe bastante poderosa e flexível para plotar visualizações. O segredo está em seu design, que utiliza gramática de gráficos (**Grammar of GraphicsPlot**).

Bertin³ delineou essa abordagem, que consiste em mapear características dos dados a elementos visuais seguindo uma sintaxe consistente. A lib ggplot implementa uma gramática em camadas, possibilitando superposições para gráficos complexos.

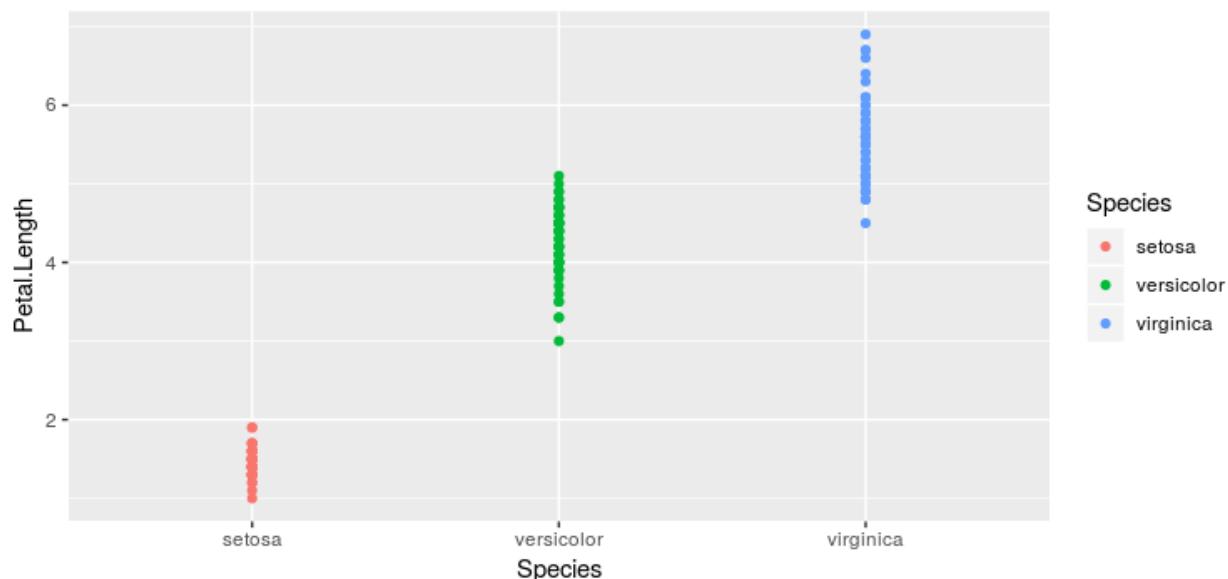
```
>head(sleep)
extra group ID
1 0.7 1 1
2 -1.6 1 2
3 -0.2 1 3
```

Para usarmos o ggplot, podemos declarar (1) o dataframe usado, (2) a relação entre medidas e parâmetros estéticos e (3) objetos geométricos. Parâmetros opcionais podem ser usados, aumentando o número de camadas ou criando transformações.

Assim, podemos plotar um histograma das medidas dos dois grupos com (1) dataset iris; (2) dimensão y: tamanho da pétala, cores:espécie, dimensão x: espécie; e (3) objeto geométrico: ponto.

Assim, teremos pontos com a altura (dimensão y) correspondente à medida da pétala e separados ao longo do eixo x por espécies. O ggplot automaticamente discretiza o eixo x.

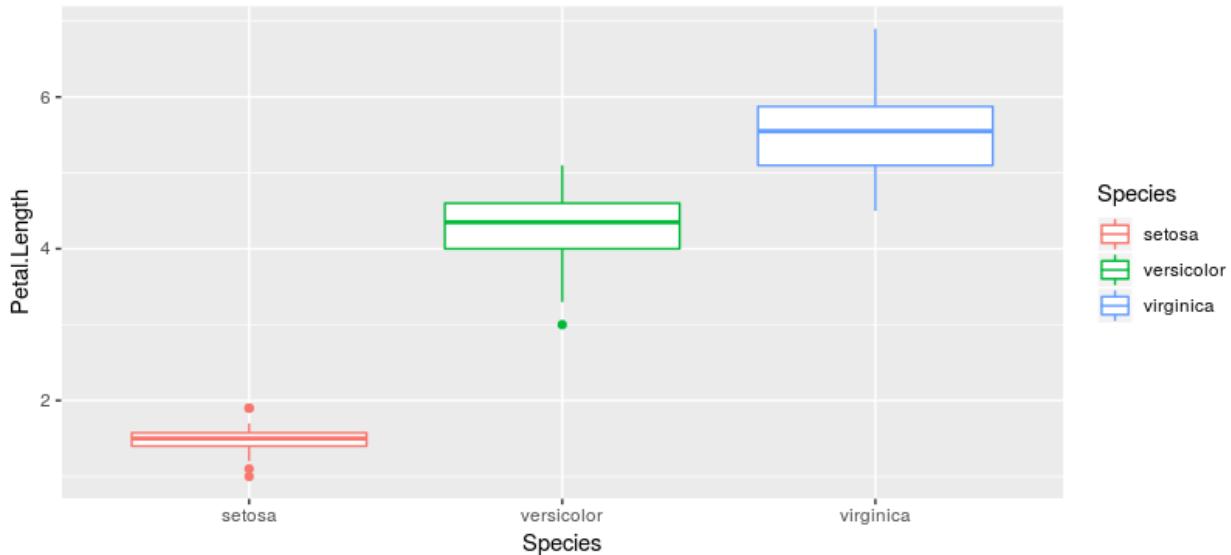
```
>library(ggplot2)
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+
  geom_point()
```



Para ilustrar a flexibilidade da biblioteca, note que mudando apenas o objeto geométrico (geom), obtemos um gráfico diferente, mantendo dados e relações (mappings) iguais :

```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+
  geom_boxplot()
```

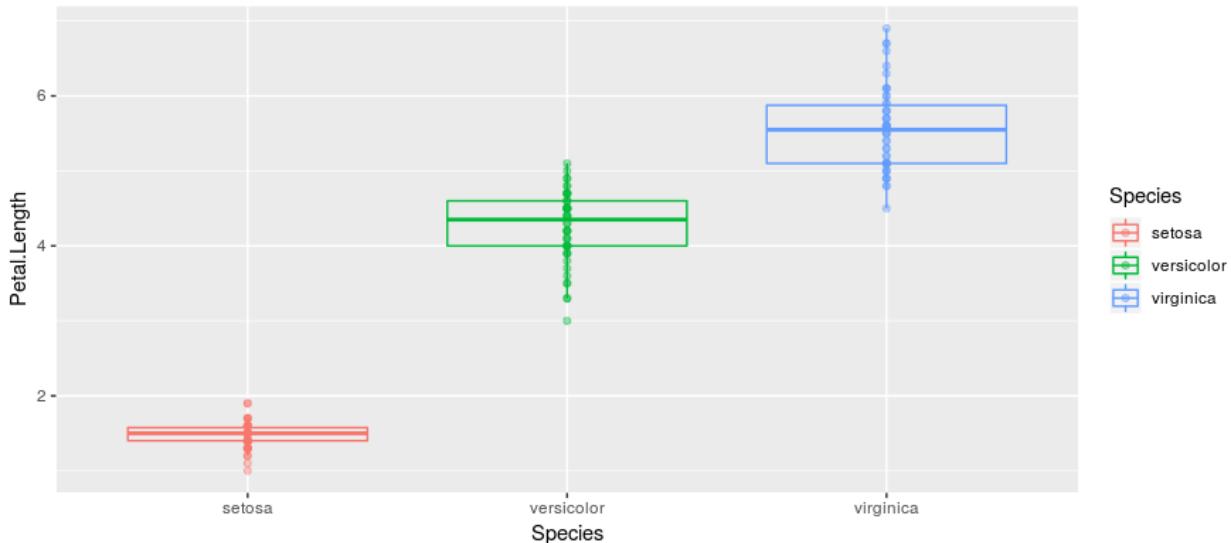
³Bertin, J. (1983),*Semiology of Graphics*, Madison, WI: University of Wisconsin Press



As figuras acima são conhecidas como boxplots. O centro correspondente à mediana (percentil 50), as bordas correspondem aos percentis 25 (inferior) e 75 (superior). Os fios, conhecidos como “bigodes”, estendem-se até $1,5^* \text{ IQR}$ (onde $\text{IQR} = \text{Percentil 75} - \text{Percentil 25}$).

É possível adicionar camadas e estas podem sobreescrivere informação de camadas anteriores. Isso torna a sintaxe do ggplot altamente modular. A seguir, superpomos pontos e boxplot:

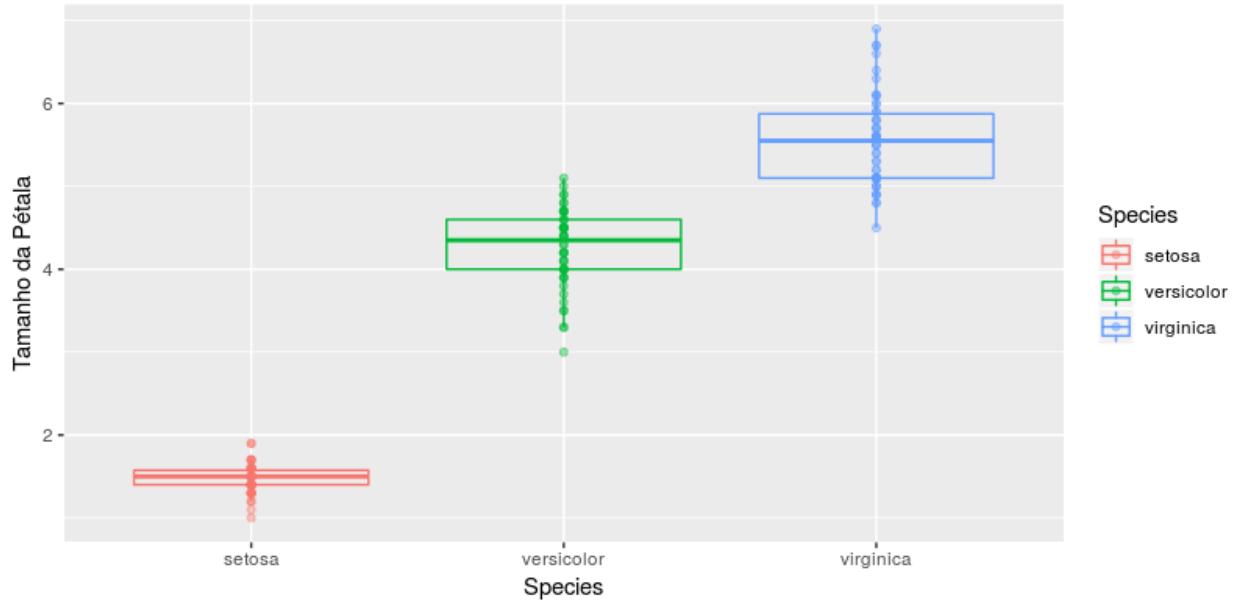
```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+  
  geom_point(alpha=0.4)+ # camada 1  
  geom_boxplot(alpha=0) # camada 2
```



O parâmetro *alpha* regula a transparência dos objetos. Colocamos os boxplot com transparência total (*alpha*=0), dando visibilidade aos pontos (*alpha*=0.4). Adicionamos algum grau de transparência para que pontos superpostos sejam mais escuros que pontos individuais. Adicionaremos uma terceira camada, que substitui o rótulo do eixo y para uma legenda em português:

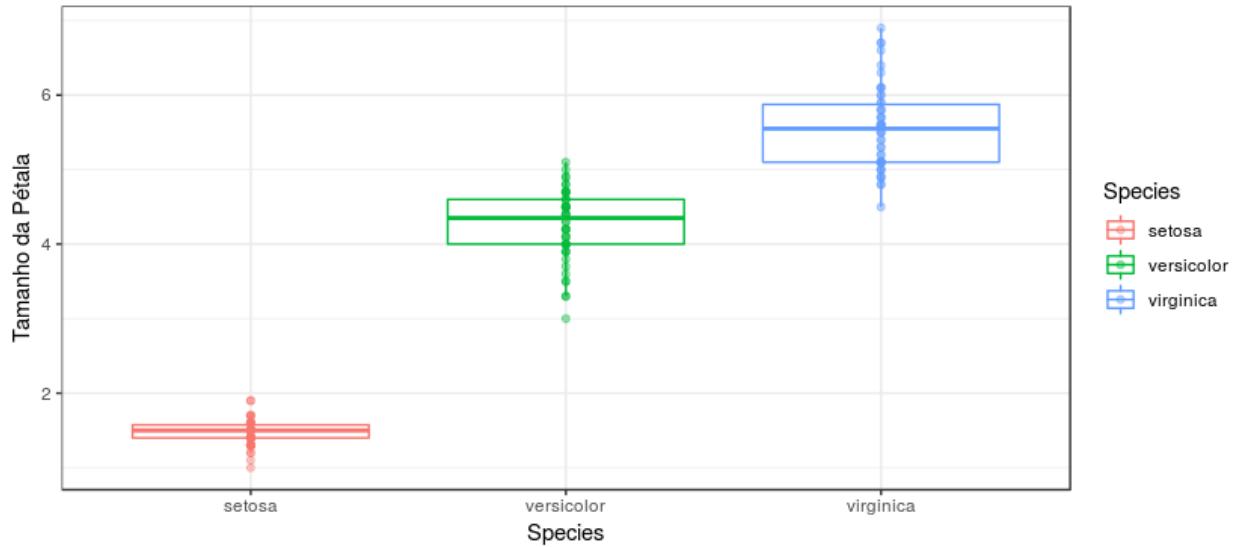
```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+  
  geom_point(alpha=0.4)+ # camada 1  
  geom_boxplot(alpha=0)+ # camada 2
```

```
ylab("Tamanho da Pétala") # camada 3
```

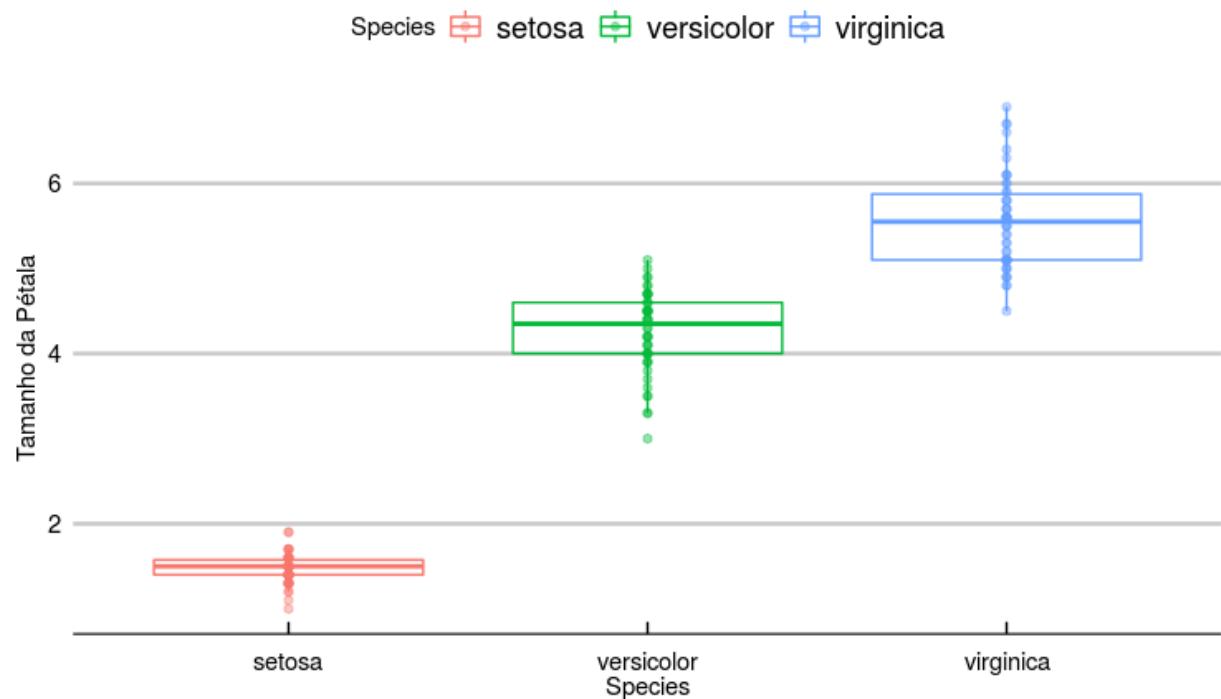


Ainda, existem temas prontos para mudar o estilo geral da imagem:

```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+  
  geom_point(alpha=0.4)+ # camada 1  
  geom_boxplot(alpha=0)+ # camada 2  
  ylab("Tamanho da Pétala") # camada 3  
  theme_bw() # camada 4: tema
```



```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+  
  geom_point(alpha=0.4)+ # camada 1  
  geom_boxplot(alpha=0)+ # camada 2  
  ylab("Tamanho da Pétala") # camada 3  
  theme_economist_white(gray_bg = F) # camada 4: tema
```



Funções

Uma das formas de escrever programas é através de funções.

Podemos declarar funções que (1) aceitam argumentos de entrada, (2) executam computações com esses argumentos e (3) devolvem resultados na saída.

Assim, podemos criar a função soma2, que recebe dois argumentos numéricos e retorna a soma de ambos.

```
>soma2 <- function(argumento1,argumento2){  
  return(argumento1+argumento2)  
}
```

Ao invocarmos soma2 com os argumentos 2 e 5, recebemos soma2(argumento1=2, argumento2=5) = 2+5 = 7.

```
>soma2(argumento1=2,argumento2=5)  
[1] 7
```

Podemos omitir o nome dos argumentos. Assim os objetos são passados na ordem de entrada.

```
>soma2(2,3)  
[1] 5
```

Por padrão, o valor retornado é mostrado no console.

R aceita em sua sintaxe que uma função seja argumento de outra numa mesma instrução:

```
>soma2(2, soma2(3,2) )  
[1] 7
```

A expressão acima é equivalente a $(2 + (3 + 2)) = 7$.

Podemos definir a função de média para um vetor de números, dado pela (1) soma dividida pelo (2) tamanho do vetor:

```

>mean_vec <- function(x){
  sum(x)/length(x)
}
>mean_vec(b) # Anteriormente definido por b <- c(2.2, 4.4, 5.5)
[1] 4.033333

```

`sum(x)` retorna a soma de todos os elementos do vetor `x`. `length(x)` retorna o tamanho (número de células) do vetor `x`.

A média é uma medida de tendência central para um conjunto de observações. É o ponto mais perto de todos os outros.

Muitas formas de calcular a variância

Também podemos calcular uma medida relacionada ao quanto nossos valores se afastam do centro.

Primeiro, calculamos uma distância entre cada elemento x e a média das observações μ . A noção de distância implica que ela deve ser um valor positivo. Supondo que x e μ são medidas num espaço ordenado, podemos usar o módulo da diferença entre os valores: $\|x - \mu\|$. Ainda, podemos usar o quadrado da diferença: $d_i = (x_i - \mu)^2$.

A variância σ^2 das observações é uma medida da dispersão de toda a amostra.

Para calcular σ^2 , somamos todas as distâncias d_i e dividimos o resultado por $n - 1$.

```

>var_2 <- function(x) sum((x - mean(x))^2) / (length(x) - 1)
>var_2 (b)
[1] 2.823333

```

Sendo proporcional às distâncias dos valores em relação à média, a variância σ^2 tende a ser maior quando os valores são muito distintos entre si:

```

>c <- c(100,200,1,45,-24)
>var_2(c)
[1] 7966.3

```

Outra medida de dispersão, dada nas unidades originais da medida observada, é o desvio-padrão σ , dado pela raiz da variância σ^2 .

```

>var_2(b) %>% sqrt
[1] 1.680278

```

O R possui funções embutidas para muitas aplicações estatísticas: `sd` (desvio-padrão), `var` (variância), `mean` (média)... Em especial, temos funções prontas para trabalhar com diversas distribuições probabilísticas de variáveis aleatórias. Para sortear 10 números de uma distribuição normal:

```

>rnorm(n=10, mean=0, sd=1)
[1] 0.2874490 0.2931469 3.1897423 1.7445002 3.3998010 -0.1482911
[7] 2.0257046 -0.6002109 -0.2840376 -0.7715565

```

Distribuição gamma.

```

>rgamma(n=10, shape=1)
[1] 1.1183441 1.2770135 1.0972053 1.4820536 2.3542620 0.8231831 0.5535210
[8] 5.0481559 0.2853060 0.1623315

```

Exponencial:

```

>rexp(n=10, rate = 1)
[1] 0.31657586 0.26676766 0.02288276 0.92801416 0.44006133 0.05238540
[7] 1.10213153 0.91931786 2.58807134 0.41825081

```

Vetores, loops e recursões

Anteriormente, definimos a função para calcular variância como:

```
>var_2 <- function(x) sum((x - mean(x))^2) / (length(x) - 1)
```

Isso só é possível porque o R aplica funções a vetores de maneira automática. Assim, a expressão $(x - mean(x))^2$ subtrai a média de cada elemento do vetor x.

Normalmente, é necessário usar estruturas recursivas para isso. O laço for (for loop) define uma sequência de tamanho n definido e repete um bloco de comandos n vezes. Se queremos imprimir números entre 1 e 10:

```
>for (i in 1:10) print(i)
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
```

A instrução avalia print(i) para valores i=1,2,3..,10 de forma repetida.

Vamos reescrever nossa função para calcular variância σ^2 usando um loop. Podemos definir um loop com o tamanho do vetor x e calcular o quadrado da diferença em cada elemento.

Assim,

```
var_3 <- function(x){
  accumulator <- numeric() #armazena distâncias
  for (i in 1:length(x)) # loop começa em 1 segue até o tamanho do vetor
    accumulator[i] <- (x[i] - mean(x))^2 # calcula e armazena distâncias.
  return (sum(accumulator) / (length(x) - 1)) #calcula media
}
```

Ambas definições apresentam o mesmo resultado que a implementação nativa do R:

```
> var(b)
[1] 2.823333
> var_2(b)
[1] 2.823333
> var_3(b)
[1] 2.823333
```

Ainda, uma maneira de manipular muitos elementos é através de funções de alta ordem. Estas funções recebem outras funções como argumentos. Um exemplo é a função map da lib purrr. Definimos uma função para a distância, $f(y) = (y - \mu)^2$, e aplicamos em todos os elementos. Só então, somamos os resultados e dividimos por n-1.

Tudo pode ser feito em apenas um pipe:

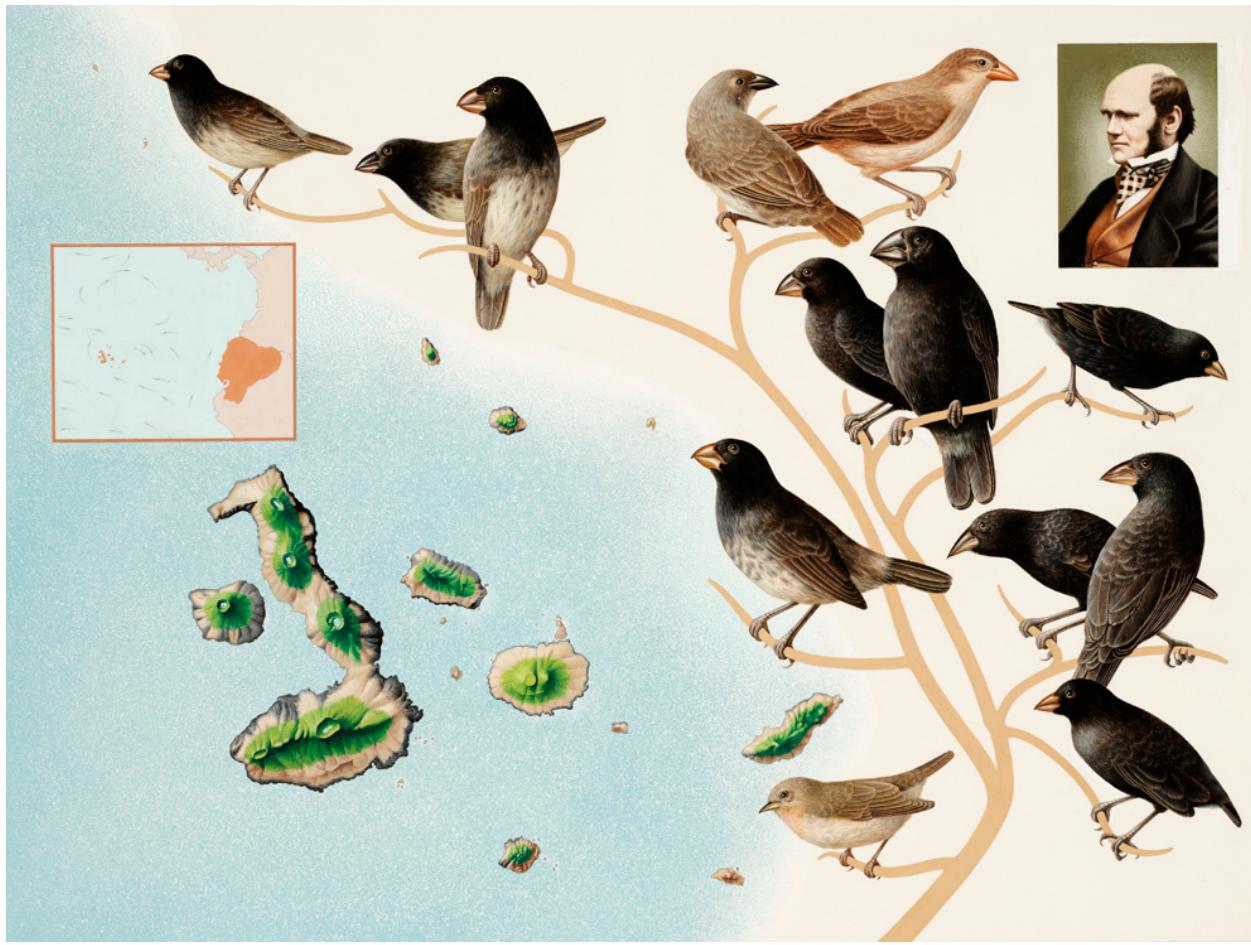
```
>map(.f = function(y) (y - mean(arg))^2, .x = arg) %>% # Define e aplica função
  unlist(.) %>% sum(.) / (length(arg) - 1) # Soma as distâncias e divide por n-1
```

Quando usamos o pipe, o caractere de ponto (.) se refere ao valor fornecido como entrada pela pipe anterior. Assim, sum(.), no exemplo, acima soma os valores passados pela função unlist(.), que por sua vez, transforma em vetor uma lista de valores passada pela função map. Nossa função pode ser escrita:

```
var_4 <- function(arg){  
  purrr::map(.f = function(y) (y - mean(arg))^2, .x = arg) %>%  
    unlist %>% sum(.) / (length(arg) - 1)  
}  
> var_4(b)  
[1] 2.823333
```

Exercícios

1. Qual a diferença entre linguagens compiladas e interpretadas?
2. Um programa escrito em R pode ser escrito em qualquer outra linguagem. Esta afirmação é verdadeira? Por quê?
3. Cite 3 recursos que uma IDE fornece ao programador.
4. Modifique o tema de fundo do RStudio para um de cor escura (menos luz para os olhos :)).
5. Usando o operador `<-`, produza:
 - Um vetor com componentes do tipo logical
 - Dois vetores de 5 elementos do tipo double
 - A soma dos elementos nos vetores do item b.
 - A divisão entre elementos dos vetores do item b.
 - Aplique as funções `sd`, `mean` e `var` em amostras normais aleatórias de $n = 10, 30, 100$ e 300 . A função `rnorm(n,mean,sd)` pode ajudar. Compare os valores da distribuição de origem com os obtidos.
7. *UnLISP it!* Transforme as seguintes expressões, substituindo parênteses aninhados pelo operador pipe (`%>%`) quando julgar conveniente:
 - `round(mean(c(10, 2, 3)))`
 - `round(mean(rnorm(n = ceiling(runif(1,0,10))))))`
 - `paste("a",seq(1:max(sample(1:10))))`
 - `round(nrow(iris) + exp(1), digits = ceiling(runif(1,0,10))))`
8. Usando o código das funções `var_2` (vetorizado), `var_3` (for loop) e `var_4` (função de alta ordem map)
 - Escreva as funções correspondentes (`sd_2`, `sd_3`, `sd_4`) para desvio-padrão e compare com a função padrão do R (`sd`). Dica: Basta aplicar raiz quadrada ao valor final retornado anteriormente!
9. Usando o dataset `iris`
 - Selecione apenas os exemplos com tamanho de pétala maior que 4.
 - Selecione os 10 maiores exemplares. Suponha que o tamanho é dado pela média das 4 medidas fornecidas.
 - Calcule a média e o desvio-padrão para duas medidas em cada espécie.
 - Faça um scatterplot entre duas medidas
 - Adicione cores de acordo com a espécie
 - Adicione o rótulo de texto a um dos pontos
 - Mude títulos (principal, eixos x e y, legenda)
 - Mude o tema de fundo. Dica: experimente os temas da lib `ggthemes`
10. Usando loops, escreva uma função que retorna uma aproximação de e .
 - Lembre-se de que $e = \lim_{n \rightarrow \infty} (1 + \frac{1}{n})^n$.



Capítulo 1 : Os pássaros de Darwin e o método hipotético-dedutivo.

Testes estatísticos e distribuições probabilísticas

Parte 1 - Introdução

Charles Darwin observou que os pássaros fringílideos nas ilhas de Galápagos apresentavam variedades de formato e tamanho dos bicos. Sua intuição sobre a origem das variedades a partir de um ancestral comum foi um dos argumentos mais contundentes do “On the Origin of Species” (1859). Essa história é o ponto de partida para este capítulo.

Estudamos a relação natural entre ciências empíricas e duas distribuições probabilísticas: a distribuição normal e a distribuição t, relacionadas entre si. A adoção da distribuição normal em trabalhos científicos é popular, porém os motivos são pouco entendidos. O Teorema do Limite Central é fundamental nesse contexto.

Usamos as distribuições citadas para estudar as medidas dos bicos dos tentilhões em pequenas amostras de cada ilha e fazer inferências sobre as populações. O racional de testes de hipótese é introduzido.

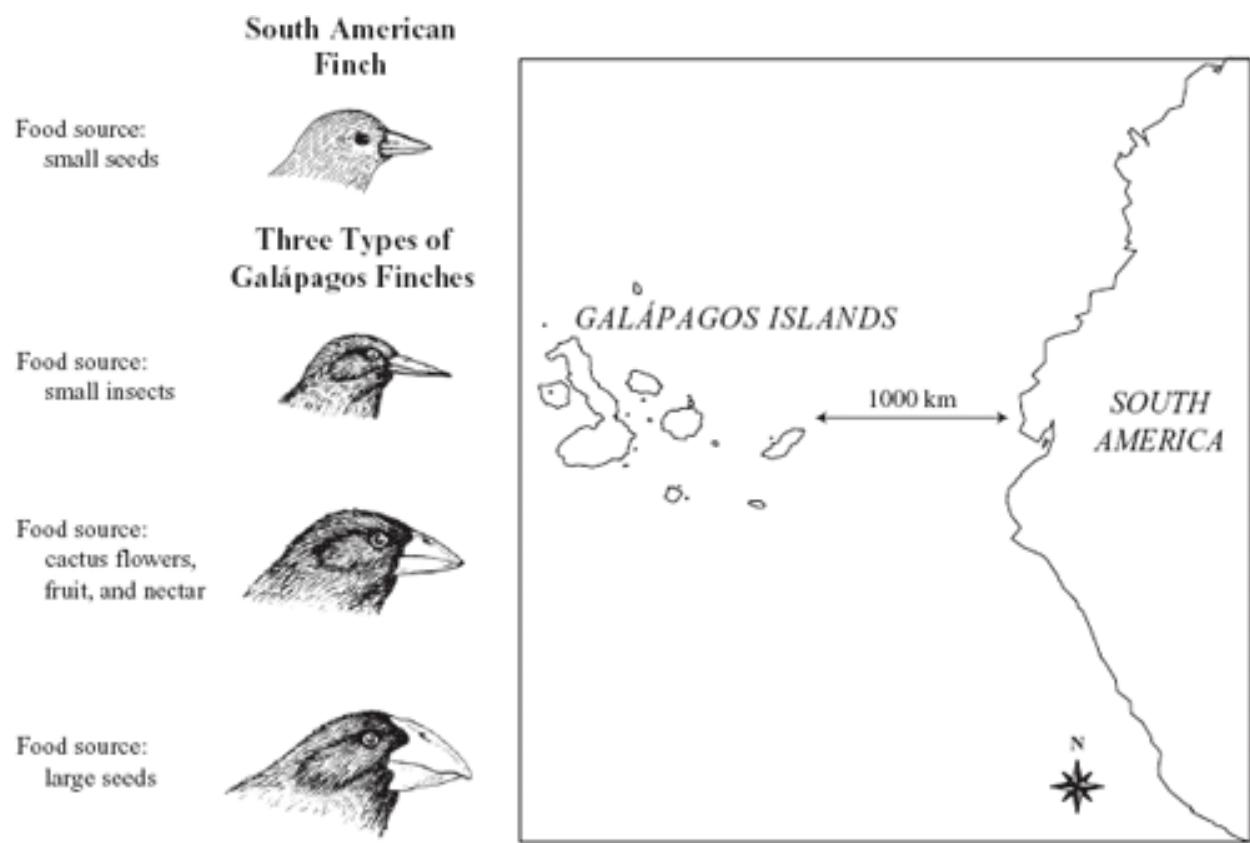


Figure 2: Figura 1. Fringilídeos em Galápagos.

Pássaros em Galápagos

Em sua viagem pelo mundo a bordo do Beagle, Charles Darwin descreveu um grupo de pássaros que habita as Ilhas Galápagos, arquipélago localizado a aproximadamente 900 km da costa do Equador (América do Sul). A variedade em tamanhos dos bicos chamou atenção: “*It is very remarkable that a nearly perfect gradation of structure in this one group can be traced in the form of the beak, from one exceeding in dimensions that of the largest gros-beak, to another differing but little from that of a warbler*”⁴

É interessante notar que a linguagem usada para denotar diferenças é eminentemente quantitativa (dimensions, largest, differing). Darwin não conduziu estudos quantitativos por razões práticas. Neste capítulo, simularemos o mesmo cenário empregando métodos estatísticos para comparar os pássaros.

Antes da publicação de A origem das Espécies, o caso dos fringilídeos (nome destas aves) já continha um embrião do processo de seleção natural. Na segunda edição, em 1845, ele especula sobre um grupo ancestral comum moldado por fins específicos:

“*Seeing this gradation and diversity of structure in one small, intimately related group of birds, one might really fancy that from an original paucity of birds in this archipelago, one species had been taken and modified for different ends.*”⁵

Darwin observou que a variedade dos bicos era adaptada à dieta de cada grupo: frutas, nozes, insetos. Os de bico pontudo conseguem comer frutas e arilo da semente do cacto, enquanto os de bico curto estraçalham a base do cacto e comem sua polpa.

A inspeção visual de um naturalista treinado foi capaz detectar essas nuances. Sob sua percepção, havia um total de 3 espécies em 4 ilhas: 1 na Ilha Charles, 1 na Ilha Albemarle e 1 nas ilhas James e Chatham. Inicialmente, notou que os pássaros eram semelhantes àqueles vistos no Chile. Darwin coletou 26 pássaros e os levou de volta para que um ornitólogo os estudasse com mais detalhe. O especialista (John Gould) sugeriu que os 26 pássaros representavam 12 espécies completamente novas, número que posteriormente passou para 25. Hoje, os taxonomistas sugerem um número de 15 espécies para os fringilídeos de Darwin.

Pensaremos como biólogos interessados em estudar quantitativamente o tamanho dos bicos. Usaremos estatística e probabilidades para testar hipóteses e fazer conclusões mais acuradas sobre as medidas, explorando diferenças entre os grupos de pássaros de Galápagos.

⁴ É bastante notável que uma graduação quase perfeita na estrutura desse grupo possa ser traçada na forma do bico, desde um excedendo as dimensões do maior dos pardais bico-gordo, até outro diferindo pouco do papa-amoras. Tradução livre. The Voyage of the Beagle (1839).

⁵ (...) [ao] ver esta graduação e diversidade em estrutura em um pequeno, intimamente relacionado grupo de pássaros, é possível imaginar que, a partir de poucos pássaros deste arquipélago, uma espécie foi escolhida e modificadas para certos fins. Tradução livre. Darwin, Charles (1845), Journal of researches into the natural history and geology of the countries visited during the voyage of H.M.S. Beagle round the world, under the Command of Capt. Fitz Roy, R.N (2nd. ed.), London: John Murray

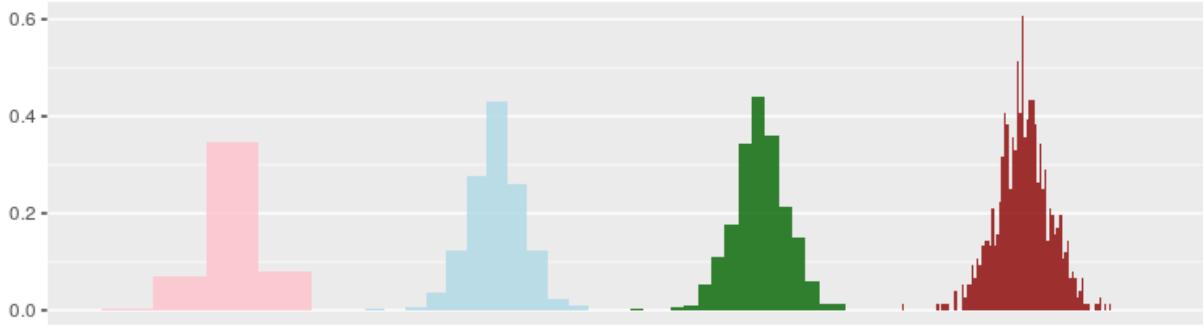


Figure 3: Figura 2. Distribuições binomiais para diferentes números de lançamentos com $p = 0.5$ (e.g: lançamento de uma moeda). Para $n > 1$, valores extremos indicam resultados com apenas caras (cauda à esquerda, 0000...) ou coroas (cauda à direita, 1111...)

A distribuição normal e um curioso teorema

Em trabalhos empíricos, é comum a suposição de que medidas de uma variável aleatória vêm de uma população com distribuição normal. A seguir, vamos estudar o comportamento dessa função probabilística.

Abraham de Moivre (26 May 1667 – 27 November 1754), sem financiamento exclusivo para estudos e pesquisa, prestava serviços secundários. Entre eles, cálculos de probabilidades em jogos de azar para clientes. Em 1733, de Moivre percebeu que as probabilidades de uma distribuição binomial, como o lançamento de moedas ($p(cara) = p(coroa) = 0.5$), aproximam-se de uma curva suave (contínua) à medida em que a quantidade de eventos aumenta.

A distribuição de Bernoulli descreve a possibilidade de dois eventos, como o lançamento de moedas. Tomando os valores discretos de caras (0) e coroas (1), a observação é 1 com probabilidade p e 0 caso contrário ($1 - p$). Para uma moeda honesta, temos uma distribuição probabilística uniforme sobre o domínio, $X = 0, 1$: $P(1) = P(0) = 0.5$.

Se somarmos distribuições de Bernoulli, obtemos a distribuição binomial. Cada observação é um conjunto de lançamentos. Tomando $p = 0.5$, resultados mais frequentes são números parecidos de caras (0) e coroas (1).

Para $n = 10$, é muito mais provável obter um número de caras próximo a 5 (centro das curvas) que um resultado com 9 ou 10 lançamentos iguais. É possível demonstrar que aumentar o valor de n faz com que a distribuição se aproxime da seguinte curva contínua:

De Moivre intuiu que a distribuição de binomiais com muitos lançamentos aproximava o de uma função suave. Ele buscava uma aproximação em termos da função exponencial [natural] e^x .

Mas quais os parâmetros da curva?

Primeiro, de Moivre deduziu a solução para o problema das moedas ($p = \frac{1}{2}$). A seguinte expressão geral descreve a probabilidade $P(x)$ correspondente à curva que procuramos, conhecida como *gaussiana*.

$$P(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

O número de Euler ($e \sim 2.72\dots$), que será melhor explicado no capítulo 5. A fórmula consiste em um fator, $\frac{1}{\sqrt{2\pi}}$ (aproximadamente 0.4), multiplicando o resultado da exponencial.

Em R, podemos definir:

```
>mgauss <- function(x) 0.4*exp((-1)*(x^2)/2)
```

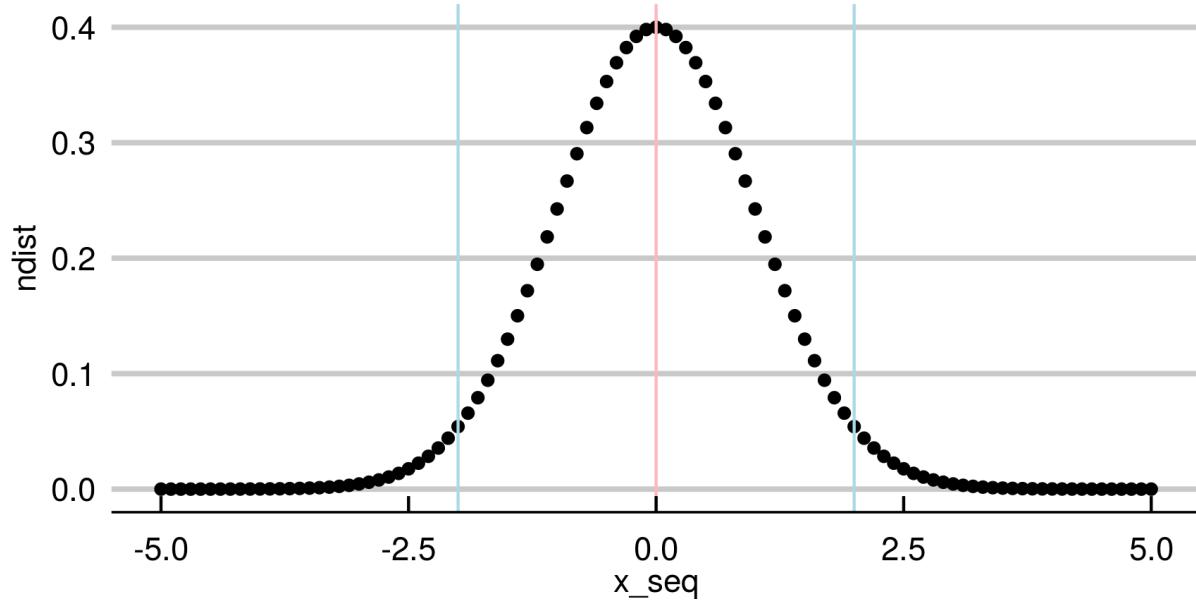
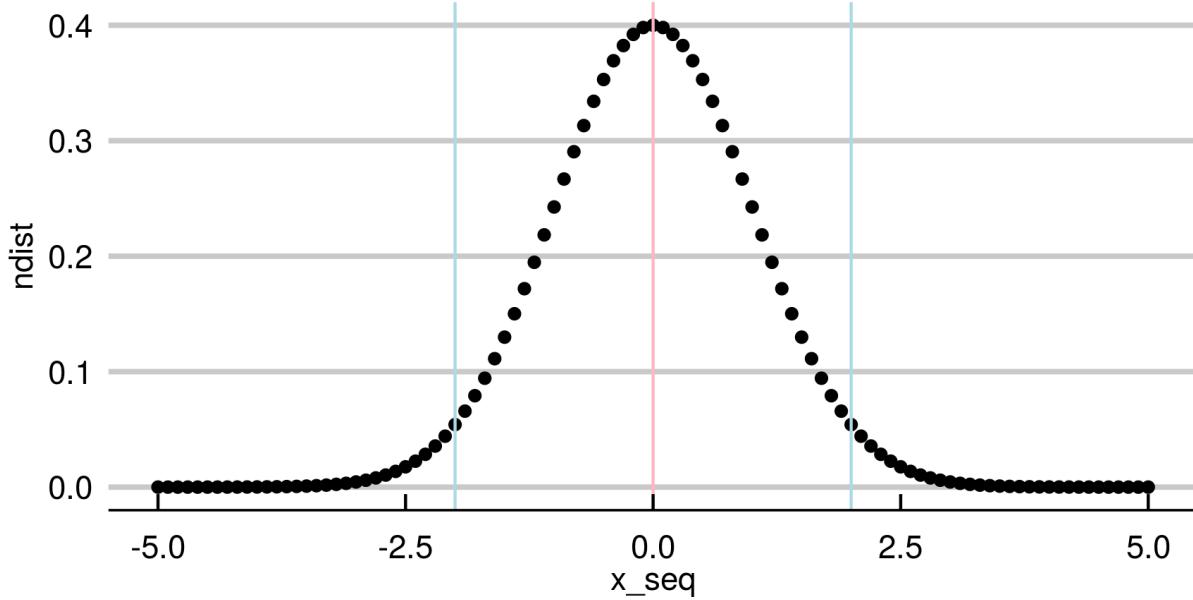


Figure 4: Figura 3: Distribuição normal (gaussiana), cujo formato lembra o de um sino

Em seguida, obter valores no intervalo $[-5, 5]$ e plotá-los:

```
>library(ggplot2)
>x_seq <- seq(-5,5,by = 0.1)
>ndist <- purrr::map(.f=mgauss,.x=x_seq) %>% unlist
>ggplot(data.frame(ndist,x_seq),aes(x=x_seq,y=ndist))+ 
  geom_point()+
  geom_vline(xintercept = 2,color="light blue")+
  geom_vline(xintercept = -2,color="light blue")+
  geom_vline(xintercept = 0,color="light pink")+
  theme_economist_white(gray_bg = F)
```



Observamos como a distribuição se dá a partir da equação.

Já que x^2 retorna apenas valores positivos, $-x^2$ sempre retorna negativos. Nossa função gera valores entre 0 e 1 exponenciando ($e \sim 2.718\dots$) a um fator negativo quadrático ($y = 0.4 * e^{-x^2/2}$).

Examinando o comportamento da equação, notamos que valores próximos ao centro ($x \sim \mu = 0$) fazem com que o expoente de se aproxime de 0, maximizando nossa função: $f(0) = 0.4 * e^{-x^2/2} = 0.4 * e^0 = 0.4$). O valor obtido (0.4) corresponde ao topo da curva no gráfico acima (linha rosa).

Observamos a curva se aproximar do máximo simetricamente para valores próximos de 0.

Isso reflete diretamente o fato de que valores próximos à média serão mais prováveis e valores extremos menos prováveis.

Para comparação: $f(2) = 0.4 * e^{-2^2/2} = 0.4 * e^{-2} = 0.4 * 0.135 \sim 0.05$ (linha azul). A probabilidade de se obter o valor médio ($x = 0, p \sim 0.4$) é oito vezes maior que a probabilidade de se obter o valor 2 ($x = 2; p = 0.05$).

O termo quadrático torna a distribuição simétrica para valores opostos em relação à média. $P(x) = P(-x)$. Como calculamos $P(2)$ antes, sabemos que: $P(-2) = P(2) = 0.05$ para $\mu = 0$. É igualmente provável encontrar valores duas unidades maiores ou duas unidades menores que a média. Esses pontos estão marcados por uma linhas azuis na figura.

Podemos trabalhar com curvas normais com centros (média μ) deslocados para a esquerda ($\mu < 0$) ou para a direita ($\mu > 0$), subtraindo o termo de x em nosso expoente. Além disso, diferentes variâncias (σ^2) refletem a frequência de valores longe da média e o quanto distante dela eles são. Visualmente, determina o tamanho da base do sino na ilustração (Figura 3).

Usamos a notação $N \sim (\mu, \sigma^2)$ para descrever uma distribuição gaussiana com média μ e variância σ^2 arbitrárias:

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

O valor $\frac{1}{\sqrt{2\pi}}$ surge como normalizador para avaliarmos a função como densidade de probabilidade. A integral de $-\infty$ a $+\infty$ deve ser 1. O valor π surge da integral de Gauss $\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}$ e decorre do fato de $2\pi i$ ser período da função e^x .

Poderíamos encontrar características desejáveis, como a simetria citada acima, em outras distribuições.

Então, por que usamos uma equação mais complexa?

Distribuições binomiais grandes e lançamentos de moedas são tão importantes?

O Teorema do Limite Central

A razão é o Teorema do Limite Central.

Se somarmos muitas distribuições de uma mesma família, a distribuição resultante se aproxima de uma normal. Sem muitas explicações, assumimos que isso era verdade para moedas.

Exemplos ajudam a ganhar intuição. Ao lançar um dado justo de 6 faces, temos probabilidade de $\frac{1}{6}$ em cada resultado.



Uma distribuição discreta uniforme, em que $P(1) = P(2) = P(3) = P(4) = P(5) = P(6)$ e definida para números naturais entre 1 e 6: $X \sim U_{discr}(1, 6)$.

A média para muitos lançamentos, ou valor esperado, é dado por:

$$E(X) = E(U(1, 6)) = (1 + 6)/2 = 3.5$$

Vamos fazer um experimento virtual usando 100 lançamentos de 11 dados.

O código em R para a seguir gera os dados e as visualizações de que precisamos:

```
>library(magrittr)
>library(ggthemes)
>library(ggplot2)
>source("aux/multiplot.R")
>set.seed(2600)
>n_plots <- 12
```

```

>dice_fun <- function(n){runif(n, min=0, max=6) %>% ceiling} # Random samples
>data_mat <- replicate(n=n_plots-1,dice_fun(100)) # Replicate
>data_mat <- cbind(data_mat,rowSums(data_mat)) # Sum

>plot_list <- vector("list", n_plots) # Plot each distribution
>plot_list <- apply(X=data_mat, MARGIN=2, FUN=function(x)
  ggplot(data.frame(obs=x),aes(x=obs)) +
    geom_histogram(binwidth = 0.2) +
    ylab("")+xlab("")+
    theme_economist())

>m_plot <- multiplot(plotlist = plot_list,cols=n_plots/3)
# Multiplot function available at:
# http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_(ggplot2)/

```

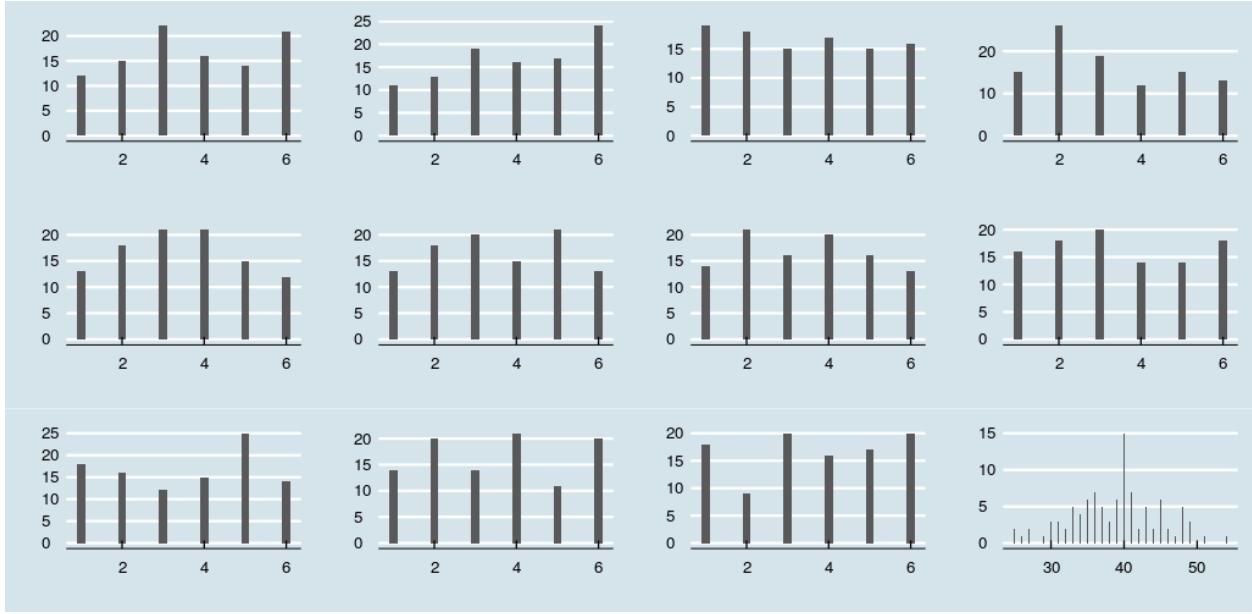


Figure 5: Soma de amostras ($n=100$) de 11 distribuições uniformes correspondentes ao lançamento de dados honestos de 6 faces. O resultado está na célula inferior à direita.

Notamos que as barras estão distribuídas com alturas bastante parecidas nas 11 primeiras células. A frequência esperada para cada valor é $\sim 1/6$ do total de 100 lançamentos. $Freq(X_i) \sim \frac{1}{6} * 100 \sim 16.66$
Algo interessante ocorre com a soma das distribuições (canto inferior direito).

O valor esperado é, como diz a intuição, a soma dos valores esperados em cada amostra:
 $E(X) = \sum_{i=1}^{11} E(U_i \sim (1, 6)) = 11 * 3.5 = 38.5$

O valor 38.5 corresponde aproximadamente ao centro da distribuição resultante (Figura 2, canto inferior direito) Entretanto, a distribuição muda de forma! Sem muito esforço, é notável a semelhança com a curva normal, com valores extremos menos frequentes e simetricamente afastados da média (valor esperado), que define o valor máximo.

É possível provar que a soma de muitas distribuições de uma mesma família converge para a distribuição normal em qualquer caso. Desde que estas sejam independentes. A esse resultado damos o nome de Teorema do Limite Central. A prova formal pode ser consultada em outro local,⁶ mas voltaremos a ela.
Este resultado tem uma util importânci para o estudo dos fenômenos naturais através de experimentos.

Ciência experimental e o Teorema do Limite Central

Muitos objetos de interesse para os cientistas são manifestações de fenômenos envolvendo múltiplos elementos. Um exemplo trivial está na cor da pele de seres humanos. Uma parte considerável depende do número de genes herdados relacionados à melanina. Eles se comportam de maneira aditiva.
Assim, cada variante de gene extra pode contribuir para a cor final com X unidades na escala para medir pigmentação.

A cor de um indivíduo será influenciada pela soma dessas distribuições, o que é análogo à matemática descrita para os lançamentos de dados.

⁶Yuval Filmus. 2010. Two Proofs of the Central Limit Theorem <http://www.cs.toronto.edu/~yuvalf/CLT.pdf>. Ela se dá mostrando a convergência de momentos entre a soma e gaussiana, um conceito que entenderemos no capítulo a seguir.



Podemos comparar grupos quanto a medidas fenotípicas finais (cor da pele) sem saber detalhes sobre as relações entre cada gene e seus mecanismos de expressão e regulação.

A distribuição final de melanina vem da soma de distribuições individuais semelhantes e tenderá a ser normal.

Como vimos, o mesmo é válido para quaisquer distribuições subjacentes: se elas forem gama, uniformes ou de Poisson, a distribuição da soma ainda tenderá à normalidade.

A figura 2 mostra a soma de distribuições uniformes para dados honestos, evidenciando que esta se aproxima de uma normal.

$$X \sim U_1(1, 6) + U_2(1, 6) + \dots + U_{11}(1, 6) = X \sim N(38.5, \sigma^2)$$

Vamos visualizar o mesmo processo para uma outra família de distribuições, gamma:

$$X \sim \gamma_1(\alpha, \beta) + \dots + \gamma_n(\alpha, \beta) = X \sim N(\mu', \sigma')$$

Para valores grandes de n:

```
>gamma_fun <- function(n){rgamma(n,1)}
>data_mat <- replicate(n=n_plots-1, gamma_fun(100))
>data_mat <- cbind(data_mat, rowSums(data_mat))

>plot_list <- vector("list", n_plots)
>plot_list <- apply(X=data_mat, MARGIN=2, FUN=function(x)
  ggplot(data.frame(obs=x), aes(x=obs)) +
    geom_histogram(binwidth = 0.2) +
    ylab("")+xlab("")+
    theme_economist())

>m_plot <- multiplot(plotlist = plot_list, cols=n_plots/3)
```

Novamente, verificamos que a soma começa a ser simétrica em torno da média, com formato de sinos (base alargada). Muitos fenômenos observáveis em nosso universo são naturalmente compostos por múltiplos elementos semelhantes. Especialmente em sistemas biológicos, há redundância de componentes e um objeto de interesse para cientistas é resultado da combinação de muitas variáveis subjacentes. O teorema do limite central permite que utilizemos distribuições normais para uma grande variedade de problemas. Ainda que as distribuições subjacentes sejam desconhecidas, o efeito resultante de uma grande combinação terá distribuição gaussiana em muitos casos.

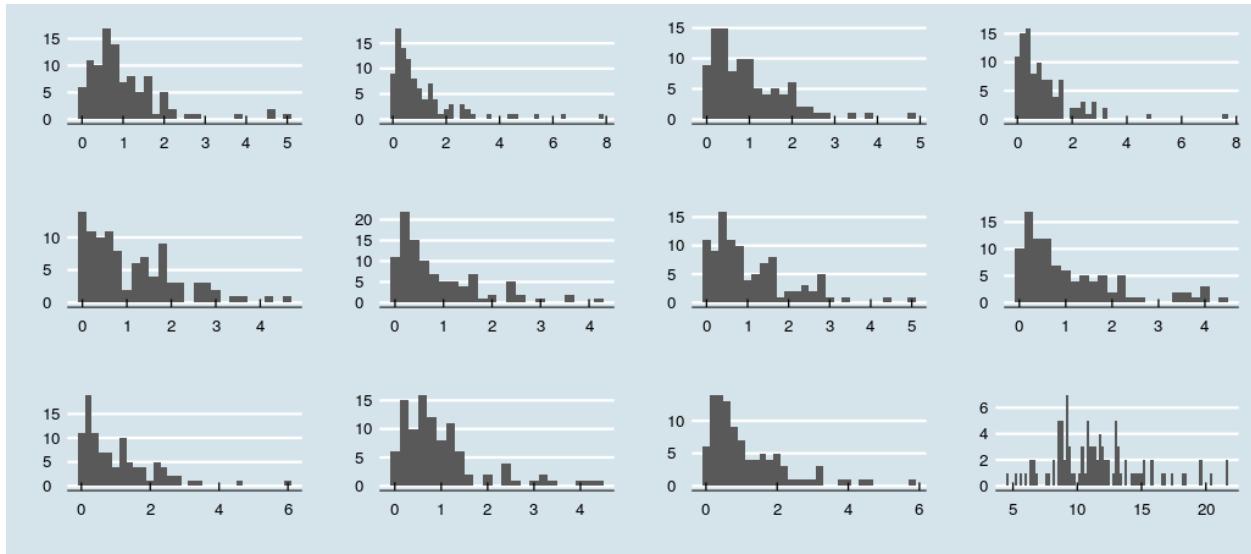


Figure 6: Soma de amostras ($n=100$) de 11 distribuições gama. O resultado está na célula inferior à direita. Função de densidade de probabilidade para distribuição gama: $f(x) = 1/\Gamma(\alpha) * \beta^\alpha * x^{\alpha-1} * e^{-\beta x}$, com $\alpha = \beta = 1$

A descoberta das equações que regem mecanismos de convergência em cenários probabilísticos foi uma grande evolução para as ciências experimentais.

Exercícios

1. Sobre a distribuição normal para uma variável aleatória, é verdadeiro (mais de uma possibilidade):
 - a. A soma da probabilidade de todos os valores possíveis é 1.
 - i. $\int_{-\infty}^{+\infty} f(x)dx = 1$.
 - b. É simétrica em relação à moda.
 - c. O valor esperado é dado por $1/\sigma\sqrt{2\pi}$.
 - d. 95% dos valores estão próximos à média.
 - e. Valores extremos são improváveis.
 - f. É unicamente determinada por variância σ^2 e média μ .
 - g. É contínua e diferenciável.
 - h. Amostras pequenas resultam em distribuições t.
2. Usando o comando “?Distributions” acesse algumas distribuições disponíveis na biblioteca de base do R.
 - a. Plote o histograma da soma de 100 distribuições X^2 (função rchisq; use n = 60).
 - b. Faça o mesmo procedimento para 100 distribuições de outra família e tamanho à sua escolha.
 - c. Obtenha os valores de skewness e kurtosis para essas distribuições. Uma distribuição normal padrão ($\sigma^2 = 1; \mu = 0$) possui skewness (assimetria) de 0 e kurtosis (frequência de valores mais extremos) de 3. Quais os encontrados por você?
 - d. Cite dois fenômenos naturais cuja distribuição estatística é conhecida e qual a distribuição correspondente.

Parte 2 - Darwins's Finches e um teste paramétrico

Mostraremos como a contribuição individual de genes com efeitos aditivo de distribuição uniforme resulta em medidas aproximadamente normais para os bicos das aves.

Vamos simular as medidas de bicos em 4 amostras ($n=150$) de pássaros.

O tamanho dos bicos é dado pelo efeito aditivo de muitos genes semelhantes, portanto esperamos que sua distribuição seja normal pelo Teorema do Limite Central.

Uma cópia do gene adiciona x milímetros ao tamanho final. O valor de x é sorteado de uma variável aleatória de distribuição uniforme, $X \sim U(0, 1)$.

Pássaros têm um número fixo de n de genes aditivos em cada amostra, sorteado no intervalo entre 80 e 100. A medida final dos bicos é dada pela soma efeitos dos n genes. Esse número é fixo em cada população e varia entre populações.

Para simular os dados com as condições acima:

```
>library(magrittr)
>library(ggthemes)
>library(ggplot2)
>set.seed(2600)

>n_birds <- 150 # sample_size
>genes_low <- 80 # lower bound on number of genes
>genes <- 100 # upper bound on number of genes
>n_islands <- 4 #samples

>unif_sum <- function(genes){
  replicate(n = genes,
            expr = runif(100, min=0, max = 1)) %>%
  rowSums
}

>generate_pop <- function(n_pop,n_genes){
  replicate(n=n_pop,
            expr = unif_sum(n_genes) %>% mean)
}

>galapagos_birds <- purrr::map(.f = function(x) generate_pop(n_pop=n_birds,
                                                               n_genes = x),
                                 .x = runif(n=n_islands, genes_low, genes) %>% ceiling) %>%
  unlist %>% matrix(nrow=n_birds,byrow=F) %>%
  data.frame
```

Como esperado, verificamos que o histograma das medidas finais se aproximam de uma gaussiana.

```
>my_alpha <- 0.5
>my_bins <- 50
>ggplot(data=galapagos_birds,aes(x=X1))+
  geom_histogram(alpha=my_alpha,bins = my_bins)+
  geom_histogram(data=galapagos_birds,aes(x=X2),fill="dark blue",
                 alpha=my_alpha,bins = my_bins)+
  geom_histogram(data=galapagos_birds,aes(x=X3),fill="dark red",
                 alpha=my_alpha,bins = my_bins)+
  geom_histogram(data=galapagos_birds,aes(x=X4),fill="dark green",
                 alpha=my_alpha,bins = my_bins)+
```

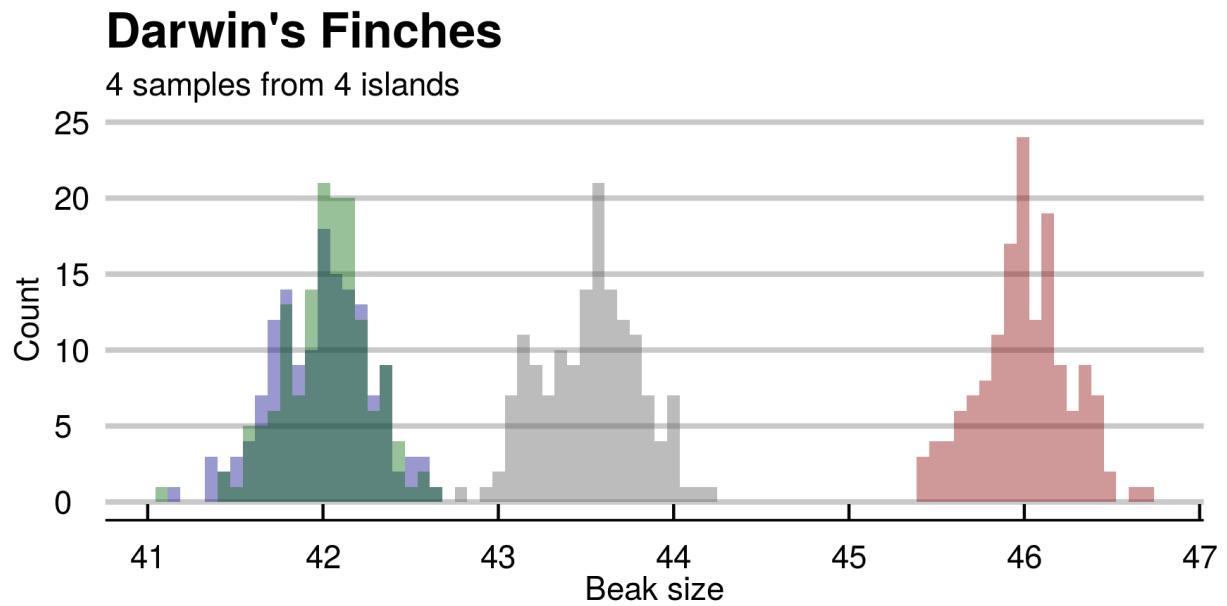


Figure 7: Figura 4. Distribuição das medidas de bicos em populações simuladas para genes com efeito aditivo.

```
xlab("Beak size") + ylab("Count") +
  ggtitle("Darwin's Finches", subtitle = "4 samples from 4 islands") +
  theme_economist_white(gray_bg = F)
```

Os números aleatórios gerados usando a semente sugerida (set.seed(2600), linha 4 do código acima) são semelhantes à suposição de Darwin: 4 ilhas (amostras) e três espécies (distribuições de bicos). Notamos que há duas amostras (verde, azul) de medidas bastante parecidas e outras duas separadas (cinza, vermelho). Supondo que medimos os bicos de algumas aves, como saber se os grupos são diferentes? Calculando as diferenças entre distribuições, podemos inferir se duas amostras têm o mesmo número de genes subjacentes! Para isso, usaremos um racional e algumas ferramentas novas.

Testes de hipótese

Filósofos da ciência estudam características no modus operandi de outros estudiosos. O que há em comum entre os procedimentos empregados por biólogos e geólogos? O que distingue Charles Darwin e Paul Dirac de John Dee e Edward Kelley? O que funciona em áreas distintas do conhecimento humano?

Adotamos a denominação coletiva de “ciências” para algumas áreas do conhecimento. Ainda, associamos a elas características em comum nos procedimentos e na estrutura interna. De alguma forma, científicidade comunica credibilidade. Nas últimas décadas, filósofos discutiram a validade do problema de demarcar ciência de pseudociência e não-ciência.⁷ Neste capítulo, vamos nos ater a um paradigma conceitual mais antigo e indiscutivelmente influente.

O método hipotético-dedutivo foi popularizado no século XX como uma bandeira de identificação associada ao trabalho científico. Um ciclo que consiste em formular teorias, desenhar experimentos, testar hipóteses falseáveis, verificar resultados e repetir o processo de forma iterativa.

O racional em usar hipóteses testáveis é de que proposições válidas sobre um sistema contém informações que ajudam a prevê-lo. Assim, “faz sol ou não amanhã” é uma proposição inútil, enquanto “faz sol amanhã” é uma proposição útil. Note que “faz sol amanhã” é uma hipótese testável (falseável), enquanto “faz sol ou não amanhã” é uma hipótese verdadeira independente das observações.

O exemplo é grosseiro, porém alguns ramos do conhecimento humano produziram hipóteses não-falseáveis. K. Popper, líder da revitalização do método hipotético dedutivo no século passado, atacou severamente o materialismo dialético de Karl Marx, assim como a teoria de evolução por seleção natural de Charles Darwin. Marx previu que a revolução aconteceria em uma nação industrializada através da classe operária e outros eventos que não se concretizaram. Seus seguidores usaram hipóteses *ad-hoc* para justificar a observação. A teoria da evolução por seleção natural de Darwin era amparada em muitos exemplos de reprodução impossível (e.g. recomposição da trajetória evolutiva em fósseis). A psicanálise também sofreu duras críticas, em virtude da irrefutabilidade de seus pilares centrais.

Para Popper, a dificuldade em gerar hipóteses testáveis e falseáveis sinalizava uma evidente fragilidade nas teorias, as quais não empregariam métodos científicos em seus avanços.

Uma maneira de formalizar essa ideia, incorporando o uso de ferramentas quantitativas, é através de probabilidades. Calculamos a probabilidade associada a observações, considerando o cenário de uma hipótese (falseável). Esse racional adequa ferramentas matemáticas robustas à plataforma epistemológica de Popper, sendo um modelo dominante de produção em ciências experimentais.

Em geral, os pesquisadores formulam uma hipótese base, chamada hipótese nula, que descreve o cenário menos interessante para o trabalho. Por exemplo, se estamos comparando dois grupos, A e B, quanto a uma intervenção, a hipótese nula costuma declarar que os grupos são iguais.

Queremos estudar o tamanho dos bicos de pássaros das ilhas A e B. A hipótese nula natural é: Não há diferença entre os bicos dos pássaros do tipo A e B.

Medimos o bico de alguns pássaros dos dois grupos e calculamos a probabilidade de encontrarmos essas medidas considerando que A e B são iguais. Se essa probabilidade for muito baixa, rejeitamos nossa hipótese.

Estruturando os passos:

1. Definimos a hipótese nula (H_0) e pelo menos uma hipótese alternativa(H_1).

- H_0 : Pássaros das ilhas A e B possuem bicos de tamanho igual.
- H_1 : Os pássaros possuem bicos de tamanho diferentes.

Então, podemos fazer um experimento, coletando medidas experimentais para o comprimento dos bicos. Essas medidas, junto a premissas matemáticas razoáveis, permitem especular: qual a probabilidade p de obter nossas observações considerando distribuições iguais entre A e B? Isto é, considerando H_0 verdade, nossos resultados seriam raros ou comuns?

⁷Massimo Pigliucci - Philosophy of Pseudoscience: Reconsidering the Demarcation Problem

Caso p seja menor que um limiar pré-definido (convencionalmente, 0.05), rejeitamos H_0 . A probabilidade é muito pequena para H_0 ser verdade.

A domínio dos procedimentos hipotético-dedutivos nas ciências produziu resultados interessantes. Especialmente no eixo de trabalho denominado por Thomas Kuhn de “ciência normal”, focada no acúmulo de evidências e testagem de hipóteses. O fantasma de desenhar um experimento imparcial com possibilidade de falha aguçou a percepção de pesquisadores para a falibilidade de ideias. O grau de sofisticação em reproduzibilidade de procedimentos foi amplificada.

Nota

Usamos o limite inferior de 0.05 como critério para rejeitar a hipótese nula, o que pode parecer arbitrário. E é. Os valores p eram interpretados de acordo com sua magnitude e estatística com base em que foram calculados. Foi Ronald Fisher, em Statistical Methods for Research Workers (1925), quem propôs (e posteriormente popularizou) o número: “The value for which $p = 0.05$, or 1 in 20, is 1.96 or nearly 2; it is convenient to take this point as a limit in judging whether a deviation ought to be considered significant or not.”⁸

Teste t e distribuição t de Student: Um exemplo prático

Para testar estatisticamente se as medidas são diferentes, executaremos um teste t para comparação dos grupos.

A distribuição t surge quando queremos entender quão improváveis são nossas estimativas (μ') supondo uma média real hipotética (μ) de origem em uma variável de distribuição normal desconhecida.

Exemplo: Medimos os bicos de 30 pássaros. Obtivemos média amostral $\mu' = 38$ mm e desvio-padrão $\sigma' = 0.3$ mm.

Problema: Supondo que a média real (μ) da população é de 40 mm, qual é a probabilidade de obtermos $\mu' = 38$ mm em uma amostra aleatória, como aconteceu em nosso experimento?

Entender a imprecisão da estimativa de uma média foi o eixo principal para a descrição dessa distribuição por William Gosset. Sob o pseudônimo Student, o estatístico, que trabalhava para a fábrica de cerveja Guinness, publicou na Biometrika (1908) o famoso artigo *The probable error of a mean*.

Para entender a imprecisão, necessitamos de uma medida da dispersão dessas medidas.

Assumimos amostras retiradas de uma variável aleatória com distribuição normal com média μ e desvio-padrão σ . Podemos retirar j amostras de tamanho n e calcular a média dessas amostras $\mu'_1, \mu'_2, \dots, \mu'_j$. As médias amostrais μ' são estimativas da média real μ .

Qual a dispersão das estimativas $\mu'_1, \mu'_2, \dots, \mu'_j$?

Para um conjunto de estimativas $\mu'_1, \mu'_2, \dots, \mu'_j$, chamamos de **erro padrão** (*standard error of the mean*) o desvio-padrão populacional σ dividido pela raiz quadrada do tamanho da família de amostras em questão ($std.err. = \sigma/\sqrt{n}$). Como não sabemos o desvio-padrão na população, aproximamos usando o valor do desvio-padrão σ' amostral.

Student propôs o uso de uma quantidade para estimar a probabilidade de uma estimativa μ' dado um centro hipotético μ .

Essa quantidade pivotal é a razão entre (1) distância das estimativas e média real, $\mu' - \mu$, e (2) o erro padrão. A estatística t:

$$t = \frac{Z}{s} = (\mu' - \mu)/\frac{\sigma}{\sqrt{n}}$$

⁸O valor [da estatística z em uma curva normal] para o qual $p = 0.05$, ou 1 em 20, é de 1.96 ou aproximadamente 2; é conveniente pegar esse ponto como um limite ao julgar quando um desvio deve ser considerado significante ou não.

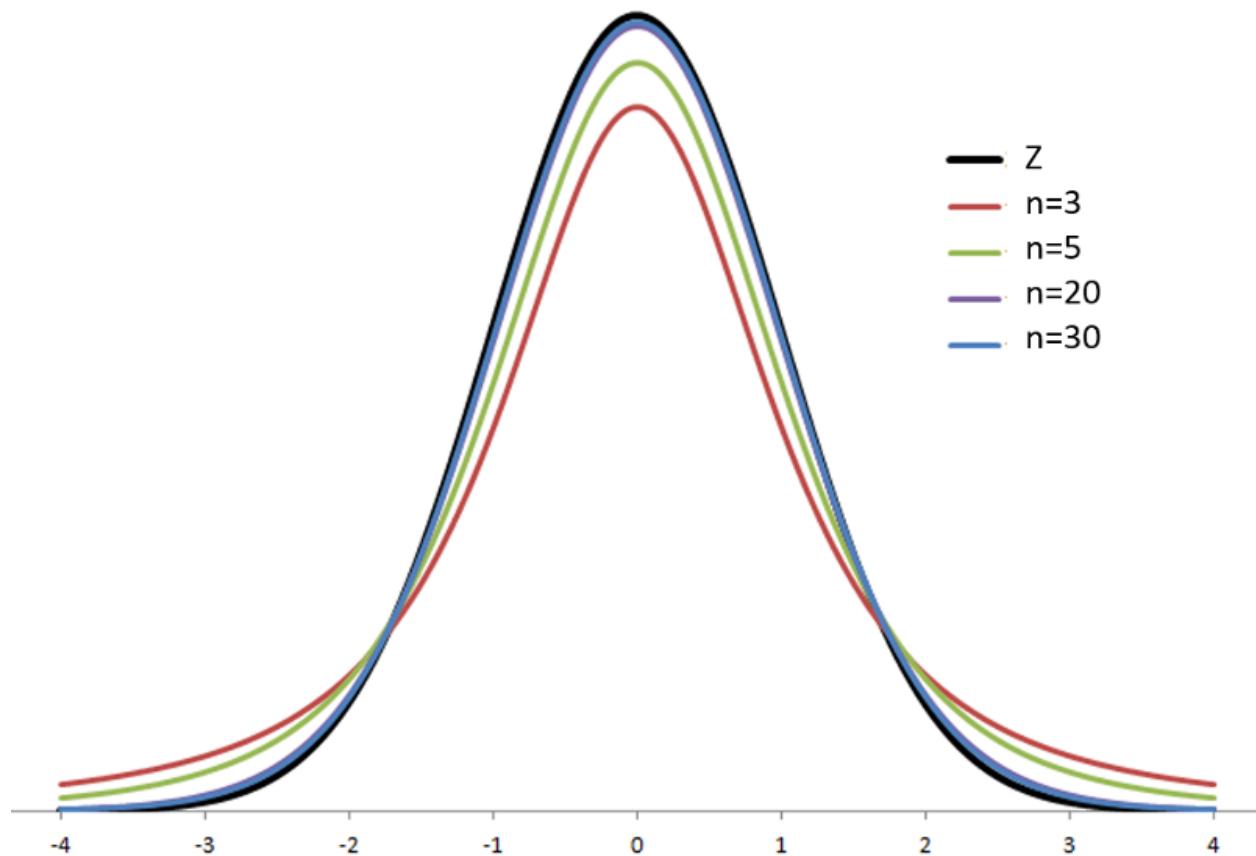
Assim, a estatística t para nosso exemplo ($\mu' = 38$; $\mu = 40$; $n = 30$; $\sigma' = 0.3$) é:

$$t = \frac{(38 - 40)}{\frac{0.3}{\sqrt{30}}}$$

Student (Gosset) mostrou que essa estatística segue uma distribuição probabilística (t de Student) definida por:

$$f(t) = \frac{1}{\sqrt{\nu} B(\frac{1}{2}, \frac{\nu}{2})} \left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

B é a função Beta⁹ e v são graus de liberdade. Possui densidade parecida com a da distribuição normal, porém com probabilidades maiores para valores extremos. O parâmetro ν (graus de liberdade) expressa essa característica. Empiricamente é estimado pelo tamanho das amostras usadas na estimativa de μ' . Associamos uma amostra (tamanho n) retirada de uma população normal (tamanho arbitrariamente alto, $n \rightarrow \infty$) a uma distribuição t com $n - 1$ graus de liberdade. Em nosso exemplo, $n = 30$, então $\nu = n - 1 = 29$.



Maiores valores correspondem a amostras maiores e fazem com que a distribuição t se aproxime de uma distribuição normal. Em um caso extremo, temos $n_{samples} = n_{pop}$ e as amostras são idênticas à distribuição de origem.

Sabendo a estatística t (-36.51) e os graus de liberdade para nossa família de amostras ($\nu = 29$), podemos usar a expressão $f(t)$ para saber a probabilidade de obtermos nossa média 38 mm numa amostra ($n = 30$) se a média populacional for de 40 mm.

⁹A função Beta é aceita dois argumentos(x, y) e seu resultado é a razão entre (1) produto das funções $\Gamma(x)\Gamma(y)$ e (2) função gama da soma $\Gamma(x+y)$. A função Γ generaliza o conceito de fatoriais (produto dos antecessores).

Para tanto, somamos as probabilidades de valores extremos menores que a estatística t fornecida.

$$\int_{-\infty}^{-36.51} f(t)dt$$

Em R, a função nativa *pt* faz o trabalho sujo:

```
>pt(-36.51, df = 29)
[1] 4.262e-26
```

Esse valor reflete a probabilidade de valores t negativos mais extremos (menores) que os nossos ($t < -36.51$).

Teste bicaudal

Parece ser nosso valor p, porém precisa de um ajuste: queremos saber a probabilidade associada a obter valores tão extremos em geral, não nos restringindo a valores extremamente menores.

Uma vez que a distribuição é simétrica, a cauda à esquerda (negativos) é idêntica à cauda à direita (positivos). Valores extremos (negativos ou positivos) em relação à média são duas vezes mais prováveis que valores negativamente extremos.

Consideramos significativos valores t muito maiores (direita) ou menores (esquerda) que a média. Então, nosso limiar deve ser robusto à possibilidade de extremos maiores que a estatística t simétrica positiva.

O valor $t = 36.51$ seria a estatística resultante de uma amostra com média simétrica (42 mm) em relação à média (40 mm). Recorde-se de que a medida original foi 38 mm.

$(t_{min} = -36.51; t_{max} = 36.51)$.

Ao fazer esse ajuste, chamamos o teste de bicaudal.

Sabendo da simetria na distribuição t, podemos fazer então usar o seguinte truque:

```
> 2*pt(-36.51, df = 29)
[1] 8.524e-26 # valor p 'bicaudal'
```

Não é possível calcular diretamente as probabilidades para $t = 36.51$, pois o R aproxima a integral acima ($p \sim 1 - 4.262^{-26}$) ~ 1 .

```
> pt(36.51, df = 29)
[1] 1
```

Nota

Uma percepção errônea comum sobre a distribuição t é de que ela descreve amostras pequenas retiradas de uma população com distribuição normal. Qualquer amostra retirada de uma variável de distribuição normal terá, por definição, distribuição normal, ainda que seja composta por 1 ou 2 observações. O que segue distribuição t é a quantidade pivotal descrita acima.

Na sessão IX do artigo, Student (Gosset) demonstra como seu insight pode ser usado para testar o efeito de isômeros da escopolamina como indutora do sono.¹⁰ São usadas duas amostras (levo e dextro hidrobromido de hyoscyamina).

Usando dados de 10 pacientes que usaram ambas as substâncias e medidas da quantidade adicional de horas de sono observadas, “Student” calcula: (1) a probabilidade dos dados supondo média 0 em cada grupo e (2) a probabilidade dos dados supondo que a diferença das médias é 0.

O primeiro procedimento é idêntico ao que realizamos com a medida dos bicos e é chamado teste t de amostra única (*one sample t-test*). Hipotetizando um valor para a média (e.g. $\mu_{bico} = 40mm$; $\mu_{sonoadicional} = 0horas$), calculamos as probabilidades de nossa estimativa.

¹⁰https://atmos.washington.edu/~robwood/teaching/451/student_in_biometrika_vol6_no1.pdf

Additional hours' sleep gained by the use of hyoscyamine hydrobromide.

Patient	1 (Dextro-)	2 (Laevo-)	Difference (2-1)
1.	+ .7	+ 1.9	+ 1.2
2.	- 1.6	+ .8	+ 2.4
3.	- .2	+ 1.1	+ 1.3
4.	- 1.2	+ .1	+ 1.3
5.	- 1	- .1	0
6.	+ 3.4	+ 4.4	+ 1.0
7.	+ 3.7	+ 5.5	+ 1.8
8.	+ .8	+ 1.6	+ .8
9.	0	+ 4.6	+ 4.6
10.	+ 2.0	+ 3.4	+ 1.4
	Mean + .75	Mean + 2.33	Mean + 1.58
	S. D. 1.70	S. D. 1.90	S. D. 1.17

Figure 8: Retirado de The probable error of a mean, pag. 20. Os dados estão disponíveis na biblioteca de base do R, sob o nome 'school'.

O segundo procedimento é chamado de teste t de amostras independentes. Hipotetizamos um valor para diferença de médias entre duas populações ($\mu_a - \mu_b = 0$) e calculamos a probabilidade de nossa estimativa. Exemplo prático: existe diferença de peso entre os bicos dos pássaros A e B?

Aplicações

Retornando ao nosso exemplo de Galápagos, faremos um teste t de amostras independentes.

1. As medidas em A e B são amostras de variáveis aleatórias com distribuição normal.
2. Definimos a hipótese nula e pelo menos uma hipótese alternativa.
 - H_0 : Pássaros das ilhas A e B possuem bicos de tamanho igual.
 - $\mu_a - \mu_b = 0$
- b. H_1 : Os pássaros possuem bicos de tamanho diferentes.

O procedimento é semelhante ao anterior. Calculamos uma quantidade intermediária que segue distribuição t usando a estimativa amostral da diferença e erro padrão associado. Então, podemos especular: qual a probabilidade p de alguém obter nossas observações considerando distribuições de médias iguais ($\mu_a = \mu_b$)? Esse teste infere a probabilidade para as populações de onde saíram as amostras.

Caso p seja menor que um limiar arbitrariamente pré-definido (convencionalmente, 0.05), rejeitamos H_0 . A probabilidade de observarmos os dados é pequena se H_0 for verdade.

Obtemos o valor p somando os valores de probabilidades correspondentes às diferenças obtidas ou valores mais extremos. Caso a diferença entre valores seja grande, o valor da estatística crescerá. Isso implica uma baixa probabilidade de observar aqueles resultados se as amostras fossem semelhantes (vindas da mesma distribuição).

Teste t de Student com R

Vamos computar um teste t para 2 amostras independentes. A estatística t é calculada com algumas mudanças. Os graus de liberdade são somados e o erro padrão (dispersão das estimativas) é balanceado através da média ponderada (pelos graus de liberdade, $n-1$) entre amostras.

$$t = \frac{X_1 - X_2}{\sigma_{pooled} \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

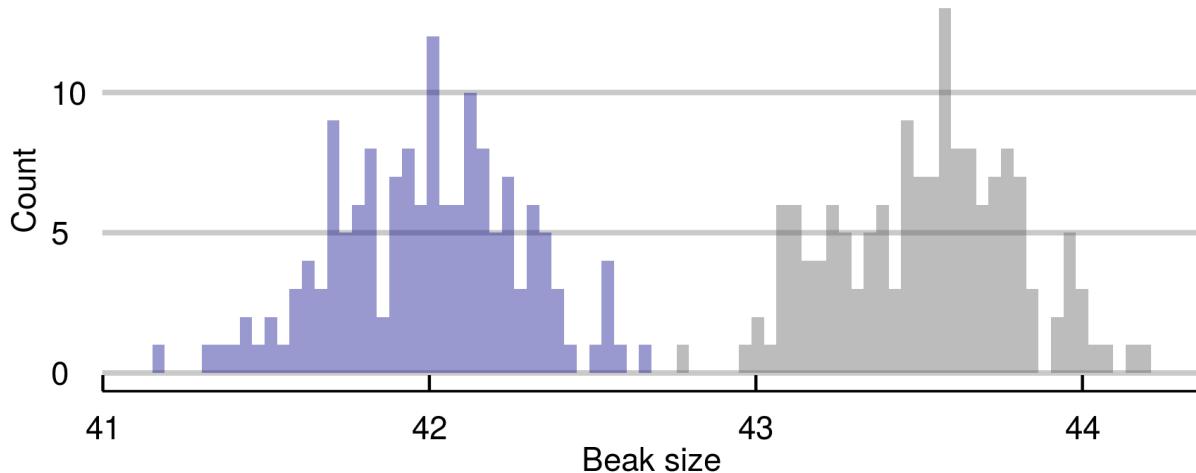
$$\sigma_{pooled} = \sqrt{\frac{(n_1 - 1)\sigma_1^2 + (n_2 - 1)\sigma_2^2}{(n_1 - 1) + (n_2 - 1)}}$$

Considerando $(n_1 - 1) + (n_2 - 1)$ graus de liberdade, calculamos a estatística t e o valor p correspondente para nossos graus de liberdade. Usando as amostras criadas anteriormente, correspondentes às barras cinza (A) e azul(B), vamos plotar os histogramas.

```
>ggplot(data=galapagos_birds,aes(x=A))+  
  geom_histogram(alpha=my_alpha,bins = my_bins)+  
  geom_histogram(data=galapagos_birds,aes(x=B),fill="dark blue",  
                 alpha=my_alpha,bins = my_bins)+  
  xlab("Beak size")+ylab("Count")+\n  ggtitle("Darwin's Finches",subtitle = "Samples A and B")+\n  theme_economist_white(gray_bg = F)
```

Darwin's Finches

Samples A and B



```
# Ajustes nos dados  
>a <- galapagos_birds$X1  
>b <- galapagos_birds$X2  
>sd_a <- sd(a) #desvio-padrão  
>sd_b <- sd(b)
```

Aqui, ao invés de comparar as estimativas das médias de distribuição t para amostras A e B.

Calculamos a (1) Diferença esperada na vigência da hipótese nula ($diff_{H_0} = 0$), (2) estimativa da diferença ($diff = \mu_A - \mu_B$), graus de liberdade (df) e erro padrão balanceado (se_{pooled}) para a distribuição das diferenças de médias.

```
>expected_diff <- 0
>mean_diff <- mean(a) - mean(b) #diferença de medias

>df_pool <- length(a) + length(b) - 2 # graus de liberdade balanceados
>sd_pool <- sqrt(((length(a) - 1) * sd_a^2 + (length(b) - 1) * sd_b^2)/
                  df_pool) # desvio padrao balanceado
```

A estatística t correspondente à diferença observada, considerando uma distribuição t com os parâmetros calculados acima.

```
# Diferenca dividida por erro padrao
>t <- (mean_diff - expected_diff)/ (sd_pool * sqrt(1/length(a) + 1/length(b))) # t-statistic
```

Valor p para hipótese bicaudal (resultados extremos considerando a possibilidade de a diferença ser maior ou menor que 0):

```
>p <- 2*pt(-abs(t), df = df_pool)
```

Finalmente, agregando o sumário dos resultados (médias A e B, diferença verificada, estatística t resultante, valor p):

```
>result <- c(mean_diff, t, p, mean(a), mean(b))
>names(result) <- c("Difference of means", "t", "p-value", "Mean A", "Mean B")
>result
Difference of means           t
1.533321e+00    4.728513e+01
      p-value        Mean A          Mean B
1.532661e-140    4.352244e+01    4.198912e+01
```

Obtivemos um valor p significativo ($p < 0.001$) usando $n = 150$. Os graus de liberdade são 149 ($150 - 1$) em cada amostra, sendo 298 ao total.

Sendo uma linguagem voltada à estatística, R possui em sua biblioteca de base uma função para automatizar o processo em 1 linha:

```
>t.test(a,b,var.equal = T)
Two Sample t-test / data: a and b
t = 47.285, df = 298, p-value < 2.2e-16
Alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval: 1.469506 1.597136
Sample estimates:
mean of x mean of y
43.52244 41.98912
```

Estatística t e graus de liberdade apresentados pela implementação nativa do R(`t.test`) são idênticos aos que encontramos realizando o procedimento passo a passo.

Ao invés do valor exato ($p = 1.53^{-140}$), recebemos a informação de que $p < 2.2^{-16}$.

Dante do valor p obtido, concluiríamos que a distribuição dos dados como observada é improvável se for verdade a hipótese nula H_0 de que a diferença entre amostras é 0.

Exemplo de relatório

A diferença estimada entre tamanho médio dos bicos entre amostras A e B foi significativamente ($p < 0.05$) diferente de 0 ($t=47.28$; $df = 298$).

	Amostra A	Amostra B	valor p
Média(μ)	43,52	41,99	<0,001
Desvio-padrão(σ)	0,28	0,28	

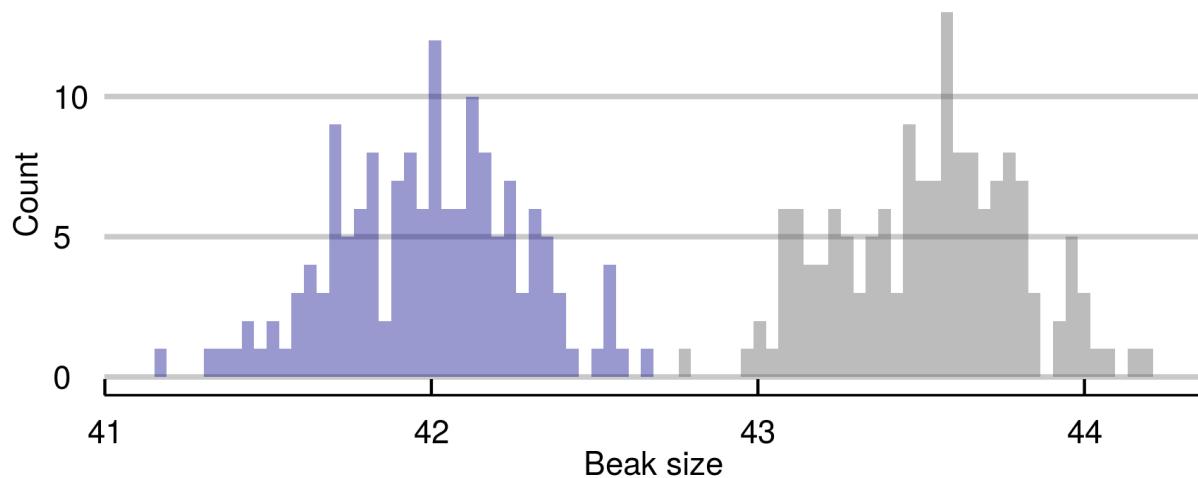
Report example

The estimated difference of beak mean sizes among samples A and B was significantly ($p<0.05$) different from zero ($t = 47.28$, $df = 298$)

	Amostra A	Amostra B	valor p
Mean (μ)	43,52	41,99	<0,001
Std. Dev. (σ)	0,28	0,28	

Darwin's Finches

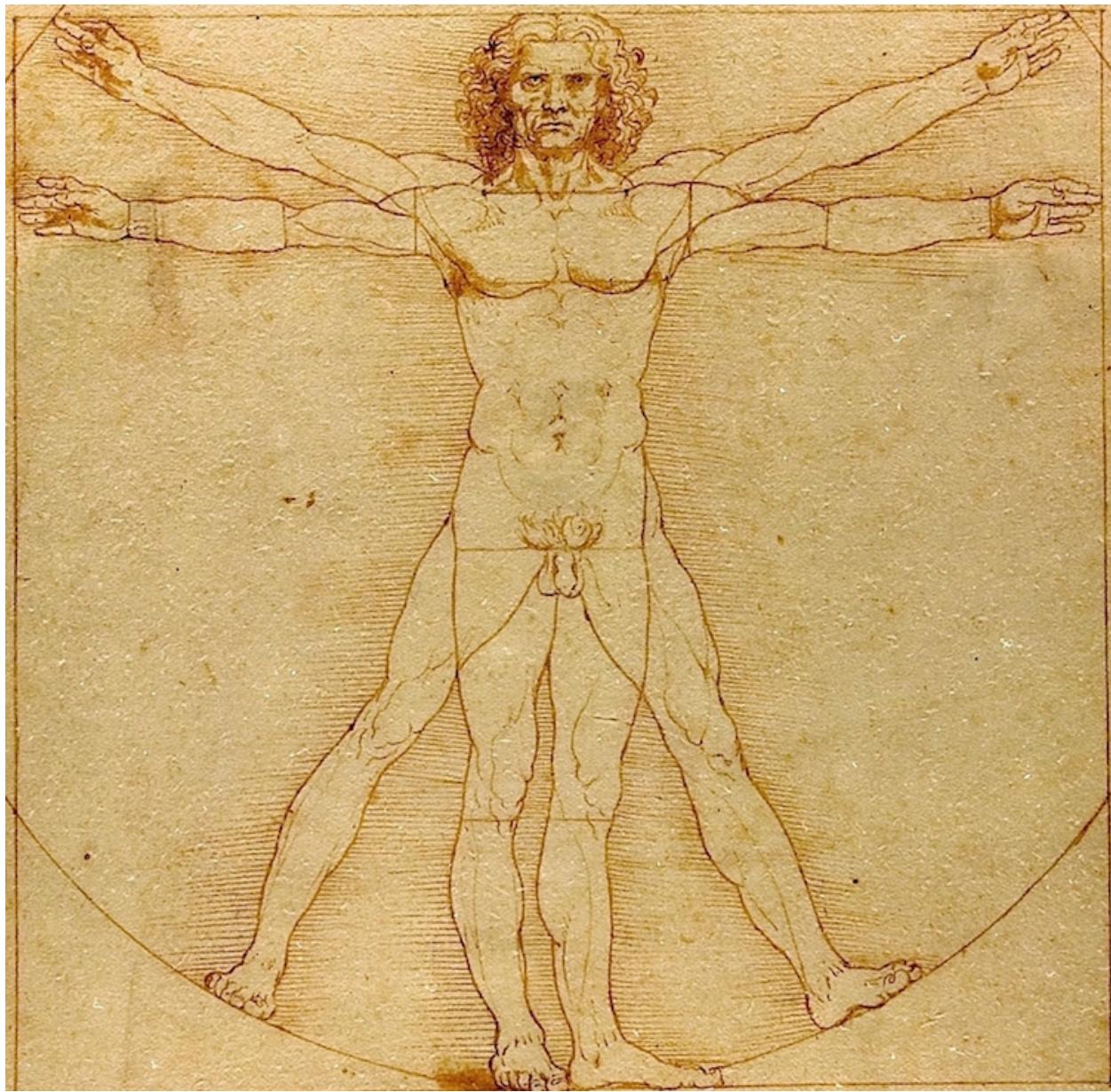
Samples A and B



Nota

Exercícios

1. Usando o dataset simulado no capítulo:
 - a. Execute teste T para cada par de amostras
 - b. Quais testes apresentam $p < 0.05$?
 - i. Descreva estatística t, graus de liberdade e valor p.
 - * 1. Como são os graus de liberdade dos diversos testes?
 - * 2. Esses valores eram esperados para nossas amostras?
 - ii. Usando ggplot, plote histogramas para todos os pares comparados em apenas um painel. Dica: grid.arrange
 - iii. Plote boxplots para uma das comparações.
 - iv. A partir do gráfico anterior, adicione uma camada com violin plots (geom_violin) transparentes ($\alpha=0$).
 2. Usando o dataset iris
 - a. Escolha duas espécies e duas medidas.
 - b. Execute testes t para ambas as medidas
 - c. Reporte os resultados em uma tabela, incluindo média e desvio-padrão de ambas as medidas nas duas espécies.
 3. Os dados usados por Student para escopolamina estão incluídas na biblioteca de base do R.
 - a. Examine os dados invocando “sleep”: >sleep
 - i. Plote histogramas para as medidas em ambos os grupos
 - ii. Execute um teste t supondo média populacional zero ($\mu = 0$).
 - iii. Execute um teste t entre amostras, supondo a mesma média ($H_0 : \mu_1 = \mu_2$).
 4. Gerando a distribuição t:
 - a. Simule um conjunto de muitas medidas (sugestão: 100,000) a partir de uma distribuição normal ($\mu = 0, \sigma = 1$).
 - b. Retire 200 amostras de $n=30$ e salve as 200 médias (função sample).
 - c. Divida os valores por pelo erro padrão, $\frac{\sigma}{\sqrt{n}}$.
 - d. Retire 200 amostras de uma distribuição t com 29 graus de liberdade (função rt)
 - e. Plote o histograma superposto da distribuição obtida e da distribuição teórica



Capítulo 2 : Sobre a natureza das relações

Prelúdio: Quem precisa do valor p?

O racional apresentado no capítulo anterior é diretamente relacionado ao método hipotético-dedutivo e seus princípios filosóficos.

Apesar dos inúmeros avanços citados, a interpretação do valor p não é muito intuitiva.

Envolve mensurar quão improváveis são as observações em um cenário hipotético na vigência da hipótese nula.

Sua tradução (errada) mais popular é de que representa “*a chance de o resultado deste estudo estar errado*”.

O arcabouço descrito no capítulo anterior é suficiente para produzir um trabalho científico críptico para leigos.

Ao seguir receitas pré-definidas (formulação de H_0 e H_1 , cálculo de estatísticas e valores p), um texto parece estar em conformação com os padrões acadêmicos, mesmo que a hipótese elementar em torno do objeto de pesquisa seja simplória. Assim, inadvertidamente, priorizamos a forma e relegamos a segundo plano o miolo de propostas científicas.

Trabalhos de pouca originalidade recebem grande atenção pelo rigor por atributos quantitativos (e.g. tamanho amostral grande, valor p baixo), enquanto criativos e revolucionários experimentos menores levam anos ou décadas até atingirem a comunidade.

Outro efeito colateral é a busca por valores p que rejeitem H_0 , desprezando precedentes teóricos e premissas probabilísticas (múltiplos testes).

A difícil interpretabilidade do valor p e as armadilhas frequentes envolvidas no processo de inferência levaram a comunidade científica a questionar a hegemonia desse parâmetro. Há uma presente tendência a abandonar o valor p e o limite $p < 0.05$ como critérios canônicos.

No próximo capítulo, vamos conhecer argumentos contundentes ao método hipotético dedutivo.

Por enquanto, basta sabermos que é sempre vantajoso obter outras informações, complementares ou alternativas.

Neste capítulos, vamos aprender a estimar (1) a magnitude da diferença entre duas amostras e (2) quão relacionados são valores pareados (e.g. peso e altura).

Tamanho de efeito

O tamanho de efeito nos ajuda a expressar magnitudes.

Retomando o exemplo anterior, de que adianta uma diferença significativa entre o tamanho dos bicos dos pássaros, se ela for de 0.00001 mm?

Ainda, existem casos em que estudos pequenos sugerem efeitos importantes, porém o tamanho amostral não fornece poder estatístico suficiente para rejeição da hipótese nula.

Além de saber quão improvável é a diferença observada, é natural imaginarmos o quão grande ela é.

Uma medida bastante popular é o *D de Cohen (Cohen's D)*.

É um parâmetro que expressa a magnitude da diferença sem usar unidades de medida.

Uma torcedora de futebol conta (feliz) a um amigo que seu time favorito venceu com placar de 4 x 1 (gols). Porém, esse amigo acompanha basquetebol e está acostumado a placares como 102 x 93 (cestas).

Como é possível comparar gols com cestas? Qual vitória representa pontuações mais discrepantes: 4 x 1 ou 102 x 93?

O problema aqui é que as pontuações se comportam de maneiras diferentes entre os esportes. O D de Cohen consiste em expressar essa diferença em desvios-padrão. Bastante simples:

$$D_{cohen} = \frac{\mu_1 - \mu_2}{\sigma_{pooled}}$$

Usando a biblioteca *effects*, podemos calcular diretamente:

```
library(effects)
# O dataset galapagos_birds foi criado no capítulo 1
>cohen.d(galapagos_birds$X1,galapagos_birds$X2)

Cohen's d

d estimate: -5.460017 (large)
95 percent confidence interval:
      lower      upper 
-5.954047 -4.965987
```

Cohen propôs algumas faixas para classificar a magnitude desses efeitos:

	Pequeno	Médio	Grande
Cohen's D	0-0.2	0.2-0.5	0.5 - 0.8

Assim, podemos atualizar nossos resultados anteriores, reportando também o tamanho de efeito da diferença e seu intervalo de confiança.

Correlações

Na empreitada científica, não nos atemos apenas a comparações. Um objetivo mais nobre é descrever exatamente como se dá a relação entre entidades estudadas.

Como sabemos, existem muitas classes de funções para expressar relações entre variáveis/conjuntos. Nos capítulos anteriores, usamos algumas funções, como $y = \sqrt{x}$ e $y = e^x$.

Diversas leis naturais tornaram-se particularmente conhecidas, como a relação entre força, massa e aceleração, elucidada por Newton:

$$\vec{F} = m\vec{a}$$

E a relação entre massa e energia para um objeto em repouso, descoberta por Einstein:

$$E = mc^2; c^2 \sim 8.988 * 10^{16} \frac{m^2}{s^2}$$

As equações acima descreve uma relação linear entre grandezas.

Relações lineares

Uma relação linear entre duas variáveis indica que elas estão correlacionadas em uma proporção constante para qualquer intervalo.

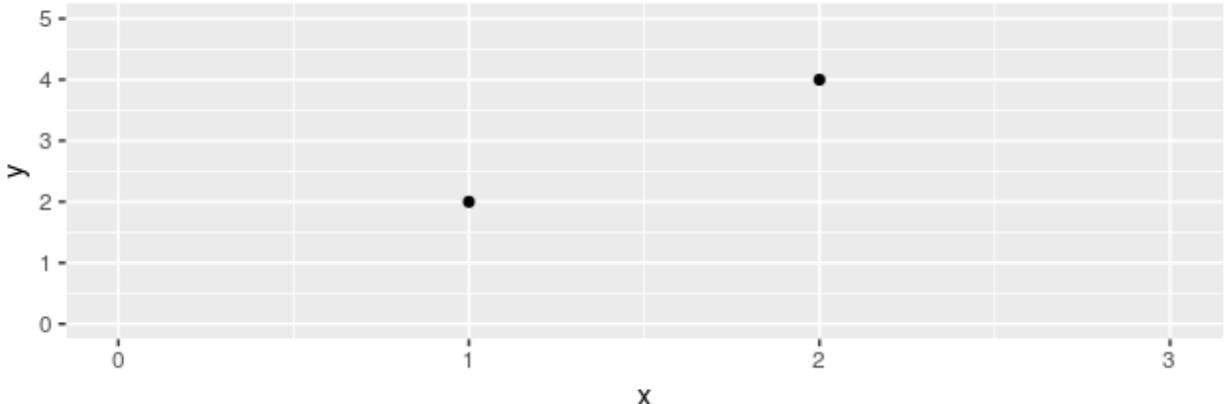
Isto é, valores maiores de massa correspondem a um aumento proporcional em energia. O valor de c^2 expressa essa proporção constante.

Exemplo: uma molécula de água pesa aproximadamente $m_{H_2O} = 2.992 \cdot 10^{-23} g$. Portanto, a energia associada é $E_{H_2O} = 2.992 \cdot 10^{-23} \cdot 8.988 \cdot 10^{16} \sim 2.689 \cdot 10^{-6} J$. Se triplicarmos o número de moléculas de água, o mesmo acontecerá com a energia associada: $E_{3H_2O} = 3 \cdot E_{H_2O}$.

Se a correlação é positiva, incrementos em x serão proporcionais a incrementos em y . Se a correlação é negativa, incrementos em x serão proporcionais a decréscimos em y .

Num cenário perfeito, se sabemos que há uma relação linear entre variáveis, precisamos de apenas duas observações para descobrir proporção entre elas. Esse problema é idêntico ao de encontrar a inclinação da reta que passa por dois pontos. É de fácil resolução usando técnicas elementares.

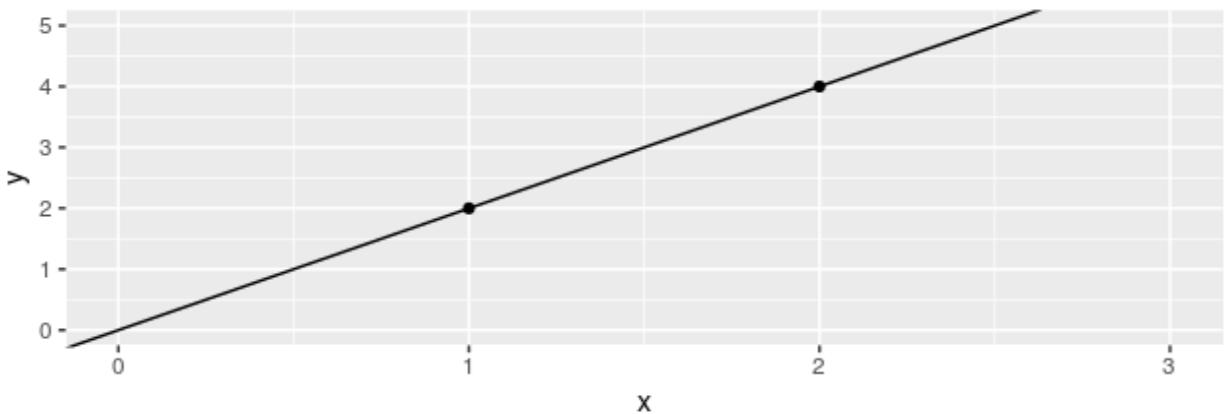
```
>library(ggplot2)
>ggplot() +
  geom_point(mapping=aes(x=1,y=2)) +
  geom_point(mapping=aes(x=2,y=4)) +
  xlim(0,3) + ylim(0,5)
```



$$y = \beta * x$$

$$a = (1, 2); b = (2, 4) \rightarrow \beta = 2$$

```
>ggplot()+
  geom_point(mapping=aes(x=1,y=2))+
  geom_point(mapping=aes(x=2,y=4))+
  xlim(0,3)+ylim(0,5)+
  geom_abline(slope = 2)
```



Erros e aleatoriedade

Controlando fatores experimentais, as relações descritas são bastante precisas. Em um cenário sem atrito com superfícies e com o ar, os erros de medida obtidos com $\vec{F} = m\vec{a}$ são muito baixos.

Entretanto, nem sempre isso é verdadeiro.

Primeiro, podemos sofrer interferência de variáveis desconhecidas.

Imaginemos um conjunto de medidas antropométricas, com altura e peso e indivíduos.

É esperado que a altura de um ser humano esteja relacionada com seu peso. Entretanto, outras características não medidas, como percentual de gordura total, podem interferir nos valores finais. Normalmente, tratamos essas flutuações como erros aleatórios¹¹.

Podemos simular este cenário partindo de variáveis idênticas e adicionando ruído aleatório.

¹¹A natureza da aleatoriedade é uma questão filosófica. Em última instância, podemos imaginar que seria possível explicar flutuações randômicas através de variáveis desconhecidas (*hidden variables*). Isso é verdade para a maioria dos fenômenos naturais. Entretanto, descobertas experimentais recentes em física quântica (*Bell's inequality experiment*) sugerem que variáveis ocultas não podem explicar a natureza probabilística das observações.

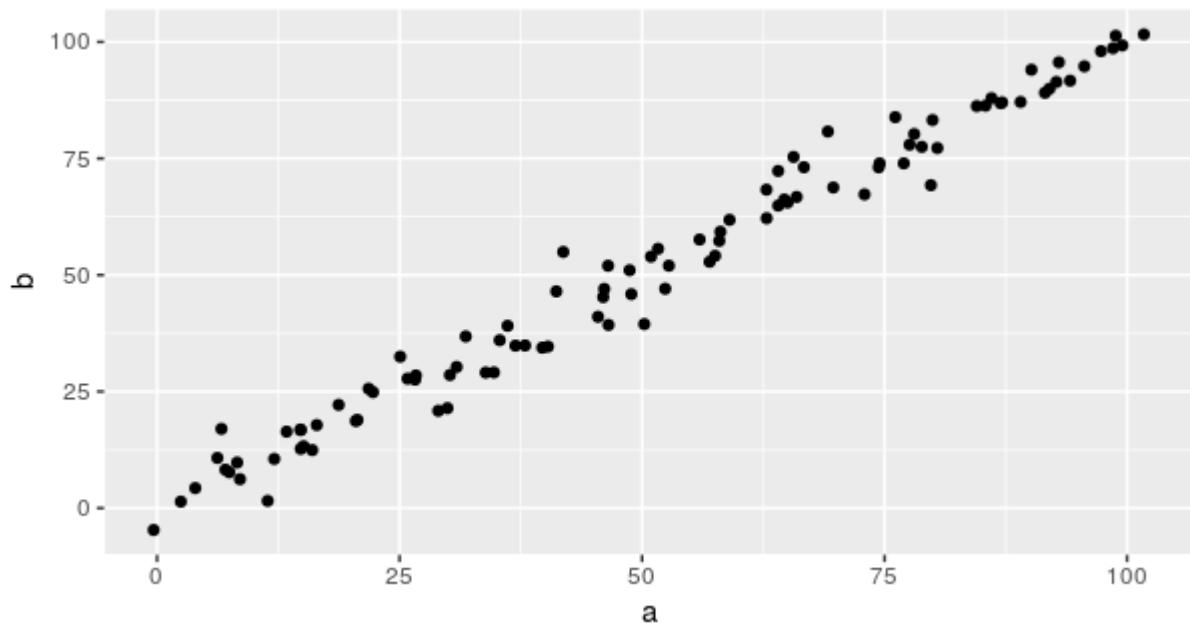
```

>set.seed(2600)
>a <- seq(1:100)+rnorm(n=100, sd=3)
>b <- seq(1:100)+rnorm(n=100, sd=3)

>cor_data <- data.frame(a,b)

>ggplot(cor_data,aes(x=a,y=b))+geom_point()

```



O resultado sugere que há uma forte relação linear entre x e y . Por outro lado, notamos que é impossível para uma reta cruzar todos os pontos. A seguir, vamos investigar como quantificar a correlação linear, assim como encontrar a reta que minimiza a distância para todas as observações.

Com essas ferramentas, podemos estender nossas inferências. Além de comparações, teremos noções sobre a magnitude de uma relação, assim como poderemos prever o valor esperado para novas observações.

O coeficiente de correlação produto-momento de Pearson, ou, simplesmente, ρ de Pearson.

O coeficiente de correlação (ρ) de Pearson é um número real garantidamente¹² entre -1 e 1. Expressa a magnitude e o sentido de uma relação linear, sendo -1 uma relação inversa perfeita e 1 uma relação direta perfeita.

Para os dados que geramos, a correlação é quase perfeita: $\rho = 0.989$.

O coeficiente possui *produto-momento* em seu nome, pois usa uma abstração originalmente empregada na física: o momento.

¹²Inequação de Cauchy-Schwarz

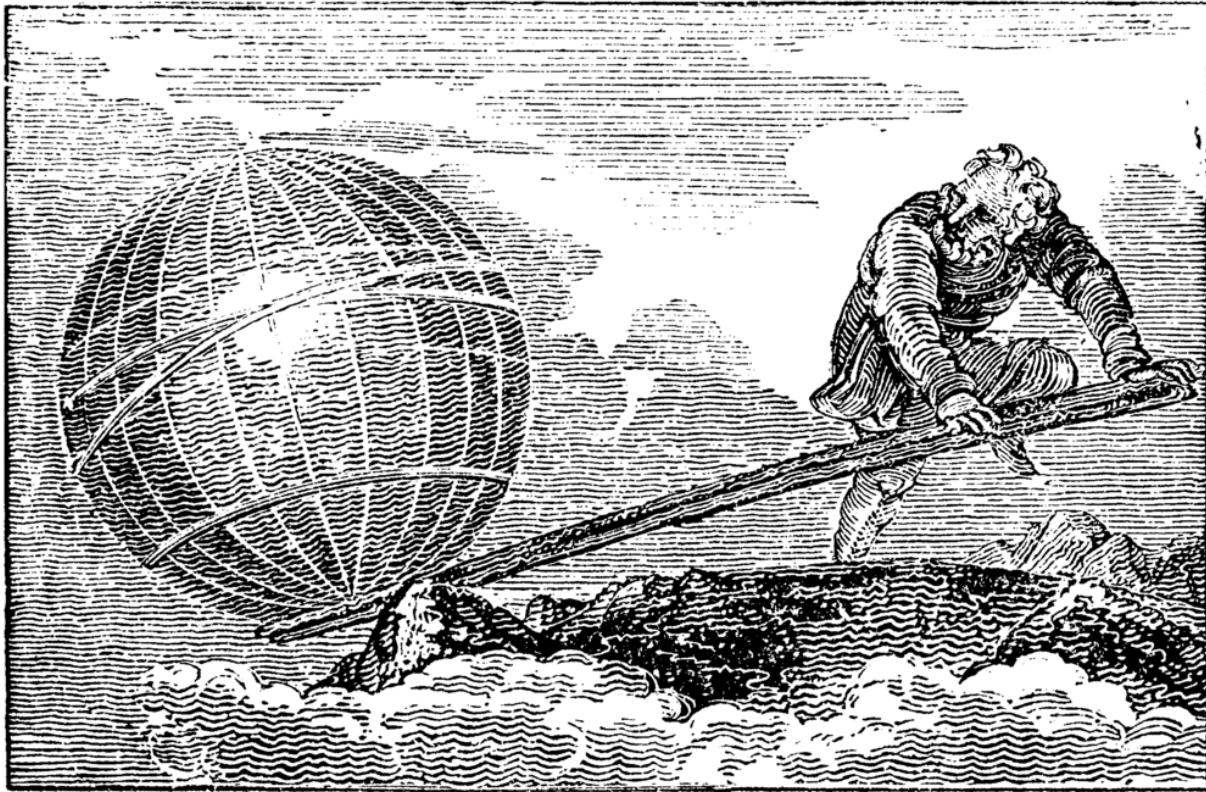


Figure 9: Dê-me um ponto de apoio e eu moverei a Terra

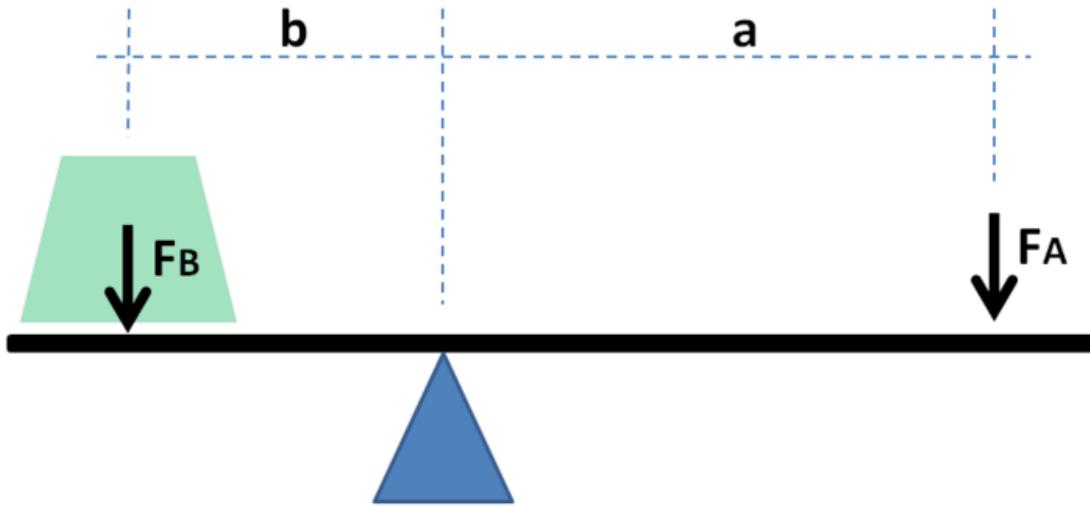
Um breve mergulho na física: Momentos

13

Para adquirir uma intuição sobre o coeficiente, é interessante resgatar o conceito físico de momento, originalmente concebido por Arquimedes. Embora não tenha inventado a alavanca, ele descreveu os princípios matemáticos por trás dela.

Em *Sobre o equilíbrio dos planos*, Arquimedes declara que *Magnitudes ficam em equilíbrio quando em distância reciprocamente proporcional aos seus pesos*.

¹³Pappus de Alexandria, Synagogue, Livro VIII



Essa é a conhecida Lei da alavanca. Dado um ponto de apoio e um plano sobre ele, aplicamos uma força em qualquer local do plano. O momento (torque) resultante é o resultado da multiplicação da grandeza física (F) pela distância até o ponto fixo (d).

$$M = F * d$$

Supondo uma força constante, quanto mais nos afastamos do ponto fixo, maior o momento resultante. Posteriormente, os físicos estenderam o conceito para outros domínios. Por exemplo, um objeto com cargas opostas $-q$ e $+q$ separados por uma distância d possui momento (momento dipolar elétrico) análogo: $M = q*d$. De uma maneira geral, falamos em momento ao multiplicarmos uma grandeza física por uma distância.

Momento resultante

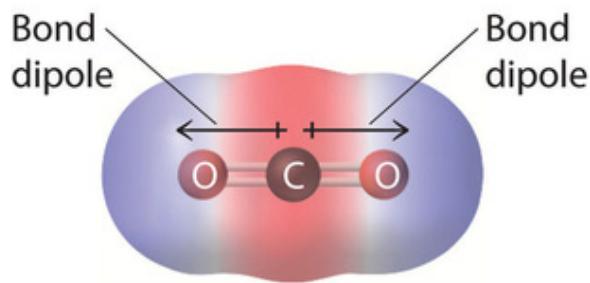
No caso da alavanca, vimos que cada força aplicada sobre o objeto está associada a um momento(torque). Sabemos que a gravidade atua sobre cada pedaço com massa compondo o todo. Podemos então calcular o momento resultante somando os momentos de todos os N pontos. Seja F_i a função descrevendo a força em cada i-ésimo:

$$M = \sum_{i=1}^N F_i d_i$$

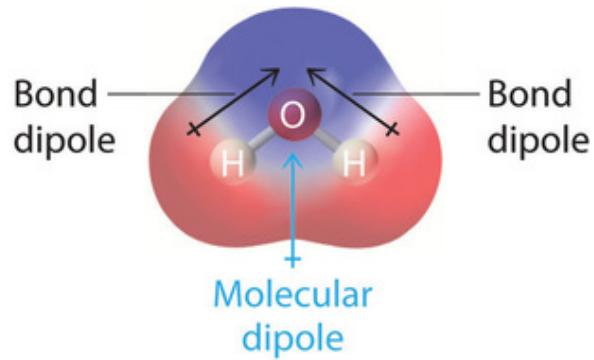
Um sistema, como o pássaro apoiado sobre o dedo, está em equilíbrio quando a soma dos momentos em relação ao ponto fixo é zero. Para cargas elétricas, o sistema é apolar quando o momento é zero. Na figura abaixo, vemos como a molécula de CO_2 é apolar, enquanto a molécula de água é polar:



Figure 10: Como o brinquedo acima fica equilibrado sobre apenas um ponto?



(a) No net dipole moment



(b) Net dipole moment

Os momentos descritos acima são expressões do *primeiro momento*, uma vez que a grandeza é multiplicada pela distância com expoente 1: $d = d^1$.

Podemos calcular outros momentos, exponenciando o componente espacial (distância). Vamos estudar agora momentos de massa de um objeto unidimensional:

O **momento zero** de massa para um objeto é $M_0 = \sum_{i=1}^N m_i d_i^0$. Como $d^0 = 1$, temos $M_0 = \sum_{i=1}^N m_i$, que é simplesmente a soma das massas de todos os pontos. O momento zero é a **massa total**.

$$M_0 = m$$

O **primeiro momento** de massa para um objeto é $M_1 = \sum_{i=1}^N m_i d_i^1$ e determina o **centro de massa** em relação à dimensão d . É o ponto em que está o dedo em que se equilibra o pássaro da foto.

$$M_1 = C_m$$

O **segundo momento** de massa é $M_2 = \sum_{i=1}^N m_i d_i^2$ e é o **momento de inércia**. Corresponde à resistência do sistema a rotações. Perceba que os termos d_i^2 estariam presentes nas área de um círculo com centro idêntico ao do objeto e raio igual à distância para o centro: πd^2 . A resistência total a rotação é análoga à resistência oferecida pelos raios destes círculos imaginários¹⁴.

¹⁴<https://physics.stackexchange.com/a/371165/218274>



O n-ésimo momento é dado por

$$M_n = \sum_{i=1}^N m_i d_i^n$$

Generalizando momentos

Podemos generalizar ainda mais a abstração e calcular momentos de entidades abstratas, como variáveis aleatórias. **Melhor: já fizemos isso anteriormente!**

Seja $f(x)$ a função que descreve uma distribuição de probabilidades para a variável,

Assim como o **momento zero** representa a soma da contribuição de cada ponto para a massa (massa total), aqui ele representa a soma das probabilidades possíveis, a probabilidade total (1).

O **primeiro momento** corresponde ao centro de massa na mecânica estática. Para probabilidades, é o centro, a **média**.

O **segundo momento** corresponde ao momento inercial e é a **variância**.

Os momentos **terceiro** e **quarto** normalizados informam sobre assimetrias (*skewness*) e peso de valores extremos (*kurtosis*).

Formalmente, seja $d(x, x_0)$ o valor da distância ao centro x_0 de referência ($x - x_0$), o n-ésimo momento μ_n é definido por:

$$\mu_n = \int_{-\infty}^{\infty} d(x, x_0)^n f(x) dx$$

A integral acima corresponde à versão contínua da soma de partes discretas apresentadas antes para uma grandeza física, como a massa: $M_n = \sum_{i=1}^N d_i^n m_i$

Momento zero:

$$\mu_0 = \int_{-\infty}^{\infty} d(x, x_0)^0 f(x) dx$$

A soma de todas probabilidades de uma distribuição deve somar 1.

$$= \int_{-\infty}^{\infty} f(x) dx = 1$$

Primeiro momento: $\mu_1 = \int_{-\infty}^{\infty} d(x, x_0) f(x) dx$, supondo centro em 0 ($x_0 = 0$), temos a média,

$$\mu_1 = \int_{-\infty}^{\infty} x f(x) dx$$

, também chamado valor esperado $E[X]$. Estende a intuição de somar as medidas e dividir pelo número de observações ao passo em que usamos uma integral para somar as infinitesimais possibilidades para $f(x)$.

Segundo momento:

$$\mu_2 = \int_{-\infty}^{\infty} d(x, x_0)^2 f(x) dx$$

. Como vimos no capítulo introdutório, a soma dos quadrados dos desvios, nossa variância,

$$\sigma^2 = E[(x - \mu)^2]$$

Notas finais sobre o Teorema do Limite Central

Podemos entender melhor o teorema do limite central. As informações fornecidas pelos momentos são valiosas: uma função de probabilidade é totalmente definida por seus momentos.

O Teorema do Limite Central, de que falamos antes, é provado mostrando equivalência entre momentos da curva normal e da soma de n distribuições idênticas através de outras ferramentas.

Podemos criar uma função geradora de momentos, $M_X(t) = E[e^{tX}]$. Chamamos ela assim, pois sua forma polinomial via expansão de Taylor corresponde à soma $1 + tX + \frac{t^2 X^2}{2!} + \frac{t^3 X^3}{3!} + \dots$

O valor esperado dessa soma:

$$\begin{aligned} E[M_X(t)] &= 1 + tE[X] + \frac{t^2 E[X^2]}{2!} + \frac{t^3 E[X^3]}{3!} + \dots \\ &= 1 + tM_1 + \frac{t^2 M_2}{2!} + \frac{t^3 M_3}{3!} + \dots \end{aligned}$$

Em que M_n corresponde ao n -ésimo momento.

A função característica é a transformada de Fourier da função de probabilidade, associando componentes periódicos no plano imaginário. É o mesmo que multiplicar o argumento t da função geradora de momentos $M_X(t) = E[e^{tX}]$ pela unidade imaginária, $\phi_X(t) = M_X(it) = E[e^{itX}]$. Funções com cumulantes idênticos possuem momentos idênticos e podemos demonstrar que a função característica que a soma de n distribuições iguais possui momentos idênticos aos da normal.¹⁵

Com os conceitos adquiridos em mãos, é fácil entender o ρ de Pearson.

¹⁵Two Proofs of the Central Limit Theorem, Yuval Filmus, 2010. <http://www.cs.toronto.edu/~yuvalf/CLT.pdf>

Calculando correlações lineares

A noção de **distância** ou **desvio** se repetiu muitas vezes.

De fato, o coeficiente de correlação linear nasceu quando Francis Galton (1888) estudava numericamente dois problemas aparentemente distintos em antropometria¹⁶:

1. **Antropologia:** Se recuperássemos de um túmulo antigo apenas um osso da coxa (fêmur) de um indivíduo, o que poderíamos dizer sobre sua altura?
2. **Ciência forense:** Com o intuito de identificar criminosos, o que pode ser dito sobre medidas diferentes de uma mesma pessoa?

Galton percebeu que, na verdade, estava lidando com o mesmo problema. Dadas medidas pareadas, (x_i, x'_i) , o que o desvio de x_i informa sobre o desvio de x'_i ?

O fêmur recuperado do esqueleto de um faraó é 5 cm maior que a média. Quão distante da média esperamos que seja sua altura? Ingenuamente, podemos pensar que se uma das medidas é 1% maior que a média, a outra também será 1% maior. Galton percebeu que havia um armadilha nesse pensamento.

Apesar de haver uma relação entre as medidas, há também flutuações aleatórias: parte do desvio é resultante disso. Precisamos entender o grau de correlação pra fazer um bom palpite.

Então, propôs um coeficiente mensurando a relação entre desvios de variáveis. Se tamanho do fêmur e altura estão muito relacionadas, um fêmur grande sugere indivíduo igualmente alto. Caso contrário (baixa correlação), um fêmur grande (desvio alto) não implica grande estatura.

Para quantificar a relação, multiplicamos os desvios de cada par de medidas:

$$Cov(X, X') = \sum_{i=1}^N (x_i - \mu_x)(x'_i - \mu_{x'})$$

A expressão acima expressa a **covariância** entre X e X' e será útil em outros contextos. A expressão lembra o cálculo do primeiro momento, porém cada desvio é multiplicado pelo desvio correspondente da medida pareada. Daí o nome coeficiente de correlação *produto-momento*.

Note que, se ambos os desvios concordam em sentido (sinal), o resultado da multiplicação será positivo. Pares consistentemente concordantes aumentam o valor da soma final. Se ambos os desvios discordam em sentido (sinal), o resultado será negativo. Pares consistentemente discordantes diminuem o valor da soma final.

Assim, podemos ter variáveis altamente correlacionadas positiva ou negativamente, desde que o sentido da associação seja constante. Em contrapartida, se as medidas são ora discordantes e ora concordantes, os valores tendem a se anular na soma e o resultado se aproxima de zero.

Observar apenas a covariância é perigoso, pois os valores dependem da unidade de medida e da dispersão dos dados.

Calculamos o coeficiente de correlação de Pearson, normalizando¹⁷ a covariância ao dividí-la pelo produto dos desvios-padrão:

$$\rho_{XX'} = \frac{cov(X, X')}{\sigma_X \sigma_{X'}}$$

De forma extensa:

$$\rho_{XX'} = -\frac{\sum_{i=1}^N (x_i - \mu_x)(x'_i - \mu_{x'})}{\sqrt{\sum_i^N (x_i - \mu_x)^2} \sqrt{\sum_i^N (x'_i - \mu_{x'})^2}}$$

Uma boa notícia: ρ segue uma distribuição conhecida, a distribuição t, com $n-2$ graus de liberdade. Podemos usar as ferramentas anteriores para testar hipóteses.

¹⁶Francis Galton's account of the invention of correlation. Stephen M. Stigler. Statistical Science. 1989, Vol. 4, No. 2, 73-86.

¹⁷Aqui, normalização tem o sentido de ajustar a escala das medidas. Não confundir com transformações para que os dados passem a ter distribuição gaussiana.

Exemplo prático

O exemplo a seguir foi um feliz achado. Na época, o governo brasileiro discutia a necessidade da ampliar número de médicos para melhorar a assistência à saúde. Alguns defendiam ser uma decisão acertada, enquanto outros advogavam que os investimentos deveriam ser feitos em outras áreas da saúde.

Por curiosidade, acessei os dados da WHO (World Health Organization) e do banco mundial (World Bank) sobre quantidade de médicos por país e indicadores de saúde. Minha expectativa era encontrar pelo menos uma tímida relação entre indicadores. Mais do que isso, entender qual a localização do Brasil em relação a outros países. Fui surpreendido por uma forte correlação, que exploraremos a seguir.

Adotamos países como unidade observacional com medidas x , o número de médicos 1,000 habitantes, e y , a expectativa de vida saudável ao nascer.

Usando dados obtidos dos portais da WHO e do World Bank, plotamos os pontos no plano cartesiano.

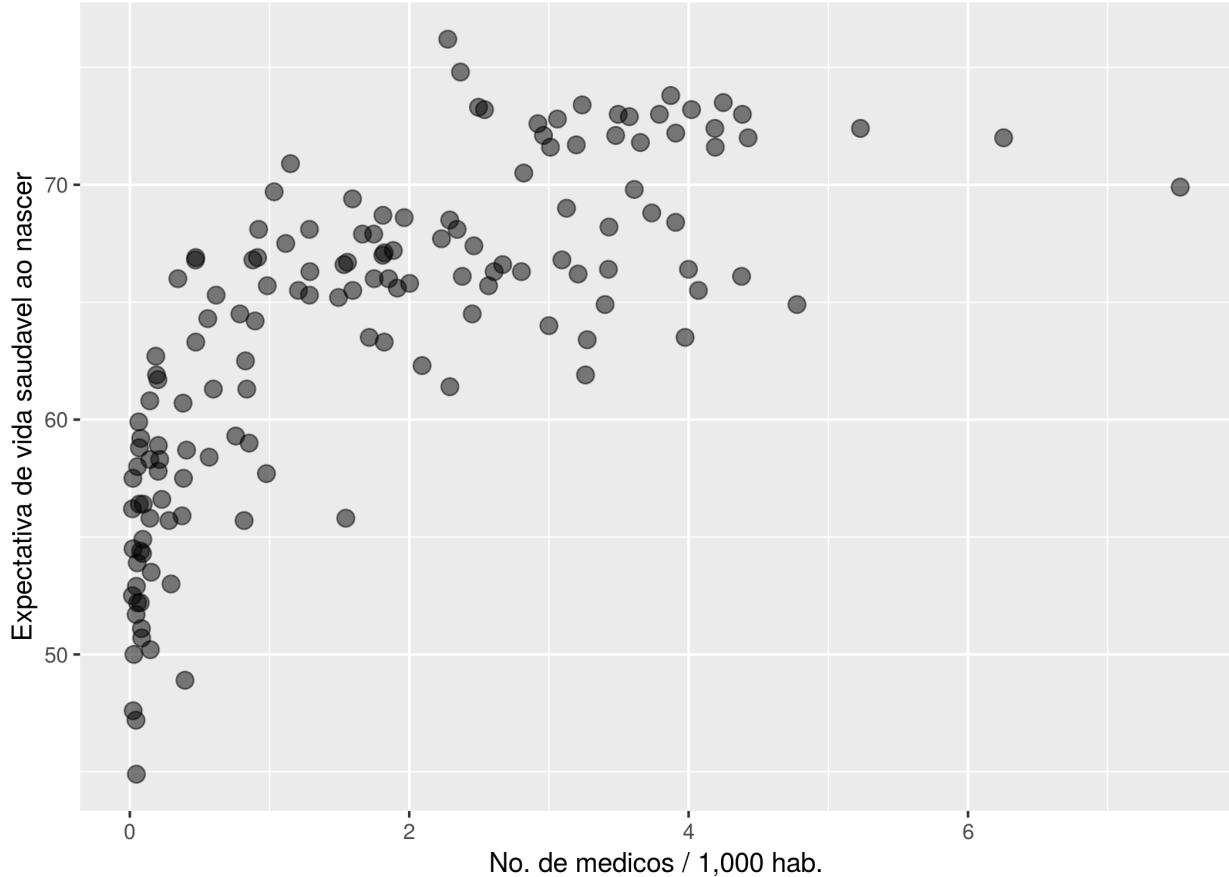
```
# http://apps.who.int/gho/data/view.main.HALEXv
# https://data.worldbank.org/indicator/SH.MED.PHYS.ZS
>library(magrittr)
>library(ggplot2)
>library(dplyr)

>worldbank_df <- read.csv("data/API_SH.MED.PHYS.ZS_DS2_en_csv_v2_10227587.csv",
  header = T, skip = 3)
>colnames(worldbank_df)[1] <- "Country"

>worldbank_df$n_docs <- sapply(split(worldbank_df[,53:62], #lists of values
  seq(nrow(worldbank_df))), 
  function(x) tail(x[!is.na(x)],1)) %>% #ultimos valores não nulos
  as.numeric

>who_df <- read.csv("data/who_lifeexpect.csv",skip=2)
>who_df$hale <- who_df$X2016
>uni_df <- left_join(worldbank_df[,c("Country","n_docs")],
  who_df[,c("Country","hale")],by="Country")

>ggplot(uni_df,aes(x=n_docs,y=hale))+
  geom_point(alpha=0.5,size=3) +
  xlab("No. de médicos / 1,000 hab.")+
  ylab("Expectativa de vida saudável ao nascer")
```



É evidente que o padrão não é aleatório. Visualmente, notamos que o valor da expectativa de vida aumenta com maior N° de médicos. Ainda, notamos um aumento inicialmente rápido até atingir um platô. O padrão é semelhante ao de uma curva logarítmica.

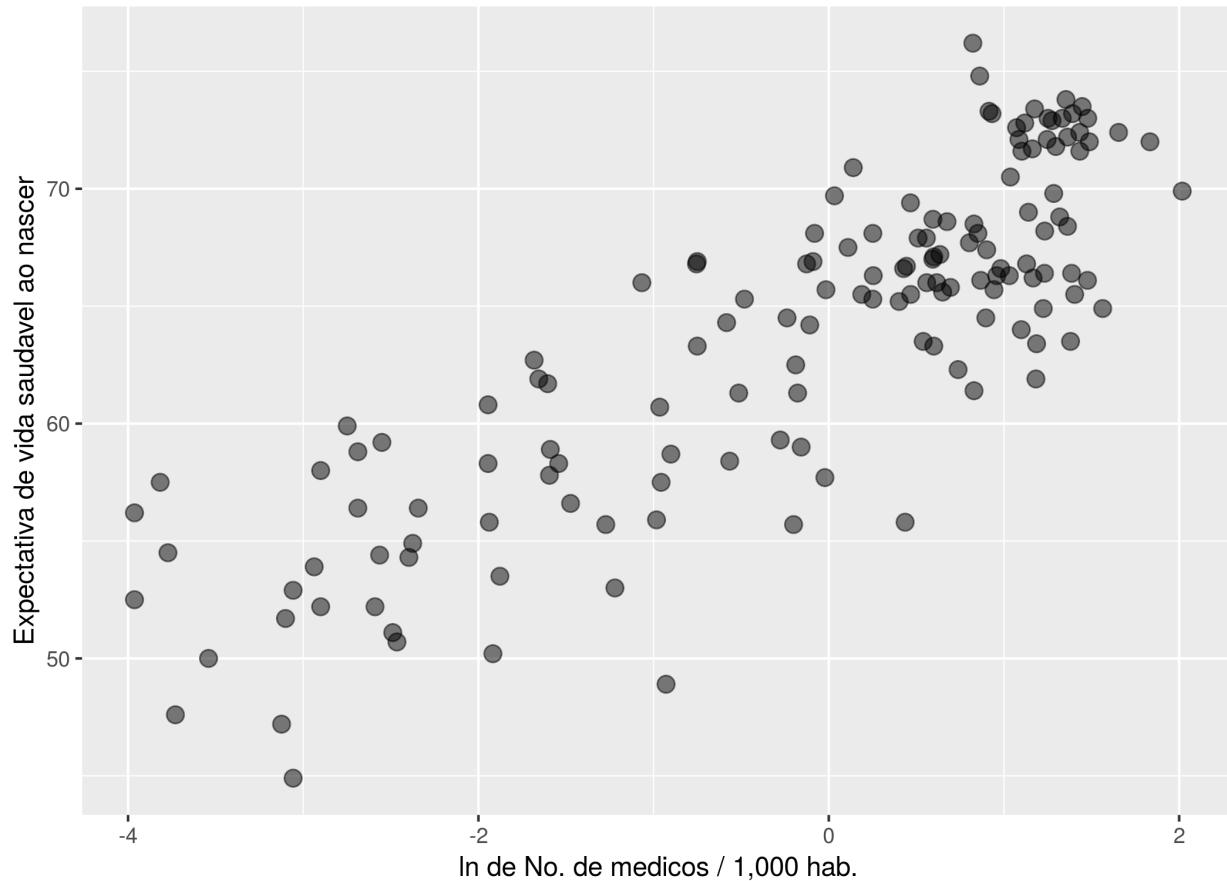
$$y = \log(x) \text{ ou } HALE = \log(N_{\text{médicos}})$$

Se essa hipótese for verdade, transformar o número de médicos usando função logarítmica tornará a relação linear com a variável transformada:

Se $y = \log(x)$, fazemos a substituição $x' = \log(x)$ para obtermos $y = x'$.

Então a expectativa de vida se torna linearmente correlacionada ao logaritmo do número de médicos.

```
>uni_df$log_docs <- log(uni_df$n_docs)
>ggplot(uni_df,aes(x=log_docs,y=hale))+
  geom_point(alpha=0.5,size=3) +
  xlab("ln de No. de medicos / 1,000 hab.")+
  ylab("Expectativa de vida saudavel ao nascer")
```



De fato, verificamos uma notável tendência linear para os pontos.

Usando a implementação nativa em R para o coeficiente de Pearson:

```
>cor.test(uni_df$log_docs, uni_df$hale)
Pearson's product-moment correlation
data: uni_df$log_docs and uni_df$hale
t = 18.572, df = 143, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval: 0.7854248 0.8828027
sample estimates:
      cor
0.8407869
```

A correlação linear obtida para nossa amostra de países é surpreendentemente grande, como sugeria a visualização ($\rho \sim 0.841$).

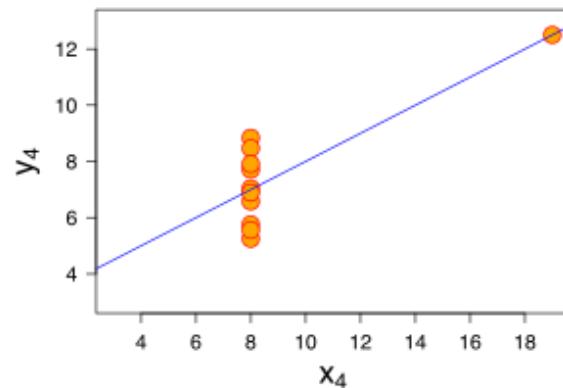
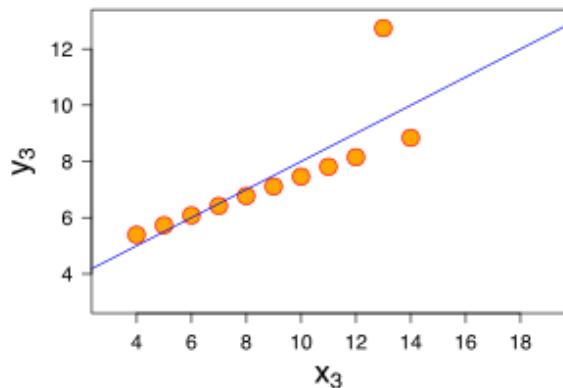
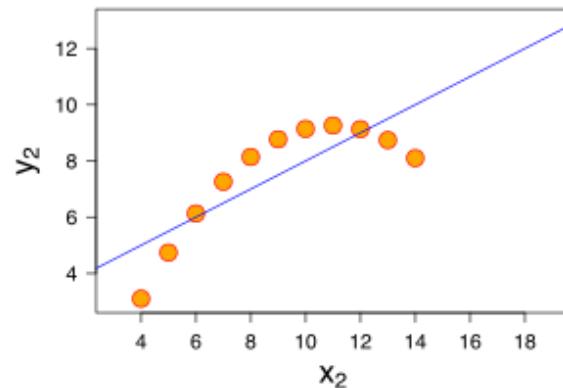
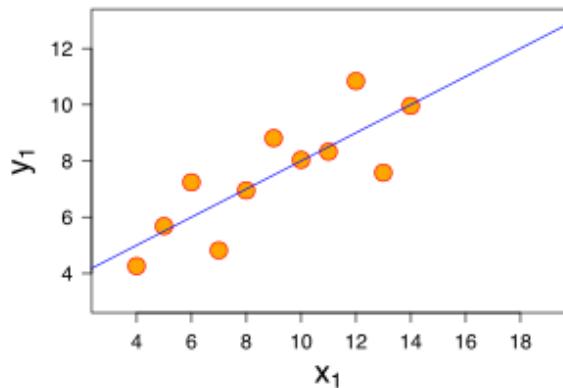
O valor p é baixo ($p < 0.001$) considerando a hipótese nula H_0 de $\rho = 0$. Concluímos então que há uma relação linear significativa de forte magnitude entre o logaritmo do número de médicos e a expectativa de vida dos países em nossa amostra.

É realmente curioso que exista uma relação matemática tão evidente entre construtos tenuamente conectados. O tempo médio que um organismo leva entre nascimento e morte e o número de profissionais atuantes. É virtualmente impossível explicitar cada relação causal por trás dessa relação, que se manifesta de forma robusta através da soma de muitos fatores relacionados.

Nota

É costume afirmar que não existe relação entre variáveis caso o coeficiente de correlação não se mostre importante. Como vimos, esse indicador informa apenas sobre relações lineares entre variáveis. A visualização dos dados pode ser de grande ajuda na inferência sobre a natureza de relações.

Dados com distribuições bastante diferentes podem resultar em coeficientes iguais, como mostra o clássico quarteto de Anscombe. As 4 amostras abaixo apresentam o mesmo coeficiente de correlação.



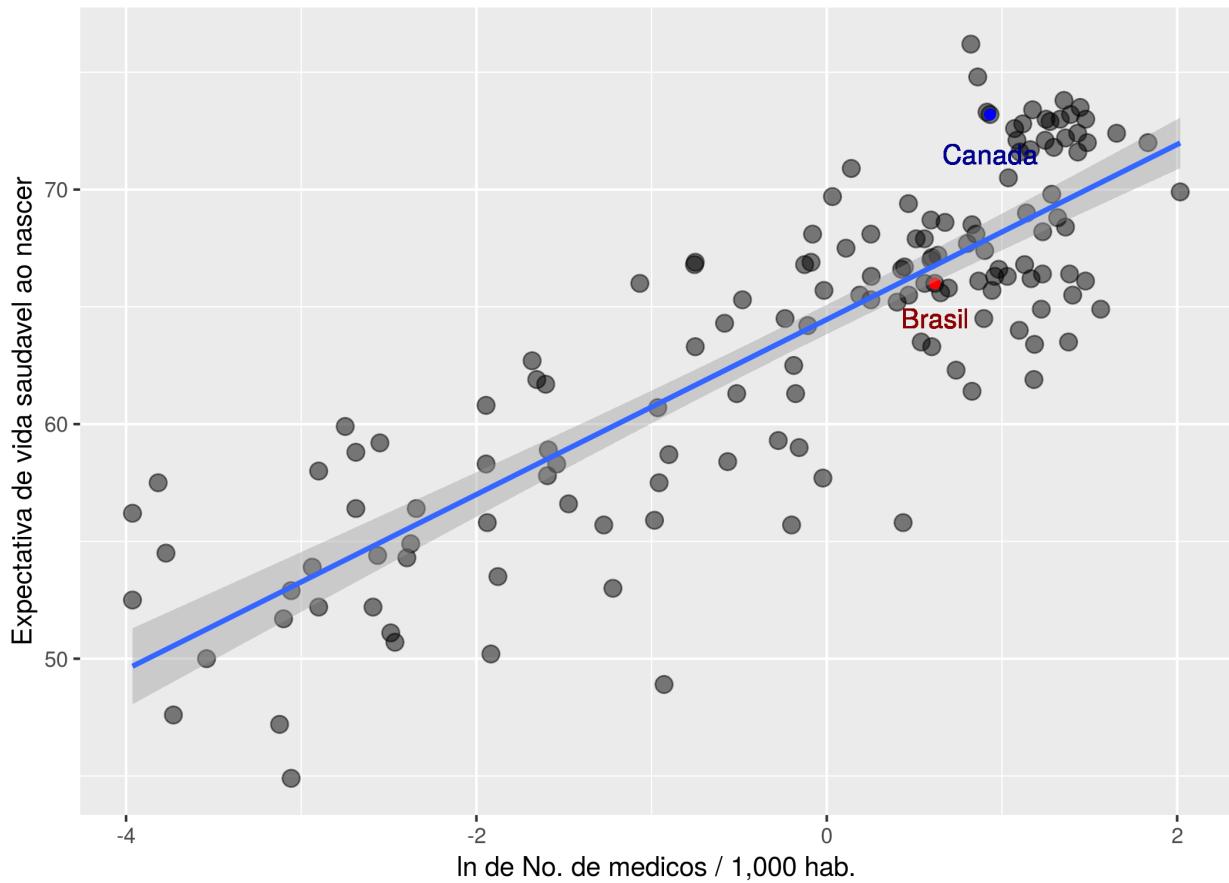
Previsões

Agora, sabemos que é razoável assumir uma relação linear entre essas variáveis. Como dito antes, podemos então encontrar a reta que minimiza a distância para as observações.

A equação que descreve essa reta nos informa o valor esperado para expectativa de vida dado o número de médicos.

```
>uni_df$log_docs <- log(uni_df$n_docs)
>ggplot(uni_df,aes(x=log_docs,y=hale))+
  geom_point(alpha=0.5,size=3) +
  geom_point(y=66.0,x=0.61626614,color="red")+
  geom_text(y=64.5,x=0.61626614,label="Brasil",color="dark red")+
  geom_point(y=73.2,x=0.93177030,color="blue")+
  geom_text(y=71.5,x=0.93177030,label="Canada",color="dark blue")+
  geom_smooth(method="lm")+
```

```
xlab("ln de No. de medicos / 1,000 hab.")+
ylab("Expectativa de vida saudavel ao nascer")
```



Assumindo que realmente há uma relação linear, vemos que o Brasil está bastante próximo do esperado para o número de médicos¹⁸. O Canadá possui uma expectativa de vida alta para o número de profissionais. Como viemos discutindo ao longo do texto, questões filosóficas e metodológicas devem ser enderaçadas antes de tomar conclusões.

Entretanto, temos uma boa ilustração de como ciência de dados pode nos ajudar a tomar decisões em contextos reais.

¹⁸É praticamente consenso entre especialistas que o Brasil possui problema de distribuição de profissionais, com déficit de médicos em áreas mais pobres e pouco populosas.

Predições com modelos lineares

Como adivinhar uma medida com base na outra? Considerando a relação linear descoberta anteriormente, podemos criar uma função que receba como input o valor de uma variável (número de médicos) e retorne como output o valor esperado para a expectativa de vida.

Descobrir a equação que descreve esta função consiste em encontrar a reta que melhor se ajusta à nuvem de pontos, como na figura anterior.

Para isso, calculamos a inclinação (β_1) e o ajuste vertical (β_0) que minimizam a soma das distâncias entre a reta e as observações. O termo ϵ corresponde aos erros, com distribuição normal de média 0 e desvio padrão σ .

$$y_i = \beta_0 + \beta_1 x_i + \epsilon$$

Ajustamos o modelo usando a função lm(linear model) do R:

```
# log_docs : x' = log(x)
> lm(hale ~ log_docs, data=uni_df)

Call:
lm(formula = hale ~ log_docs, data = uni_df)

Coefficients:
(Intercept)      log_docs
       64.46          3.73
```

Temos $\beta_0 \sim 64.46$ e $\beta_1 \sim 3.73$.

Nossa estimativa para a expectativa de vida saudável “começa” em 64.46 anos e aumenta com o número de médicos no país. Especificamente, aumenta em 3.73 para cada unidade de nossa variável transformada ($\log(x)$).

Em nosso dataset, o Brasil possui 1.852 médicos/1,000 hab. Nossa predição então é:

$$\hat{y}_{Brasil} = \log(1.852 * 3.73 + 64.46 \sim 66.8, \text{ o que está bastante próximo do número real}(66).$$

Estimadores

Existe mais de uma maneira de estimar esses parâmetros.

Uma de particular interesse, que também servirá em outros contextos, é a de Maximum likelihood (máxima verossimilhança).

Primeiro, determinamos uma função que descreve a probabilidade da observação na variável alvo (y_i) ocorrer dadas medidas das variáveis preditoras (x_i) e um conjunto de parâmetros (β_k).

Podemos adotar como função de verossimilhança (*likelihood function*) para os valores y_i uma distribuição de probabilidades gaussiana cuja média é dada pela reta $\mu_{y_i} = \beta_0 + \beta_1 * x_i$. Assim, a probabilidade de cada valor y_i é dada por uma gaussiana, de acordo com o desvio para o valor previsto pela reta.

$$L \sim N(\mu_{y_i}, \sigma^2)$$

Assumindo que as observações são independentes, a probabilidade do conjunto de observações é dada pelo produto delas.

$$L = \prod_{i=1}^n P(y_i | x_i; \beta_0, \beta_1, \sigma^2)$$

Substituindo os valores de μ para a gaussiana pelas previsões da reta:

$$f(y_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i - \mu)^2}{2\sigma^2}}$$

$$L(\beta_0, \beta_1, \sigma^2) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{y_i - (\beta_0 + \beta_1 x_i)^2}{2\sigma^2}}$$

Essa é nossa função de verossimilhança e expressa a probabilidade de observarmos as medidas y_i dadas as medidas x_i e considerando um conjunto de parâmetros (β_0, β_1) .

O objetivo então é encontrar parâmetros que maximizem essa função. Por conveniência, aplicamos uma transformação logarítmica nesta função (*log likelihood function*). Isso transforma nosso produtório em um somatório e passamos o contradomínio do intervalo $[0; 1]$ para $[-\infty, 0)$.

$$\begin{aligned} \text{log likelihood}(\beta_0, \beta_1, \sigma^2) &= \log \prod_{i=1}^n P(y_i | x_i; \beta_0, \beta_1, \sigma^2) \\ &= \sum_{i=1}^n \log P(y_i | x_i; \beta_0, \beta_1, \sigma^2) \\ &= -\frac{n}{2} \log 2\pi - n \log \sigma - \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2 \end{aligned}$$

Os parâmetros que maximizam a função de verossimilhança (max. likelihood, ML) são os mesmos que maximizam o logaritmo da função de verossimilhança (log-likelihood).

Introduzimos o racional do estimador ML pois ele será útil futuramente. Em verdade, é fácil entender as fórmulas fechadas para nossos parâmetros, pois apenas expressam as relações lineares exploradas ¹⁹:

$\hat{\beta}_1$ expressa a magnitude da correlação entre X e Y . É natural que seu valor seja a covariância normalizada pela variância do preditor.

$$\hat{\beta}_1 = \frac{\text{cov}(XY)}{\sigma_x^2}$$

$\hat{\beta}_0$ é nosso intercepto, então é a diferença entre médias preditas e previsões considerando o valor médio em X.

$$\hat{\beta}_0 = \mu_y - \hat{\beta}_1 \mu_x$$

Por fim, a variância dos erros σ^2 é dada pelo quadrado dos desvios das previsões em relação às medidas.

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^n (y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i))^2$$

As soluções acima fornecem as melhores estimativas que podemos obter minimizando a distância da reta aos pontos.

Devemos então nos preocupar em saber se o modelo linear encontrado é bom na previsão dados.

¹⁹Detalhes das deduções dos estimadores OLS and Max. Likelihood: <https://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/05/lecture-05.pdf> ; <https://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/06/lecture-06.pdf>

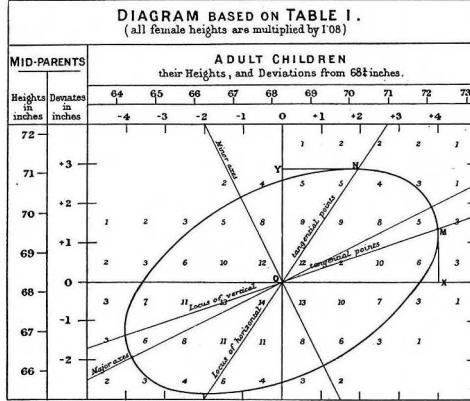


Figure 11: O primeiro gráfico de regressão linear. Ilustração de Francis Galton (1875) relação entre altura de pais e filhos.

Avaliando performance

Existem diferentes parâmetros para avaliar a performance de um modelo. Boa parte da pesquisa em aprendizagem estatística hoje consiste em implementar heurísticas que levem a melhores indicadores de performance.

Para regressão linear, o R^2 (coeficiente de determinação) é um coeficiente bastante usado. Expressa a proporção entre (1) variância explicada pelo modelo e (2) variação total. Chamamos de resíduo(ou erro) a diferença entre valores preditos e valores reais.

(1) Para capturar a magnitude dos erros do modelo, somamos o quadrado de todos os resíduos (*sum of squared residuals, SSR*) em relação aos valores preditos. Sejam y_i as observações e \hat{y}_i as previsões:

$$SSR = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

(2) A variabilidade total é quantificada pela soma do quadrado dos desvios em relação à média (*total sum of squares, TSS*), um termo que vimos no cálculo da variância (segundo momento):

$$TSS = \sum_{i=1}^n (y_i - \mu_y)^2$$

Então a fração $\frac{SSR}{TSS}$ é a proporção desejada. Definimos R^2 como:

$$R^2 = 1 - \frac{SSR}{TSS}$$

Uma visualização intuitiva de SSR e TSS:

```
>source("aux/multiplot.R")
>doc_lmfit <- lm(hale ~ log_docs, data=uni_df)
>uni_df$preds[complete.cases(uni_df)] <- predict(doc_lmfit)
>uni_df$hale_mean <- mean(uni_df$hale,na.rm = T)
>ssr_res <- ggplot(uni_df,aes(x=log_docs,y=hale))+
  geom_point(alpha=0.5,size=3) +
  geom_segment(aes(xend = log_docs, yend = preds)) +
  geom_smooth(method="lm")+
  xlab("")+
```

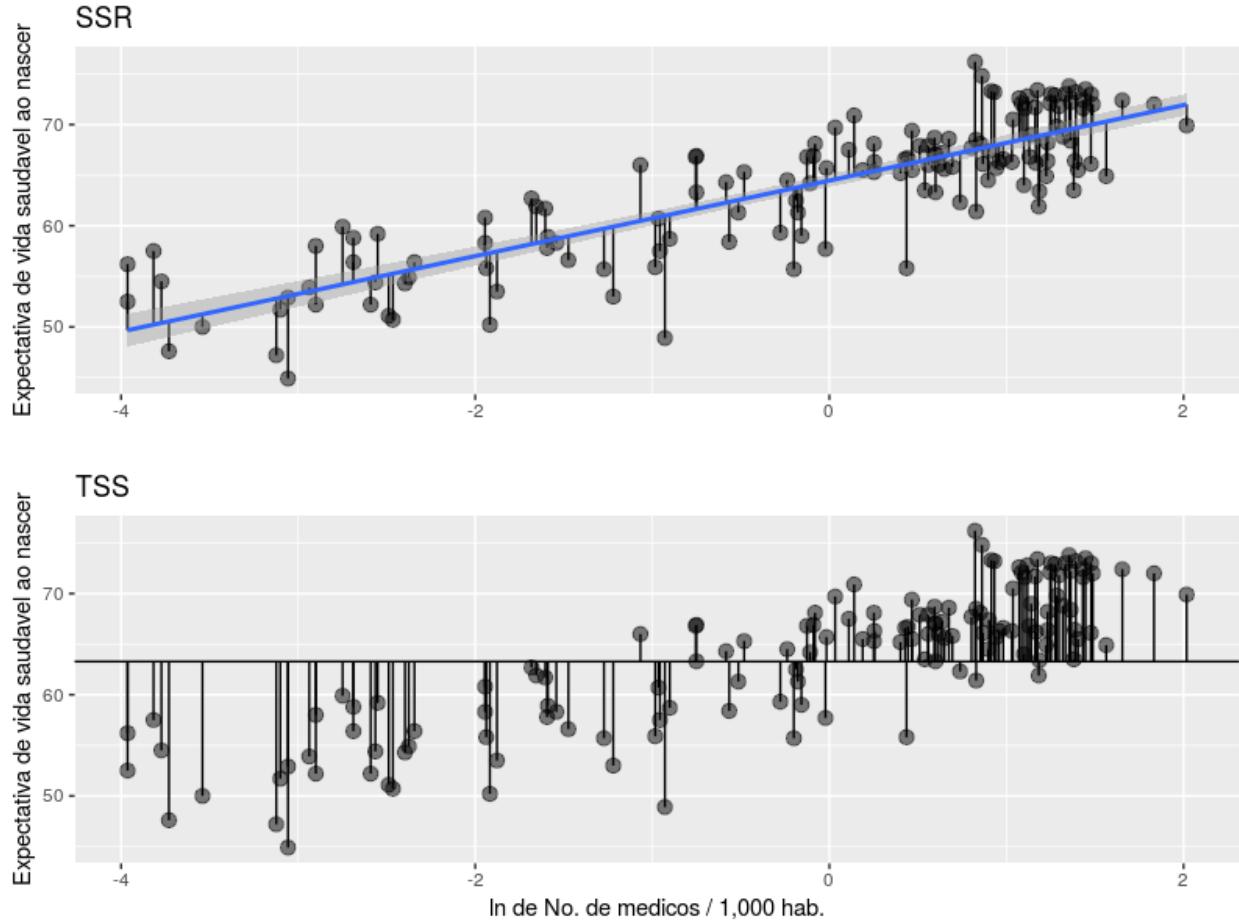


Figure 12: O quadrado da distância entre um ponto e a reta corresponde a um resíduo. Obtemos SSR e TSS somando todos os resíduos nas figuras superior e inferior, respectivamente.

```

ylab("Expectativa de vida saudável ao nascer")+
ggplot2:::ggtitle("SSR")

>tss_res <- ggplot(uni_df,aes(x=log_docs,y=hale))+
  geom_point(alpha=0.5,size=3) +
  geom_segment(aes(xend = log_docs, yend = hale_mean)) +
  geom_abline(slope = 0,intercept = 63.28165)+
  xlab("ln de No. de médicos / 1,000 hab.")+
  ylab("Expectativa de vida saudável ao nascer")+
  ggplot2:::ggtitle("TSS")

>multiplot(ssr_res,tss_res)

```

Valores de R^2 próximos a 1 indicam soma de resíduos (SSR) similar a 0. Usar a reta como guia acumula erros quase nulos. Valores de R^2 próximos a 0 indicam $\frac{SSR}{TSS} \sim 1$ e as previsões obtidas pelo modelo são tão boas quanto chutar a média para todos os casos.

```

>lm(hale ~ log_docs, data=uni_df) %>% summary
Call:
lm(formula = hale ~ log_docs, data = uni_df)

```

```

Residuals:
    Min      1Q  Median      3Q     Max 
-12.0964 -2.3988  0.3233  2.8229  8.6708 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 64.4613    0.3162 203.84 <2e-16 ***
log_docs     3.7303    0.2009  18.57 <2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.779 on 143 degrees of freedom
(119 observations deleted due to missingness)
Multiple R-squared:  0.7069, Adjusted R-squared:  0.7049 
F-statistic: 344.9 on 1 and 143 DF, p-value: < 2.2e-16

```

Para obter os valores preditos, usamos o método *predict*:

```

>head(predict(doc_lmfit))

 2      3      4      7      8      9 
59.90747 57.23226 65.39962 66.11533 69.54483 68.30608

```

É possível também obter previsões para novos valores especificando o argumento *newdata*. Para um país com 1.5 médicos/1,000 habitantes:

```

>predict(doc_lmfit,newdata = data.frame(log_docs=log(1.5)))
 1 
65.97381

```

Premissas

Existem alguns procedimentos auxiliares para checar possíveis falhas e pontos no modelo que precisam de atenção. Por exemplo, os resíduos podem ser assimétricos. Isso indica que o desempenho muda em diferentes intervalos (heteroscedacidade). Diferentes violações necessitam de atitudes diferentes, como tratar outliers ou mudar tipo do modelo. Uma lista completa de premissas, junto aos códigos em R para testá-las, está disponível no material auxiliar (*lm-asssumptions.R*)

Exercícios

1. Usando o dataset *iris*:
 - Calcule medidas de tendência central e dispersão para cada variável.
 - Execute um teste t para uma das medidas entre duas espécies.
 - Obtenha o tamanho de efeito (D de Cohen) para a diferença.
 - Faça um scatterplot entre duas medidas
 - Verifique se há correlação linear significativa entre as variáveis.
 - Se existir, ajuste um modelo de regressão linear.
 - Adicione cores de acordo com a espécie.
 - Ajuste um modelo de regressão para cada espécie.
 - Observe os valores de R^2 para cada modelo. Qual a sua impressão sobre as mudanças de performance?

Correlações e testes não paramétricos

Verificamos minuciosamente análises envolvendo a distribuição normal, a distribuição t e relações lineares. Entretanto, muitas vezes as medidas não seguem uma distribuição definida. Assim, realizar inferências usando os **parâmetros** descritos ($\mu, \sigma, t\dots$) nos levaria a conclusões erradas.

Para lidar com distribuições arbitrárias, vamos abrir mão deles e conhecer ferramentas *não-paramétricas*: o coeficiente de correlação de ranks ρ de Spearman e o teste U de Mann Whitney.

Ranks e o ρ de Spearman

Relações lineares mantêm proporções constantes e aprendemos como quantificá-las. Por outro lado, duas variáveis podem ter relações de outros tipos, não lineares. Em especial, se as medidas apresentam valores muito extremos (*outliers*) um cálculo como o anterior sofre bastante com vieses.

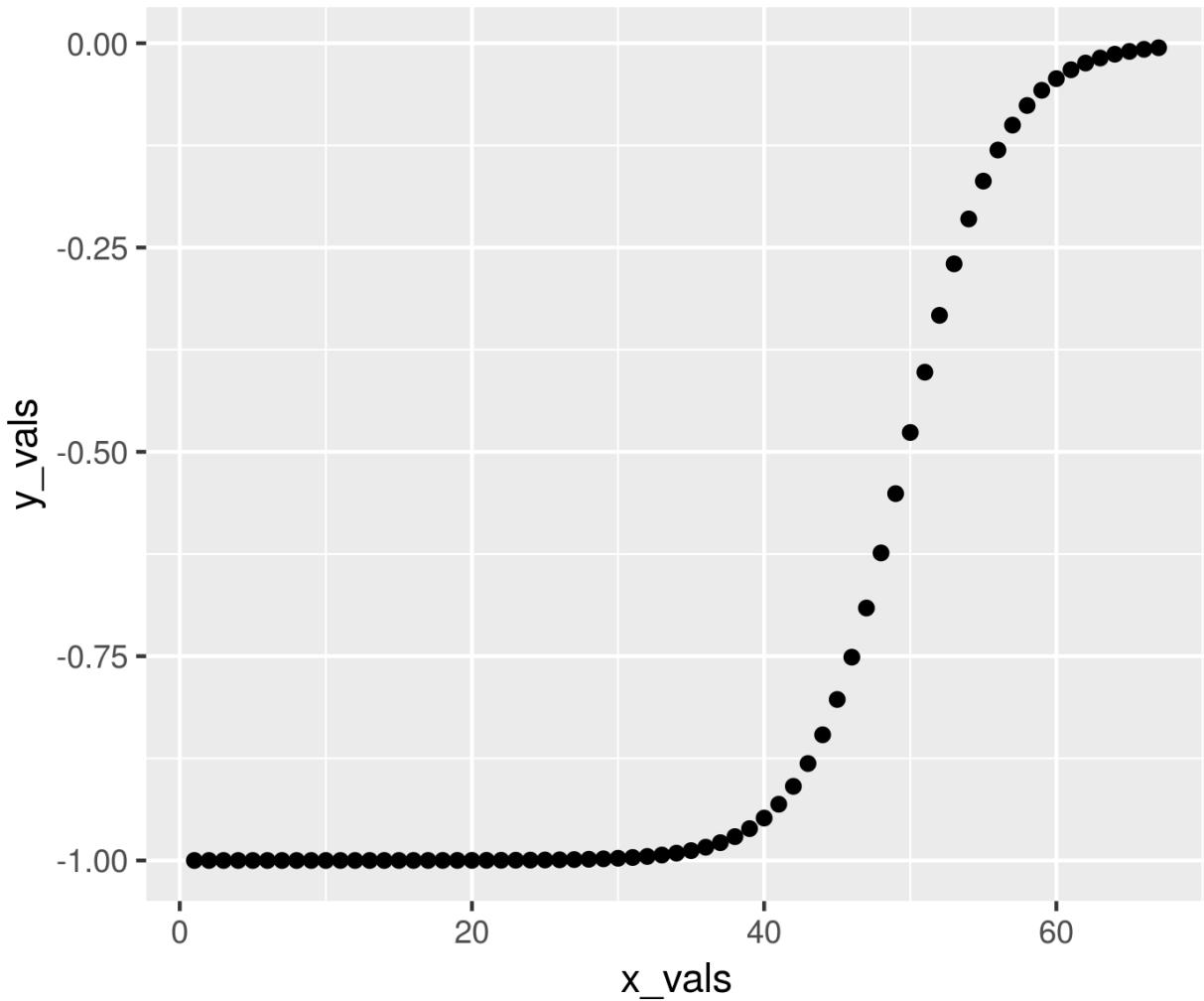
Uma simples solução para esse problema é ranquear os valores. Assim, os itens do conjunto são tratados pela sua posição em relação a outros itens, de forma independente dos valores associados. Exemplo:

$$S = (1, 3, 89, 89, 39, 209) \rightarrow S_{ranked} = (1, 2, 4, 4, 3, 5)$$

O ρ de Spearman é que o coeficiente produto-momento de Pearson aplicado aos ranks. Assim, medimos o grau em que duas variáveis aumentam (ou diminuem) em magnitude observando apenas a ordem das observações. Isto é: **maior que, igual ou menor que**. Especificamente, investigamos se há uma relação de *monotonicidade* entre elas.

Para a relação (sigmoide), entre x e y abaixo:

```
>set.seed(2600)
>sig_data <- data.frame(y_vals = -(1 / (1 + exp(seq(-10,10,by =0.3) )*100 )) ,
  x_vals = 1:67)
>ggplot(sig_data,aes(x=x_vals,y=y_vals))++
  geom_point()
```



O coeficiente de Pearson é $\rho \sim 0.850^{20}$:

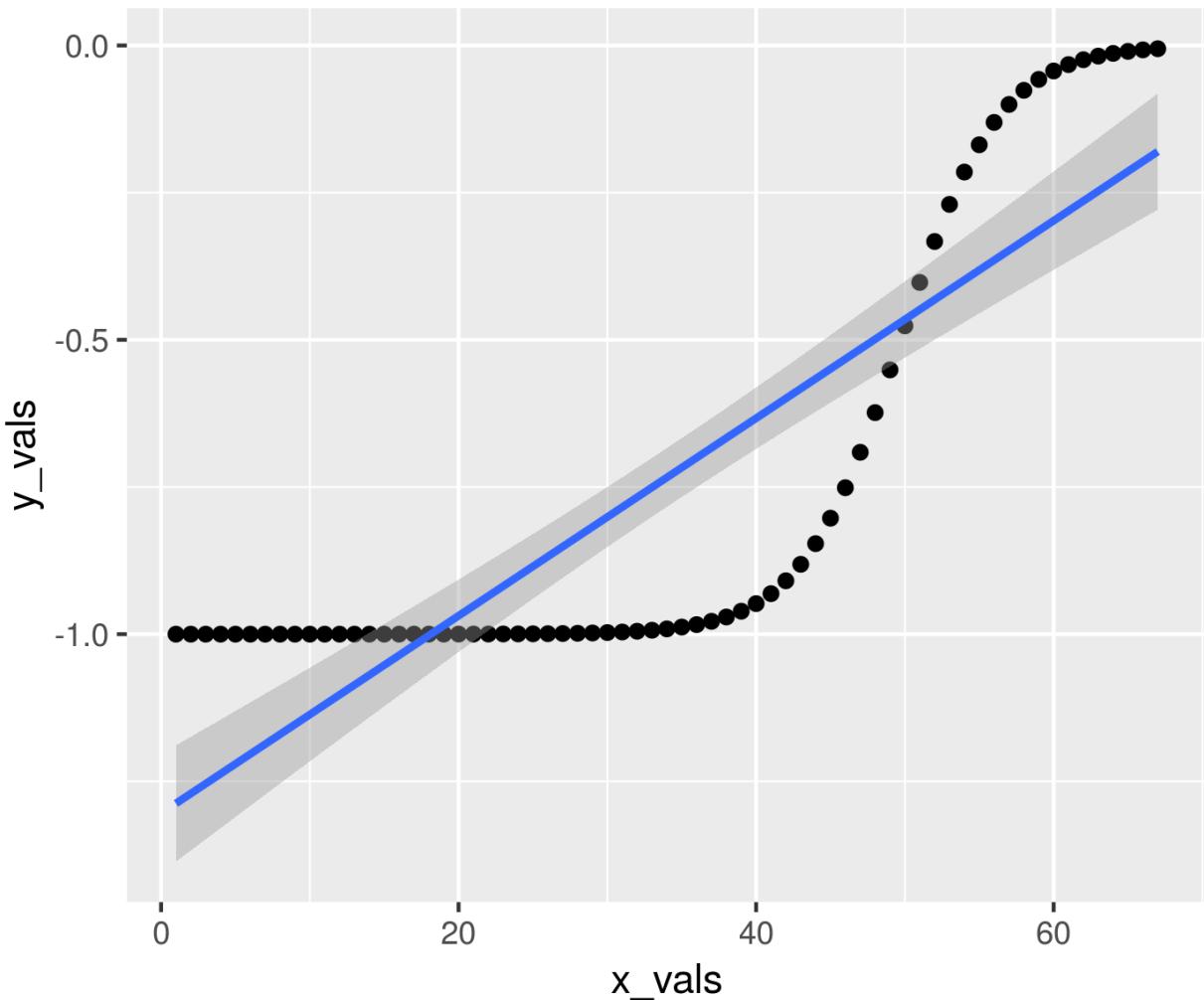
```
>cor.test(sig_data$y_vals,
+          sig_data$x_vals)

Pearson's product-moment
correlation

data: sig_data$y_vals and +sig_data$x_vals
t = 12.993, df = 65, p-value <
2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
0.7658181 0.9051711
sample estimates:
cor
0.8497162
```

²⁰Como observamos no gráfico, a correlação linear não é tão alta. O coeficiente se aproxima de 1 $\rho \sim 0.850$ pois os desvios superiores compensam simetricamente os inferiores. O exemplo reforça a importância de plotar os dados para um melhor entendimento (ver Quarteto de Anscombe).

```
>ggplot(sig_data,aes(x=x_vals,y=y_vals))+  
  geom_point() +  
  geom_smooth(method="lm")
```



Como a relação é perfeitamente monotônica, os pares ordenados (x_i, y_i) sempre possuem o mesmo rank. O quinto valor mais alto em x é também o quinto valor mais alto em y. Portanto, o coeficiente de Spearman é 1:

```
>cor.test(sig_data$y_vals,  
+          sig_data$x_vals,method = "spearman")  
  
Spearman's rank correlation rho  
  
data: sig_data$y_vals and sig_data$x_vals  
S = 0, p-value < 2.2e-16  
alternative hypothesis: true rho is not equal to 0  
sample estimates:  
rho  
 1
```

O coeficiente ρ de Spearman é preferível quando as medidas diferem bastante de uma distribuição normal. Também oferece robustez quando é necessário lidar com *outliers*.

Teste U de Mann-Whitney

O teste U de Mann-Whitney faz uso da estatística U para fazer inferências. O racional é idêntico ao do teste t de Student.

Estabelecemos hipótese nula H_0 e hipótese alternativa H_1 .

Então, calculamos a probabilidade de nossas observações acontecerem caso a hipótese nula seja verdadeira. Desta vez, usaremos a estatística U. Lembremos que a estatística t era calculada com base em parâmetros extraídos da amostra:

$$t = Z/s = (\mu' - \mu) / \frac{\sigma}{\sqrt{n}}$$

A estatística U não depende de parâmetros, sendo calculada com base em cada observação.

Primeiro, calculamos os ranks de cada medida r_i unindo as observações das amostras A e B, de tamanhos amostrais n_a e n_b em apenas um conjunto ($N_{tot} = n_a + n_b$).

Depois, separamos novamente as amostras e calculamos a soma dos ranks em cada grupo, chamadas R_a e R_b . A estatística U é dada pela seguinte expressão:

$$U_a = R_a - \frac{n_a(n_a + 1)}{2}$$
$$U_b = R_b - \frac{n_b(n_b + 1)}{2}$$

Usamos o menor valor de U para consultar a probabilidade (valor p) correspondente para a hipótese nula.

O termo $\frac{n(n+1)}{2}$ corresponde à soma mínima dos ranks para a amostra.

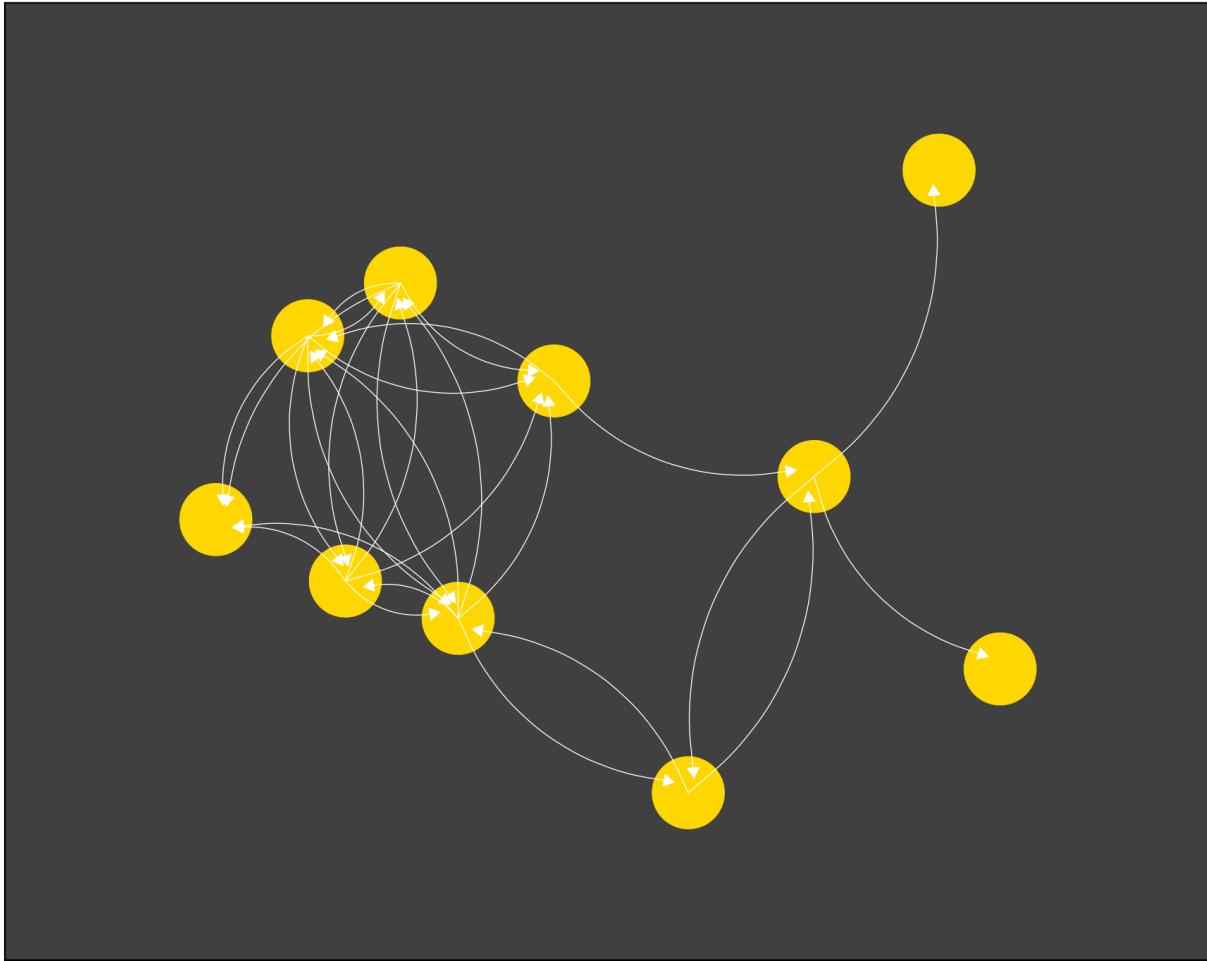
Os ranks são uma sequência regular (1, 2, 3, ...), de forma que a soma de todos os valores é idêntica à soma de uma progressão aritmética de N termos.

$$\Sigma_{ranks} = \frac{N(N + 1)}{2}$$

Enquanto R_i corresponde à soma dos ranks calculados com as duas amostras, o termo acima corresponderia à soma mínima dos ranks para uma amostra, caso os ranks ocupassem a sequência inicial $A = (1, 2, 3, 4, \dots, n_a)$ na amostra conjunta.

A definição para o teste não é unânime na literatura, de forma que alguns autores e softwares (e.g. R) implementam o cálculo com a subtração acima e outros (e.g. S-PLUS) não o fazem.

Em R, as funções **dwilcox(x,m,n)** e **pwilcox(q,m,n)** retornam a distribuição e a densidade cumulativa para a estatística U correspondente a amostras com tamanhos m e n. **wilcox.test(x,y,...)** é a implementação base do teste de Mann Whitney. O teste de Mann Whitney é o teste de Wilcoxon de duas amostras.



Capítulo 3 : Análise multivariada, grafos e inferência causal

Intrudução

Até este ponto, aplicamos modelagem matemática para uma ou duas variáveis aleatórias. Procedimentos diferentes foram empregados para correlação, comparação e regressão. Neste capítulo, incorporaremos novas medidas e construtos. Começamos expandindo a regressão linear simples.

Regressão múltipla

Nos modelos lineares simples, calculamos parâmetros para um intercepto β_0 , inclinação da reta β_1 e variância dos erros σ_ϵ^2 . No exemplo apresentado, relacionamos o número de médicos (n) com a expectativa de vida saudável $hale$ em um país.

$$y_i = \beta_0 + \beta_1 x_i + \epsilon$$

$$hale_i = \beta_0 + \beta_1 n_i + \epsilon$$

Na *regressão linear múltipla*, introduzimos mais uma variável preditora. Em nosso exemplo, poderia ser o valor do IDH do país:

$$hale_i = \beta_0 + \beta_1 n_i + \beta_2 IDH'_i + \epsilon$$

Em geral, temos dois objetivos:

(1) melhorar a performance do modelo ao adicionar informações pertinentes; (2) examinar o efeito sobre as demais variáveis preditoras.

O primeiro objetivo é intuitivamente óbvio, entretanto precisamos ter cuidado com redundância de informações. Especificamente, há uma troca quase inevitável entre complexidade e robustez do modelo. Acrescentar variáveis ou usar classes de relações mais flexíveis implica dar liberdade para um sobreajuste aos dados. Isto é, nosso modelo aprenderá idiossincrasias sobre os dados disponíveis (datasets WHO e World Bank) e não sobre a relação entre as abstrações (e.g.expectativa de vida saudável). Veremos nas próximas sessões como mitigar esse problema.

Para o caso da regressão linear múltipla, podemos verificar se há colinearidade (relação linear) entre variáveis preditoras. Se as variáveis preditoras são altamente correlacionadas, é provável que estejamos fornecendo informações redundantes ao modelo, o que é nocivo. Existem alguns indicadores que podem ajudar a tomar essa decisão.

Comumente, observamos o VIF *Variance inflation factor*.

VIF

A intuição aqui é de que se as variáveis são muito relacionadas $X_1 \sim X_2$, os valores de β estimados em $Y = \beta_1 X_1 + \beta_2 X_2 + \dots$ não serão únicos. Por exemplo, poderíamos trocar β_1 por β_2 e a solução permaneceria praticamente inalterada. O VIF estima a colinearidade em relação à combinação de outros preditores usados.

Para calcular o VIF referente a um preditor X' , ajustamos uma nova regressão, em que a variável resposta é X' e as preditoras são as outras variáveis preditoras. O VIF é dado por: $\frac{1}{1-R^2}$, sendo R^2 o coeficiente de determinação da regressão, como calculamos antes.

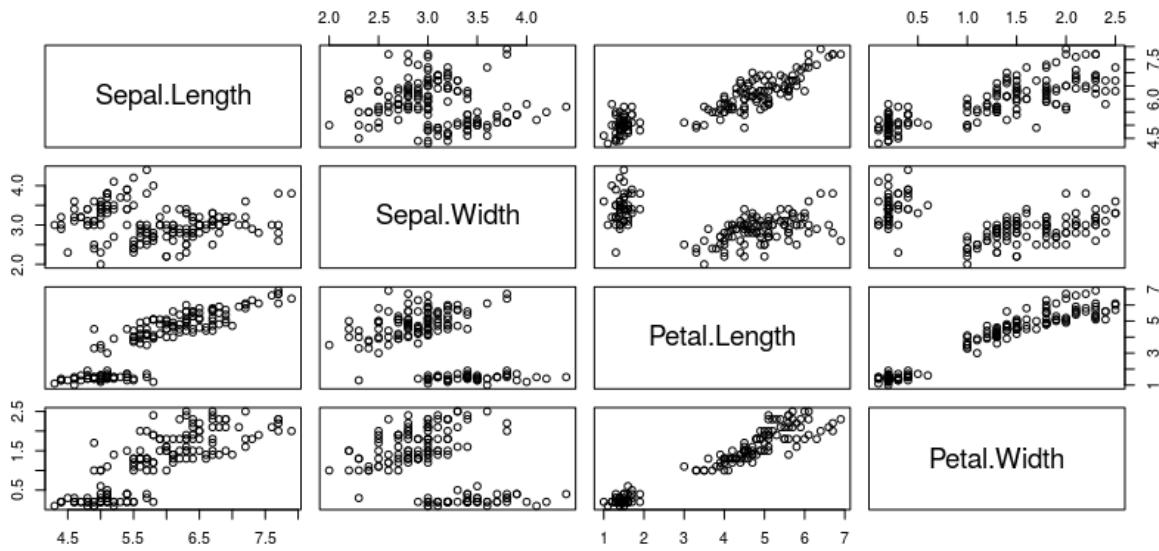
Valores de VIF altos refletem valores de R^2 altos, isto é: a combinação linear de outras variáveis explicaria muito bem a variável preditora em questão. Não há regra canônica, porém $VIF > 10$ ($R^2 = 0.9$) e $VIF > 5(R^2 = 0.8)$ são citados como fronteiras indicando colinearidade inaceitável.

A função `vif` do pacote `car` implementa o procedimento. Ajustamos uma regressão linear múltipla para o comprimento das sépalas no dataset `iris` a partir de outras 3 variáveis. Podemos verificar que há colinearidade ($VIF_{pet.leng.} \sim 15.1$, $VIF_{pet.wid.} \sim 14.2$) entre largura e comprimento da pétala. Por outro lado, a colinearidade com o comprimento da sépala é baixa ($VIF_{pet.wid.} \sim 1.3$).

```
>car::vif(lm(Sepal.Length ~ Petal.Length + Petal.Width + Sepal.Width,
              data=iris))
Petal.Length  Petal.Width  Sepal.Width
 15.097572    14.234335    1.270815
```

Se há colinearidade, é recomendado remover um dos preditores para eliminar a redundância. Como sempre, a inspeção visual ajuda.

```
>pairs(iris[,1:4])
```



Como podemos ver, usar duas variáveis preditoras (regressão múltipla) não colineares aumenta a performance do modelo em relação à regressão simples ($R^2 \sim 0.84$ vs $R^2 = 0.76$).

```
>lm(Sepal.Length ~ Petal.Length,
+     data=iris) %>% summary

(...)

Multiple R-squared:  0.76, Adjusted R-squared:  0.7583
F-statistic: 468.6 on 1 and 148 DF,  p-value: < 2.2e-16

>lm(Sepal.Length ~ Petal.Length + Sepal.Width,
+     data=iris) %>% summary
(...)

Multiple R-squared:  0.8402,   Adjusted R-squared:  0.838
F-statistic: 386.4 on 2 and 147 DF,  p-value: < 2.2e-16
```

Um outro objetivo para a regressão múltipla é examinar o efeito modificador das variáveis acrescentadas. Em especial, é comum incluir variáveis auxiliares para corrigir estimativas.

Exemplo: queremos estimar um parâmetro β_1 para a relação entre altura e peso. Ajustamos um modelo: $Altura = \beta_0 + \beta_1 * Peso + \epsilon$. Entretanto, sabemos que a altura média de homens é maior que a de mulheres. Ao examinar a relação entre a altura e peso, podemos incluir a variável *sexo* no modelo, $Altura = \beta_0 + \beta_1 * Peso + \beta_2 * Sexo + \epsilon$.

Nossa estimativa de β_1 é modificada de maneira a levar em conta os efeitos do sexo.²¹

Veremos uma formalização desse conceito a seguir, com o procedimento para examinar mediação.

²¹Sexo é uma variável dicotômica (macho/fêmea). Costumamos codificá-las de forma binária (0/1; e.g: macho = 1 / fêmea = 0). Assim, um sujeito macho terá a estimativa de altura acrescida em $\beta_2 * 1$, enquanto fêmeas terão este termo zerado $\beta_2 * 0$. Chamamos esse truque de *dummy coding*.

Costumeiramente, traduzimos os procedimentos acima afirmando que a estimativa para “*a relação entre X e Y é controlada para confundidores [A, B e C]*”. A esse ponto, fica óbvio que a simplificação linguística é perigosa. A falta de cautela em traduzir abstrações matemáticas para linguagem natural é responsável pela injusta fama da estatística como ferramenta para enganos.

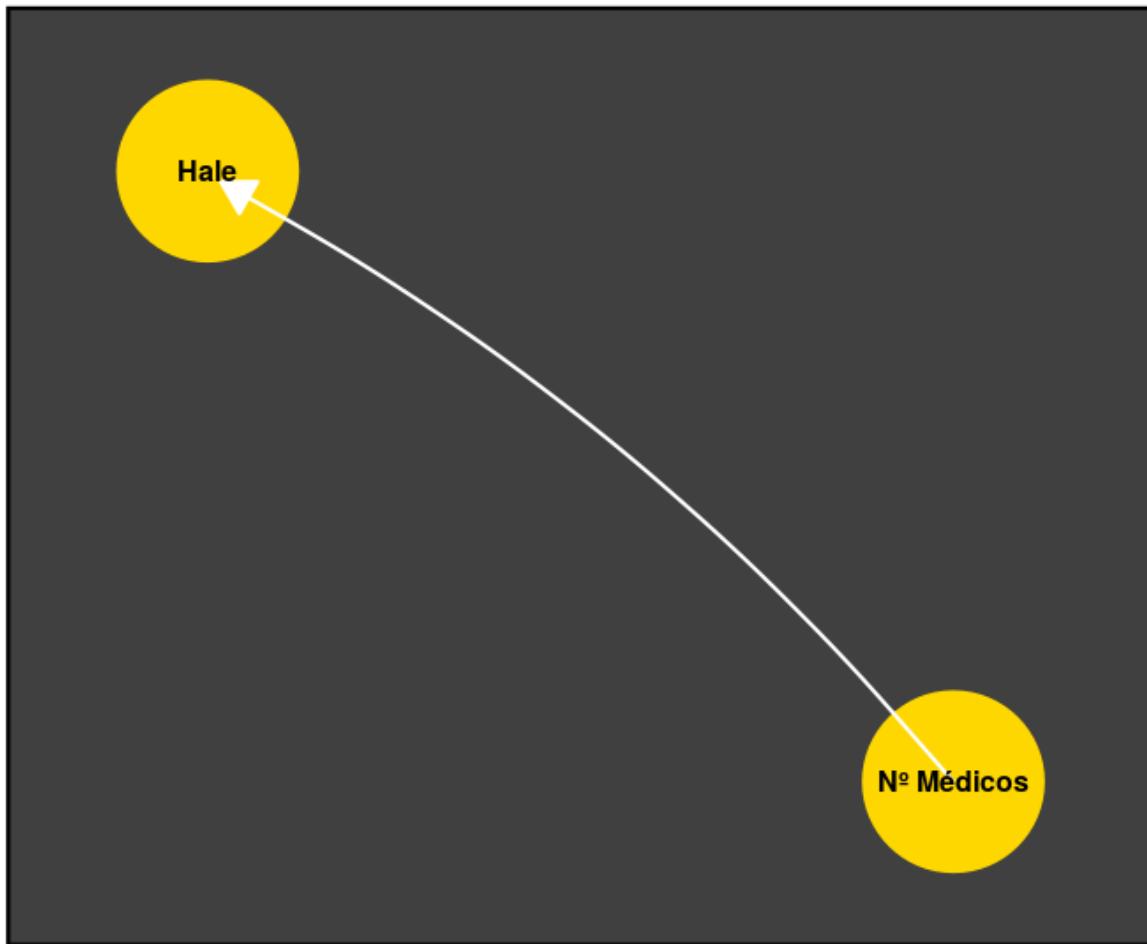
Assim como o valor p é indevidamente interpretado muitas vezes, o “controle para confundidores” nada mais é que o ajuste de estimativas selecionando outras variáveis arbitrariamente para uma combinação linear.

O prejuízo é herdado por todas as disciplinas que usam métodos quantitativos. Pior, criamos a possibilidade para testar múltiplas combinações de confundidores. Nas mãos de alguém incauto ou mal intencionado, testes sucessivos têm grandes chances de alcançar resultados “significantes”. De uma forma global, vemos uma série de verdades transitórias ventiladas na comunidade científica (e na mídia leiga), resultantes de análises mal conduzidas.

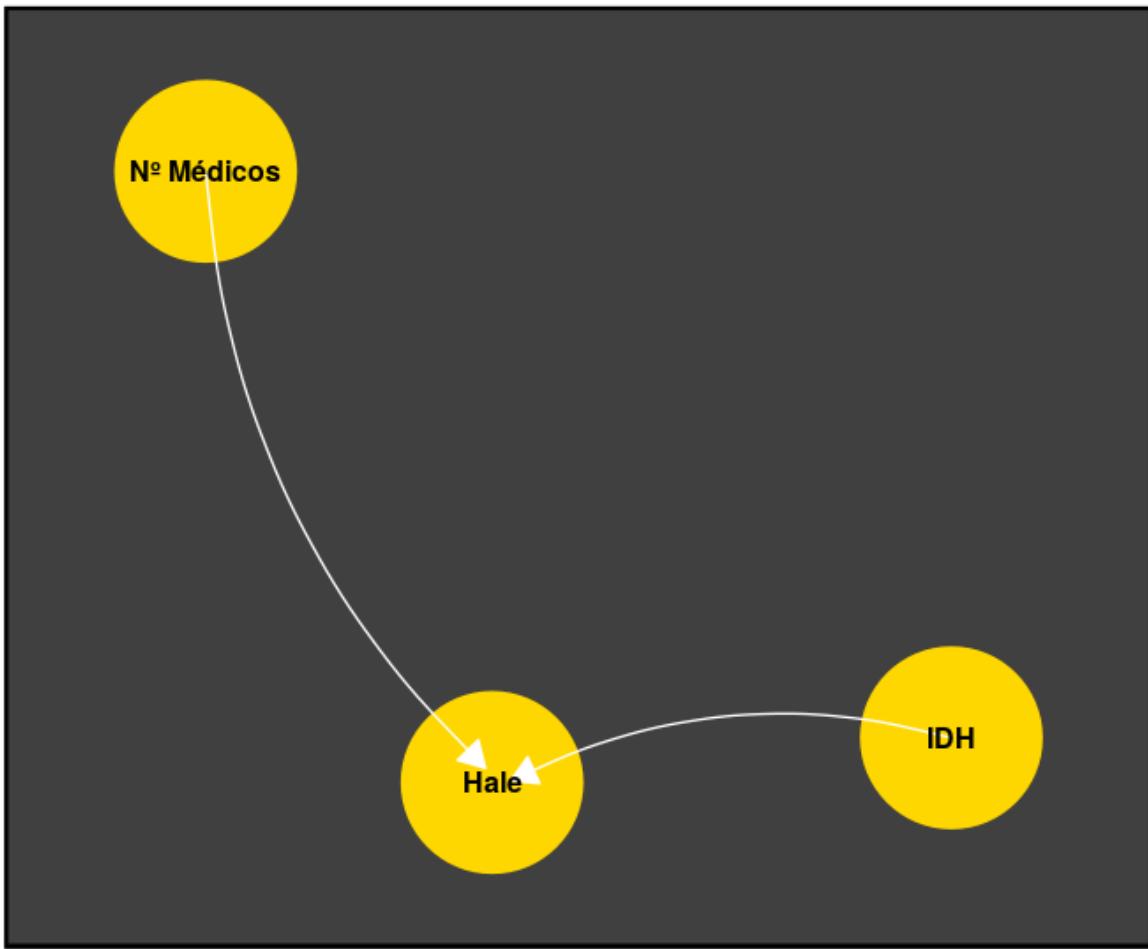
Para inferências desse tipo, é recomendado que os confundidores sejam mitigados experimentalmente (e.g. randomização) sempre que possível.

Grafos e trajetórias causais

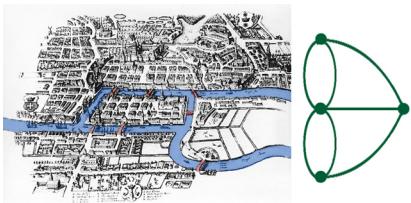
Podemos usar os diagramas a seguir para ilustrar uma regressão linear simples:



Ou múltipla com dois preditores:



É fácil relacionar *nodos com variáveis* e *conexões com relações* descritas pelas equações estimadas. Formalmente, tratamos essas abstrações com o nome de **grafos**. O campo começou a ser tratado por Euler em 1736. Chamamos os pontos de nodos, ou vértices, e as ligações de arestas (*edges*). Cada aresta conecta dois nodos. O conceito foi usado para resolver o problema das pontes de Königsberg. Dada uma série de pontes conectando partes diferentes da cidade, fazer um percurso que cruzae cada uma apenas uma vez?



Euler mostrou que era impossível. Note que não usamos distâncias. Apenas descrevemos como elementos são conectados. Podemos atrelar diversas estruturas. Os grafos acima, por exemplo, são direcionados e possuem equações vinculadas.

As equações e procedimentos de que lançamos mão anteriormente são soluções equivalentes às representações gráficas. É possível generalizar a ideia, usando diagramas para tratar matematicamente formulações de teorias científicas.

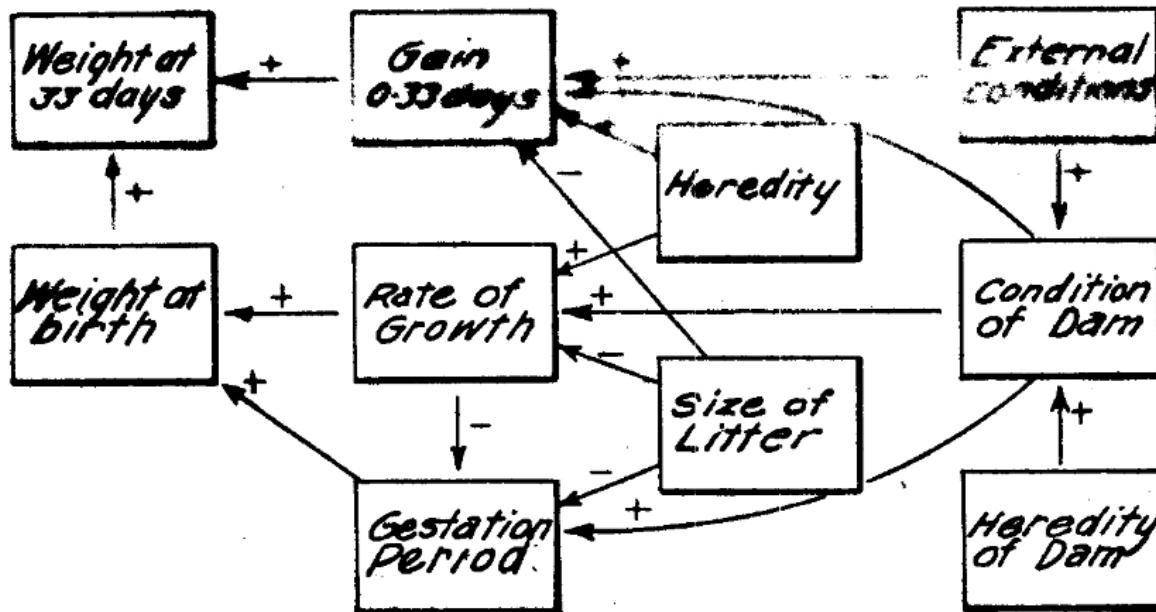


FIG. 1.—Diagram illustrating the interrelations among the factors which determine the weight of guinea pigs at birth and at weaning (33 days).

Figure 13: Diagrama mostrando relação entre fatores influenciando peso de um porquinho-da-índia. Wright, 1921

Grafos e trajetórias causais

"The ideal method of science is the study of the direct influence of one condition on another in experiments in which all other possible causes of variation are eliminated.", Sewall Wright, Correlation and Causation, 1921

A pouco conhecida origem deste campo está no trabalho de um geneticista, Sewall Wright. Ele assumiu que a correlação entre variáveis é resultante da influência de muitas trajetórias causais. Então, propôs uma forma de medir a influência de cada trajetória sobre uma variável-alvo.

Usando grafos direcionados (as conexões têm uma origem e um destino), é atrelar as noções de correlação e regressão de forma a ilustrar caminhos causais entre relações lineares. Sewall começou usando apenas grafos acíclicos (sem conexões levando ao ponto de origem do percurso) em condições restritas.

Décadas depois, o campo foi extrapolado para outros cenários mais gerais. Em específico, o boom de disponibilidade de poder computacional nas décadas de 1960 e 1970 impulsionou o surgimento de estimadores diversos para parâmetros nesses modelos.

É esperado que a quantidade de parâmetros cresça conforme a complexibilidade.

Um trabalho valoroso foi feito por Judea Pearl para unificar as abordagens. Pearl mostrou que muitos frameworks são situações especiais de modelos de equação estrutural (SEM, structural equation models), os quais também englobam versões não paramétricas. Por exemplo, o sistema causal de Rubin é equivalente a SEM: todos os teoremas podem ser deduzidos usando algumas identidades entre as abordagens.

Ele também escreveu textos compreensivos alinhando a matemática aplicada a uma base epistemológica. É especialmente digno de nota o conceito de *contrafactual*. Para estimar um efeito causal, imaginamos quais seriam as condições em um cenário sem ação do agente causal. Pearl conduz um cauteloso estudo lógico-semântico das definições na tentativa de construir um sistema coerente de pesquisa empírica. Uma explicação completa foge do escopo do livro, entretanto conheceremos algumas aplicações.

Mediação e Moderação

Mediação

Uma ideia curiosa é de que uma variável pode estar intermediando a ação de outra sobre um desfecho. Um exemplo clássico é a relação entre o hábito de fumar e câncer. Sabemos que existe uma ação nociva pela temperatura do ar inalado, assim como dos componentes químicos absorvidos.

Em modelos de mediação, tentamos quantificar a porção que é explicada por uma variável intermediária. Para tanto, empregamos o seguinte procedimento:

1. Verificar plausibilidade de relações individualmente através modelos de regressão entre variáveis de interesse.

Ajustamos 3 modelos:

- (1) variável independente e variável alvo ($Y \sim X_1\beta_1$),
- (2) variável mediadora e variável alvo ($Y \sim X_2\beta_2$),
- (3) variável independente e variável mediadora ($X_2 \sim X_1\beta_3$).

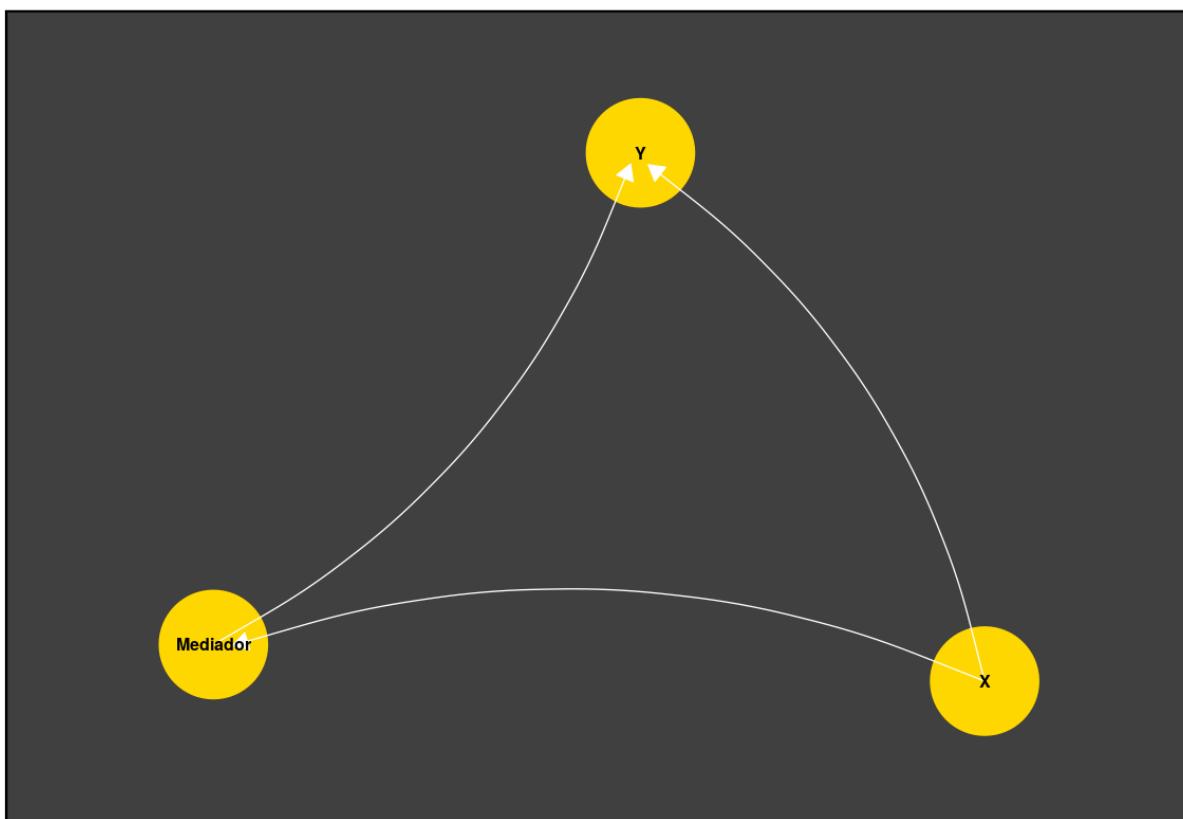
O efeito direto da variável independente sobre a variável alvo é quantificado β_1 .

2. Verificar mudanças obtidas pela introdução da variável mediadora.

Ajustamos um quarto modelo (4), com a combinação linear de variável independente e variável mediadora. Observamos então a diferença entre o novo (β'_1) coeficiente de X_1 e o antigo (β_1) $Y = X_1\beta'_1 + X_2\beta_4$.

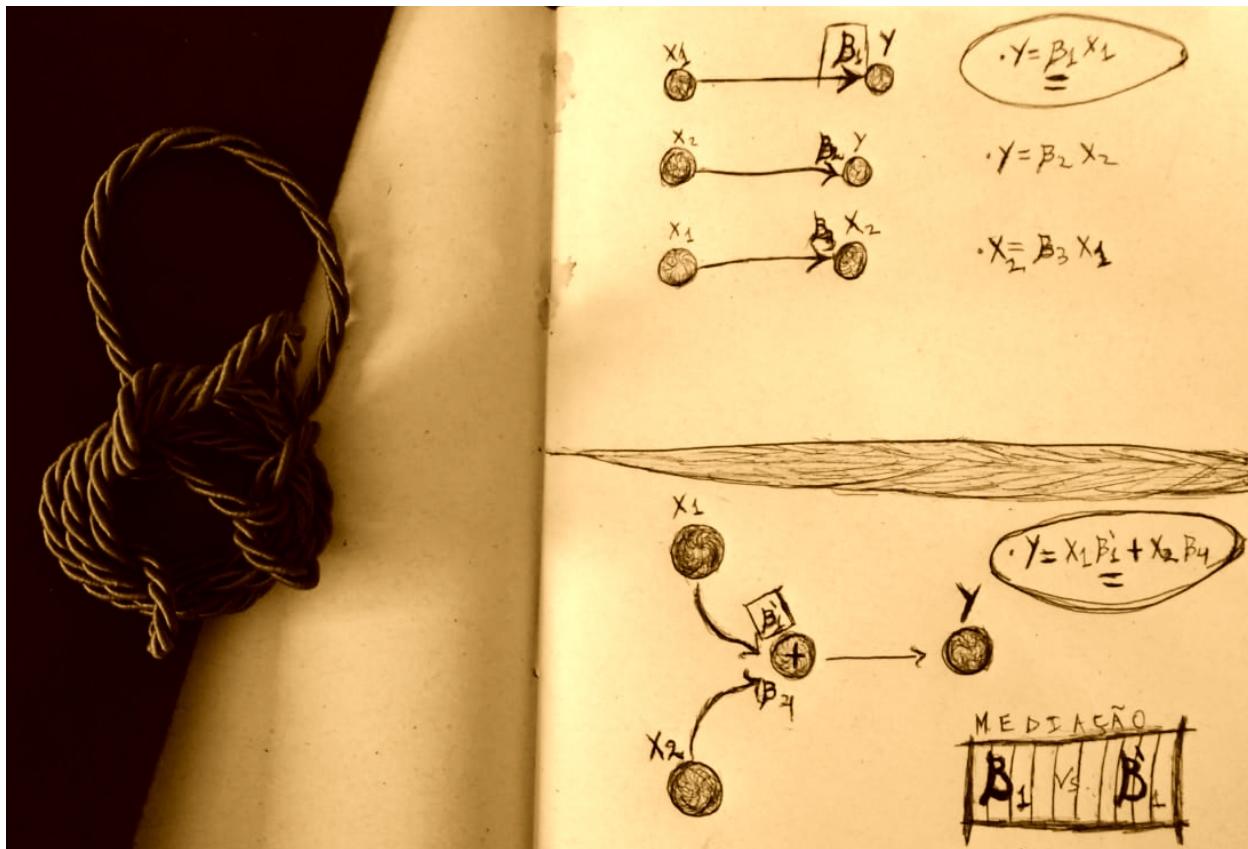
Caso exista mediação, espera-se que o coeficiente β'_1 seja não significativo ou que possua magnitude bastante reduzida em relação ao coeficiente do efeito direto β_1 .

Segundo o exemplo sugerido, espera-se que exista uma relação entre hábito de fumar e câncer. Ainda, espera-se que a inclusão de um mediador (e.g. concentração de nicotina) explique parte do efeito, reduzindo o coeficiente de X_1 . O diagrama a seguir expressa a ideia contida no processo desejado.



O diagrama acima ilustra passos rigorosamente. As 3 regressões para checar premissas estão na sessão

superior e a regressão múltipla no setor inferior). Foram suprimidos termos de erro. Estimativas para a relação entre X_1 e Y são $\hat{\beta}_1$ e $\hat{\beta}_1'$ grifados nas equações. O comportamento desses parâmetros define as conclusões sobre o modelo de mediação.



Não há garantias de que os sistemas reais se comportarão seguindo os parâmetros estimados. Usamos regressão múltipla para estimar o efeito parcial atribuído aos mediadores, porém a retirada desses fatores no fenômeno real pode resultar em alterações no sistema não previstas pelo modelo.

A certeza dependeria de uma descrição bastante acurada do fenômeno pelas regressões ($R^2 \sim 1$), o que raramente é verificado fora de fenômenos físicos mais simples.

Portanto, é recomendável que ajustes sejam feitos na fase experimental. Em nosso exemplo, isso implicaria em controlar a concentração de nicotina absorvida *in vivo*. Obviamente, razões éticas e limitação de recursos precluem muitas vezes a manipulação direta do objeto de estudo. Métodos tais como o descrito, ainda que frágeis, permitem estudar interações e relações causais. Entretanto, é necessário atenção aumentada ao fazer conclusões e, especialmente, ao traduzi-las para linguagem natural.

Em R:

```
>fit_yx1 <- lm(y ~ x1, data)
>fit_yx2 <- lm(y ~ x2, data)
# Mediation
>fit_yx1x2 <- lm(y ~ y1 + y2)
>summary(fit_yx1)
(...)
>summary(fit_yx2)
```

```
(...)
>summary(fit_yx1x2)
(...)
```

A diferença numérica entre valores de β_{x_1} é a magnitude do efeito indireto (*Ind. Effect*). Podemos usar uma estimativa do erro padrão para derivar uma estatística t e um valor p associados (teste de Sobel). Usando libs do CRAN:

```
>library(bda)
>library(multilevel) # dataset bh1996
>data(bh1996)

# LEAD : Clima de liderança
# WBEING : Bem-estar
# HRS : Horas de trabalho

# Clima de liderança media relação entre horas de trabalho e bem-estar?

>sobel(pred=bh1996$HRS,med=bh1996$LEAD,out=bh1996$WBEING)
$`Mod1: Y-X`
      Estimate Std. Error   t value   Pr(>|t|) 
(Intercept) 3.51693620 0.052902697 66.47934 0.000000e+00
pred        -0.06523285 0.004590274 -14.21110 3.078129e-45
$`Mod2: Y-X+M` 
      Estimate Std. Error   t value   Pr(>|t|) 
(Intercept) 1.86832973 0.06413083 29.13310 1.024201e-176
pred        -0.04311316 0.00421918 -10.21837 2.382257e-24
med         0.48386196 0.01242129 38.95426 4.967825e-302
$`Mod3: M-X` 
      Estimate Std. Error   t value   Pr(>|t|) 
(Intercept) 3.40718349 0.045154735 75.45573 0.000000e+00
pred        -0.04571488 0.003917997 -11.66792 3.488366e-31

$Indirect.Effect
[1] -0.02211969
$SE
[1] 0.001978985
(...)

>mediation.test(iv = bh1996$HRS,mv = bh1996$LEAD,dv = bh1996$WBEING)
      Sobel      Aroian      Goodman
z.value -1.117729e+01 -1.117391e+01 -1.118067e+01
p.value  5.267356e-29  5.471647e-29  5.070460e-29
# Aroian e Goodman são outros testes para o parâmetro de efeito indireto
```

Exercícios

1. Examine o VIF da regressão múltipla usada no processo de mediação que com banco de dados *bh1996*.
 - Há colinearidade entre mediador e preditor principal?
2. Examine a mudança de performance (e.g. R^2) após inclusão do mediador no modelo.
3. Se a variável mediadora M explicar as mesmas vias causais que a variável preditora X_1 , é esperado que essa mudança seja grande? Discuta.
3. Usando dados à sua escolha:
 - Ajuste uma regressão linear simples
 - Adicione outro preditor (regressão linear múltipla)
 - Verifique se há colinearidade
 - Cheque outras premissas observando o material auxiliar **/aux** (e.g. independência dos erros com Durbin-Watson)
 - Teste uma relação de mediação usando 3 variáveis

Moderação e Interações

Modelos incluindo termos de moderação são aqueles que incluem **interação** entre variáveis. Como discutimos antes, a relação entre hábito de fumar e câncer pode ser explicada por fatores intermediários, como a concentração de nicotina e presença de variantes genéticas de risco.

Podemos supor que a concentração de nicotina inalada diariamente tenha um efeito independente. Igualmente, uma configuração genética tem efeito causal por si.

$$Risk = Nicotina * \beta_1 + Genes_{(+)}\beta_2$$

Em moderação, adicionamos um termo à nossa combinação linear. É um coeficiente para a multiplicação entre variáveis independentes.

$$Risk = Nicotina * \beta_1 + Genes_{(+)}\beta_2 + Nicotina * Genes_{(+)}\beta_3$$

Será que *fumar e ter genes de risco* é diferente da combinação do efeito de ambos em separado?

Esse é um dos poucos casos em que é mais fácil observar o aspecto algébrico antes. Estamos multiplicando os valores de preditores X_1 e X_2 . Se ambos tiverem mesmo sentido (+ ou -), a interação terá efeito positivo. Caso contrário, negativo. Ainda, vemos que as magnitudes são multiplicadas. O coeficiente β_3 quantifica essa multiplicação em relação ao efeito em y , seja alterando o sentido (β_3 negativo) ou escalando o valor absoluto.

$$y = X_1 * \beta_1 + X_2 * \beta_2 + X_1 X_2 \beta_3$$

A relação de y em relação com cada preditor deixa de ser linear. Como podemos verificar analisando as derivadas parciais. Para $\frac{d}{dx_1}$:

$$\frac{d}{dx_1}(y) = \frac{d}{dx_1}(x_1\beta_1 + x_2\beta_2 + x_1x_2\beta_3)$$

O segundo termo não depende de X_1 , então:

$$\frac{d}{dx_1}(y) = \frac{d}{dx_1}(\beta_1 + x_2\beta_3)$$

A inclinação (*slope*), que antes era uma constante (linha reta) β_1 passa a ter um termo somado, que é a multiplicação da constante estimada β_3 pelo valor de x_2 . Então temos inclinação diferente para cada valor de moderador!

Esses detalhes tornam a interpretabilidade dos coeficientes difícil. Normalmente, são usadas heurísticas, como centralizar os dados em torno da média, para simplificar o contexto.

Medidas latentes e análise fatorial

Considere o problema de medir algo inacessível através de meios secundários.

Por exemplo, o conceito de *qualidade de vida* é facilmente concebível, apesar de não estar atrelado a uma medida tangível, tal qual *altura* ou *tamanho do fêmur*.

Uma série de métodos foi desenvolvida para lidar com a tarefa de estimar *variáveis latentes*. Em especial, esses modelos são muito populares entre psicometristas. Podemos aplicar modelos de variáveis latentes para muitos contextos.

Isso é feito quando usamos respostas corretas em um teste formulado por especialistas para quantificar uma habilidade. A *Teoria de Resposta ao Item* é usada em testes como ENEM (Brasil), SAT e GRE (EUA). Relacionamos a estimativa de habilidade (θ) com a probabilidade de acertar (1) ou errar (0).

Traços de personalidade também podem ser estudados dessa maneira. Podemos atribuir um grau de extroversão F de uma pessoa através de sua pontuação em uma bateria de testes X_1, X_2, X_3, \dots relacionados a esse atributo.

Sejam os items:

1. Gosto de estar com outras pessoas (1 a 7)
2. Costumo conversar com desconhecidos (1 a 7)
3. Costumo expressar minhas opiniões (1 a 7)
4. Sou considerado(a) uma pessoa comunicativa (1 a 7)

A pontuação de um indivíduo será uma sequencia de 4 números. Um indivíduo muito extrovertido pode pontuar (7,7,6,7) e um introvertido (2,3,2,1). Podemos pensar que a série de medidas é influenciada por um construto (extroversão).

Análise fatorial parte da premissa de que a **covariância** nas medidas diretas é fruto das **influências latentes compartilhadas** pelos items. Assim, podemos estimar um parâmetro λ para a relação entre cada item e o traço latente F . Para isso, usaremos a matriz de covariâncias.

Os valores de λ quantificam a relação entre items e fatores latentes e servem, por exemplo, para selecionar items mais relacionados aos traços alvo em um instrumento psicométrico.

Como na regressão linear, o modelo descreve cada medida como uma combinação entre score individual para fator latente F multiplicado pelo peso para o item λ_{Item1} e erros.

A medida do item 1 para o i -ésimo sujeito considerando n fatores latentes F_n é:

$$x_{1,i} = \sum_1 nF_i\lambda_n + \epsilon$$

Assim, falamos em mais de um construto latente. Ao invés de trabalhar com um grande fator latente (extroversão), podemos ligar os quatro items anteriores a dois conceitos menos específicos: “sociabilidade” e “expressividade”.

O valor dos 4 itens para o n -ésimo sujeito, considerando dois fatores latentes, com pesos λ_i, λ_i' é:

$$\begin{aligned} x_{1,n} &= F_{1,n}\lambda_1 + F_{2,n}\lambda'_1 + \epsilon \\ x_{2,n} &= F_{1,n}\lambda_2 + F_{2,n}\lambda'_2 + \epsilon \\ x_{3,n} &= F_{1,n}\lambda_3 + F_{2,n}\lambda'_3 + \epsilon \\ x_{4,n} &= F_{1,n}\lambda_4 + F_{2,n}\lambda'_4 + \epsilon \end{aligned}$$

Podemos perceber que a matriz Λ terá 8 elementos, com 4 pesos para o fator F_1 e 4 pesos para o fator F_2 . Sabendo os dois scores latentes de cada sujeito, seria possível reconstruir as observações com algum grau de perda. Perceba que expressamos qualquer item com apenas dois parâmetros (F_1 e F_2). As informações em nosso dataset poderiam então então ser condensadas de $[nx4]$ dimensões para $[nx2]$.

Para estimar os parâmetros acima, supomos que a variância de **cada item possui uma variância intrínseca e uma variância compartilhada, que é determinada pelos fatores latentes**. Usamos uma matriz de covariâncias entre os items para estimar os pesos dos fatores latentes. Além disso, estimamos parâmetros relacionados à diagonal da matriz (variâncias).

Em nosso exemplo, teríamos uma matriz de dimensão [4x4].

$$CovMat_x = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix}$$

Como vimos no capítulo 2, cada valor é dado por:

$$Cov(X, X') = \sum_{i=1}^N (x_i - \mu_x)(x'_i - \mu_{x'})$$

A diagonal reflete a covariância de uma variável com ela mesma, a variância:

$$\begin{aligned} Cov(X, X) &= \sum_{i=1}^N (x_i - \mu_x)(x_i - \mu_x) \\ &= \sum_{i=1}^N (x_i - \mu_x)^2 \\ &= Var(X) \end{aligned}$$

Por exemplo, a matriz de covariâncias para o *iris*:

```
> cov(iris[, 1:4])
   Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length  0.6856935 -0.0424340  1.2743154  0.5162707
Sepal.Width   -0.0424340  0.1899794 -0.3296564 -0.1216394
Petal.Length  1.2743154 -0.3296564  3.1162779  1.2956094
Petal.Width   0.5162707 -0.1216394  1.2956094  0.5810063
> var(iris[, 1])
[1] 0.6856935
```

Usando notação matricial, seja X uma matriz com $m = 4$ colunas de $n = 150$ observações, a matriz de covariância Cov_{4x4} é:

$$Cov(X') = X'^T X' \frac{1}{n} = X'^T X' \frac{1}{150}$$

X' é a matriz cujos valores foram centralizados pela média $x' = x - \mu$. Assim o produto de X pela transposta retorna em cada elemento x_{ij} o valor $\sum_i^n (x_i - \mu_i)(x_j - \mu_j)$. Fácil implementar manualmente:

```
> iris2$Sepal.Length <- iris$Sepal.Length - mean(iris$Sepal.Length)
> iris2$Sepal.Width <- iris$Sepal.Width - mean(iris$Sepal.Width)
> iris2$Petal.Length <- iris$Petal.Length - mean(iris$Petal.Length)
> iris2$Petal.Width <- iris$Petal.Width - mean(iris$Petal.Width)
> (t(as.matrix(iris2[, 1:4]))) %*% as.matrix(iris2[, 1:4])*1/150
   Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length  0.68112222 -0.04215111  1.2658200  0.5128289
Sepal.Width   -0.04215111  0.18871289 -0.3274587 -0.1208284
```

Petal.Length	1.26582000	-0.32745867	3.0955027	1.2869720
Petal.Width	0.51282889	-0.12082844	1.2869720	0.5771329

Com base nos princípios delineados, a solução desejada por nós é tal que:

1. A covariância entre medidas é explicada por combinações de variáveis latentes compartilhadas.
2. Os dados serão explicados por uma matriz de rank mais baixa. Em nosso caso: $\Lambda_{[n \times m]}, m < 4$.
3. Para cada observação, teremos um valor de score latente F_i para cada fator. O valor final de um item é dado pela contribuição individual de cada fator mais uma variância individual. Como vimos:

$$x_{1,i} = \sum_1 nF_i\lambda_n + \epsilon$$

4. Cada fator possui uma variância intrínseca, a qual estimaremos somando uma matriz diagonal ψ à nossa matriz de pesos.

Estimamos os parâmetros para maximizar as probabilidades (*Max. Likelihood*) dos valores observados em X dadas as equações.

$$L(X^T X \frac{1}{n} | \Lambda, \psi)$$

Determinamos a função de custo conhecendo Λ e ψ :

$$C \sim \Lambda \Lambda^T + \psi$$

Em que ψ é uma matriz diagonal com mesmo rank que Λ . Como vimos anteriormente, a diagonal contém as variâncias, então os parâmetros em ψ regulam a porção de variância dos items governadas por fatores λ . Dizemos que a diagonal em $\Lambda \Lambda^T$ contém as **communalities** (variância intrínseca).

O processo de otimização para minimizar erros é mais complexo que o da regressão linear. Os estimadores possíveis aqui são muitos, nenhum deles com solução analítica simples ou garantia de convergência.

Semelhanças entre técnicas de redução de dimensões: EFA, PCA probabilístico, PCA, Autoencoder.

Podemos levar em consideração a solução anterior sem uma matriz diagonal ψ atrelada:

$$Cov \sim \Lambda \Lambda^T$$

Essa formulação é equivalente à análise de componente principal (Principal Component Analysis, PCA). Aqui, nossos pesos estimarão também a variância intrínseca. É um método computacionalmente barato para reduzir dimensões preservando informações. Matematicamente, a diferença entre PCA e EFA está no fato de o segundo estimar separadamente parâmetros para covariância compartilhada e variância individual. Uma técnica ‘intermediária’ pouco conhecida é o PCA probabilístico (PPCA), em que levamos em conta uma matriz diagonal mais simples.

$$Cov \sim \Lambda \Lambda^T + \sigma^2 I$$

Isto é: uma matriz identidade com ruído introduzido através de apenas um parâmetro (σ^2).

Uma curiosidade é que a diagonal acaba influindo menos com o aumento do rank das matrizes. Então, o resultado das técnicas acima converge em situações com alta dimensionalidade ($n \rightarrow \infty$). Uma discussão mais completa pode ser conferida em outro lugar (ver referências).

Em sumário:

$$PCA : Cov \sim \Lambda \Lambda^T$$

$$PPCA : Cov \sim \Lambda \Lambda^T + \sigma^2 I$$

$$EFA : Cov \sim \Lambda \Lambda^T + \psi$$

(Aqui, usamos \sim não para denominar semelhança, mas sim que maximizaremos o likelihood de Cov com uma expressão em função dos termos à direita)

Ainda, redes neurais do tipo *autoencoder* possuem formulação semelhante. Especificamente, um autoencoder com uma camada interna e certas restrições na função de ativação é idêntico ao PCA. Entretanto, podemos usar **mais** dimensões que o input, além de múltiplas camadas e funções não-lineares. Dessa maneira, incrementamos o poder do modelo gerativo, assim como ficamos mais vulneráveis a sobreajuste.

Voltaremos ao assunto quando o foco for modelos de ambiente, compressão de informação, modelos gerativos e redução de dimensões.

Número de fatores

Não tocamos em um ponto crucial: qual o número ótimo de fatores? É melhor um modelo que leve em conta *extroversão* ou um que use *sociabilidade* e *expressividade*?

Podemos explicar a covariância usando um número arbitrário de fatores latentes. A tendência é observarmos melhora nos indicadores de performance sob a pena de saturação (e.g. sobreajuste, interpretabilidade difícil). Existem procedimentos estabelecidos para balancear o poder explicativo com simplicidade do modelo.

Em geral, busca-se um número mínimo de fatores que maximize o poder de explicação. Considerando graus de liberdade(df) e erros do modelo (estatística X^2), dois índices populares são o RMSEA e o CFI. Assim como no cálculo de R^2 , o racional é dimensionar erros, porém aqui penalizamos a quantidade de parâmetros.

Outra métrica bastante utilizada é observar a influência de cada fator sobre a matriz de covariância.

Ao multiplicarmos um vetor por uma matriz, mudamos sua magnitude e sua direção.

Os vetores alinhados com a matriz (e.g. aqueles na diagonal da transformação acima) apenas mudam de tamanho após a transformação.

São os autovetores da matriz.

Uma das formas de extração de fatores é através dos eixos principais. Neste método, decomponemos a matriz original em vetores ortogonais multiplicados por escalares (autodecomposição, *eigen/spectral decompositon*): autovalores e autovetores (eixos).

Em geral, os primeiros eixos têm maior autovalores. Existem diversas heurísticas recomendando métodos para escolher números de fatores pelo tamanho dos autovalores. Uma delas é considerar apenas autovalores maiores que 1. Outra é considerar o ponto da curva em que há um aparente ponto de descontinuação (“joelho”).

É razoável pensar que autovetores associados a autovalores altos capturam muita informação sobre variância (individual e compartilhada) dos items.

Análise fatorial confirmatória

Os processos descritos acima são exploratórios por natureza. Buscamos o melhor ajuste para fatores latentes sem antes determinar uma estrutura. É um bom procedimento para redução de dimensões e compressão de informação, porém, se desejamos interpretabilidade e validade científica, há alguns pontos sensíveis.

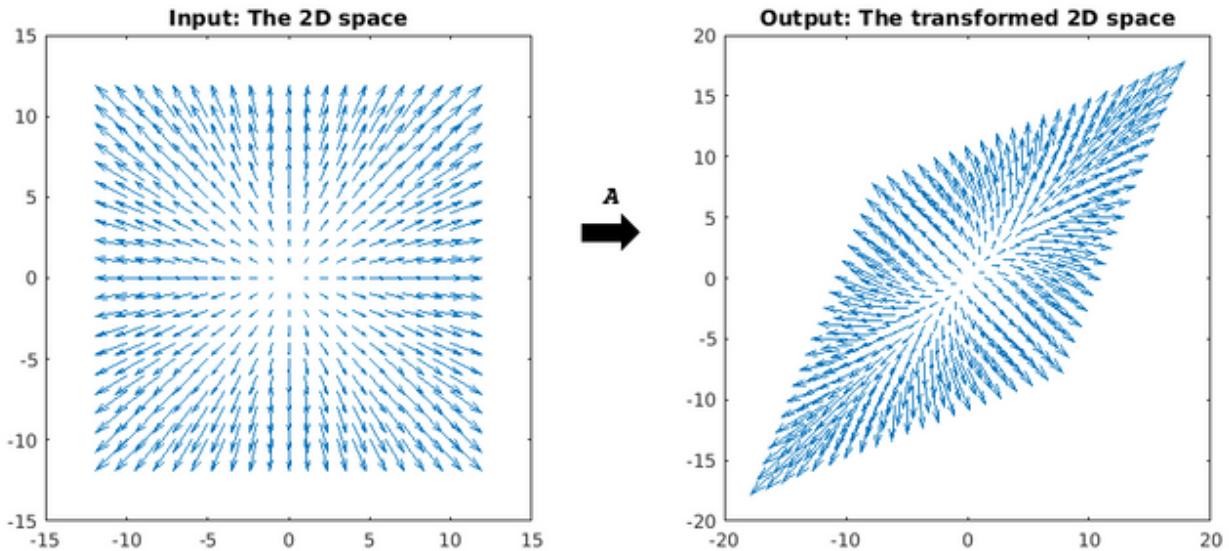
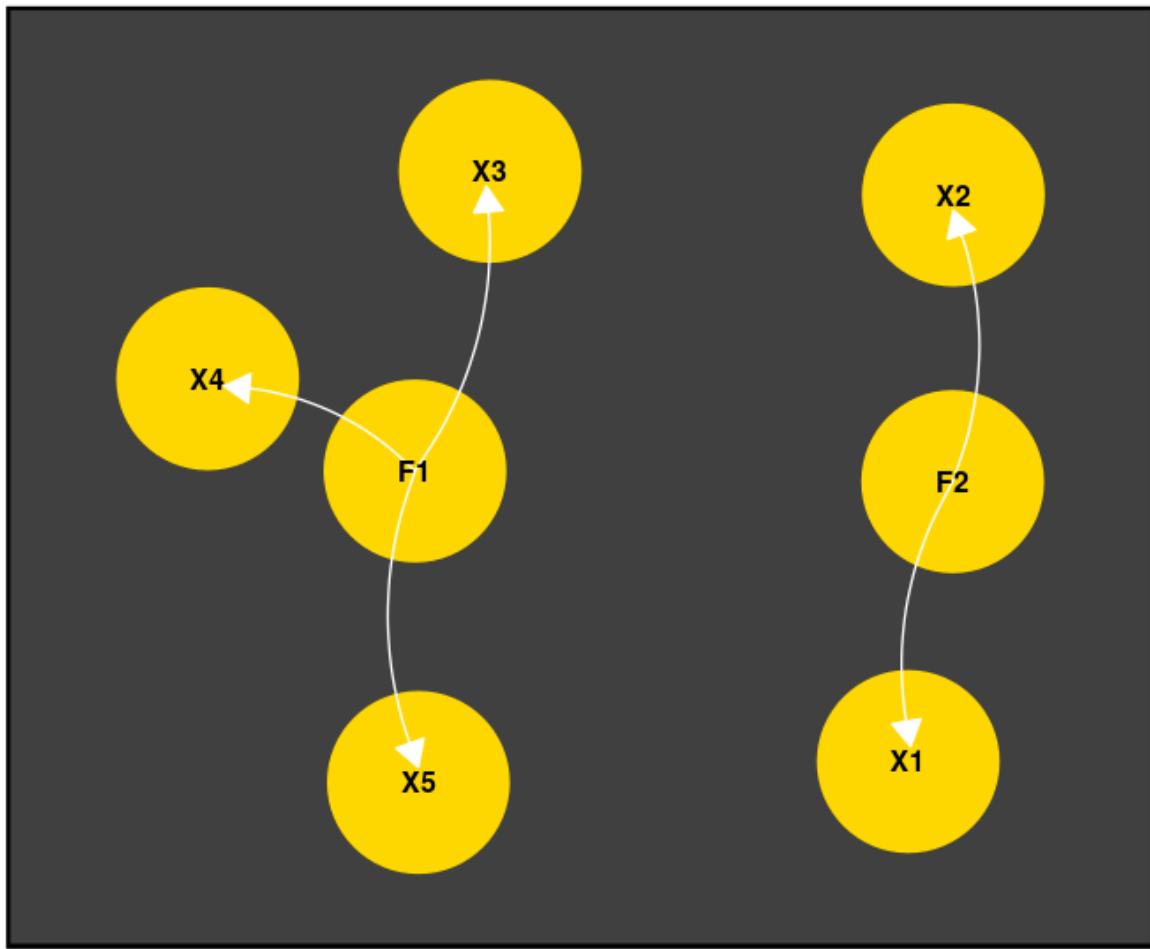


Figure 14: Efeito de multiplicação entre vetores e uma matriz A

Pensando na elaboração de uma escala para medir um traço de personalidade, retomamos o argumento de Popper (capítulo 2) contra o indutivismo. É desejável que tenhamos um modelo prévio e hipóteses testáveis de antemão. Do contrário, é fácil encontrar um modelo oferecendo bom ajuste em quase qualquer caso.

Na análise factorial confirmatória, fazemos uma restrição direta ao modelo. Os parâmetros são pré-determinados com base em diagrama (grafo) expresso por quem conduz a análise. Assim, podemos especificar uma relações:



No diagrama acima, o primeiro fator latente possui cargas nas relações com items X_3, X_4, X_5 e o segundo fator com X_1, X_2 .

Nesse caso, os estimadores serão um pouco mais complexos.

Equações estruturais

Equações estruturais são o *framework* abrangendo quaisquer dos modelos anteriores, incluindo topologias de grafos e relações arbitrárias (e.g. não paramétrica/probabilísticas).

Assim, podemos desenhar um diagrama de relações entre entidades, declarar relações entre medidas e testar adequação do modelo. Como vimos, Judea Pearl costurou esses métodos quantitativos com uma base filosófica coerente, fazendo uso dos conceitos de contrafactual e testagem de hipóteses.

Esses modelos são úteis em muitos campos para descrever estatisticamente relações de elementos múltiplos num sistemas complexo. Como sempre, devemos ter cuidado com a flexibilidade do modelo. Em especial, alguns procedimentos recomendados são de difícil conciliação com uma base hipotético-dedutiva (e.g. mudança ad-hoc do modelo após observação de índices de modificação).

Aplicações

Os grande cinco (Big Five) traços de personalidade são construídos consistentemente encontrados na busca

por fatores latentes. Eles são: agradabilidade, neuroticismo, abertura a experiências, conscienciosidade, extroversão.

Usaremos dados do <https://openpsychometrics.org/>. O dataset BIG5 tem dados demográficos (idade, gênero, país) e 50 medições em itens do International Personality Item Pool. O tamanho amostral é de 19,719. Faremos análise factorial exploratória e confirmatória através dos pacotes **psych**, **sem** e **lavaan**.

```
>system("wget http://openpsychometrics.org/_rawdata/BIG5.zip")
(...)

Resolving openpsychometrics.org (openpsychometrics.org)... 69.164.197.103
Connecting to openpsychometrics.org (openpsychometrics.org)|69.164.197.103|:80... connected.
(...)

Saving to: 'BIG5.zip'
(...)

2019-02-04 09:09:39 (624 KB/s) - 'BIG5.zip' saved [523351/523351]
> system("unzip BIG5.zip")
Archive:  BIG5.zip
inflating: BIG5/codebook.txt
inflating: BIG5/data.csv

>library(psych)
>library(lavaan)
>library(sem)
>bigf_data <- read.csv("BIG5/data.csv",sep = "\t")
>names(bigf_data)
[1] "race"      "age"       "engnat"    "gender"    "hand"
[6] "source"    "country"   "E1"        "E2"        "E3"
[11] "E4"        "E5"        "E6"        "E7"        "E8"
[16] "E9"        "E10"       "N1"        "N2"        "N3"
[21] "N4"        "N5"        "N6"        "N7"        "N8"
[26] "N9"        "N10"       "A1"        "A2"        "A3"
[31] "A4"        "A5"        "A6"        "A7"        "A8"
[36] "A9"        "A10"       "C1"        "C2"        "C3"
[41] "C4"        "C5"        "C6"        "C7"        "C8"
[46] "C9"        "C10"       "O1"        "O2"        "O3"
[51] "O4"        "O5"        "O6"        "O7"        "O8"
[56] "O9"        "O10"
```

Vamos verificar o que acontece se ajustarmos um modelo com 5 fatores latentes:

```
>library(lavaan)
>library(psych)
>efa_big <- fa(bigf_data[,8:57],nfactors = 5)
>efa_big
(...)

RMSEA index =  0.055  and the 90 % confidence intervals are  0.054 0.055
```

Observamos um valor baixo de RMSEA, o que indica baixa magnitude de erros por grau de liberdade. É interessante notar que não termos indicamos quais items avaliam quais fatores (e.g. Items O1 e O2 estão atrelados à abertura a experiência). Se as premissas estiverem corretas, para cada item, a solução encontrada deve indicar alta carga em um fator e baixa em outros.

É o que se verificar. Selecionando as estimativas para três items de três grupos. O fator com maior carga está marcado com um asterisco.

```
(...)
Factor Analysis using method = minres
```

```

Call: fa(r = bigf_data[, 8:57], nfactors = 5)
Standardized loadings (pattern matrix) based upon correlation matrix
      MR1    MR2    MR3    MR5    MR4    h2    u2 com
E1   0.69*   0.04 -0.03 -0.01  0.00  0.46  0.54 1.0
E2  -0.70*  -0.08 -0.04  0.04  0.00  0.48  0.52 1.0
E3   0.63*  -0.17  0.16  0.09 -0.06  0.57  0.43 1.3
(...)
N1  -0.06  0.69*   0.10  0.05 -0.05  0.49  0.51 1.1
N2   0.07 -0.50*  -0.01 -0.09  0.05  0.26  0.74 1.1
N3  -0.12  0.61*   0.20  0.10  0.01  0.43  0.57 1.3
(...)
A1   0.05  0.09 -0.44*   0.02 -0.07  0.20  0.80 1.2
A2   0.28 -0.04  0.50*  -0.05  0.06  0.41  0.59 1.6
A3   0.17  0.27 -0.41*  -0.15  0.10  0.27  0.73 2.6
(...)

```

Extraindo a solução:

```

>efa_bigst <- structure.sem(efa_big)
>efa_bigst
  Path      Parameter Value
[1,] "MR1->E1"    "F1E1"     NA
[2,] "MR1->E2"    "F1E2"     NA
[3,] "MR1->E3"    "F1E3"     NA
[4,] "MR1->E4"    "F1E4"     NA

```

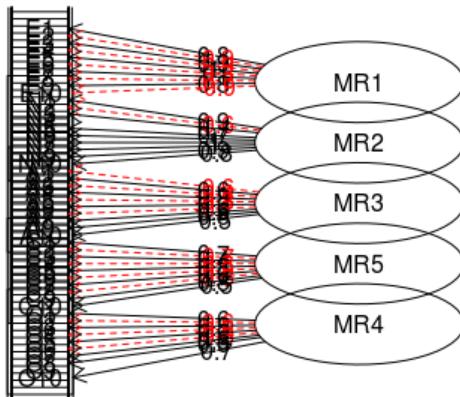
Temos os fatores (MR) e os nodos aos quais eles estão conectados, descartando aqueles de menor magnitude/significância. Podemos ajustar um modelo confirmatório a partir dessas especificações:

```

>big_sem <- sem(efa_bigst,S = cov(bigf_data[,8:57]),N = 19719)
>summary(big_sem)
(...)
>sem.diagram(big_sem,main = "Big Five",e.size=0.05)

```

Big Five



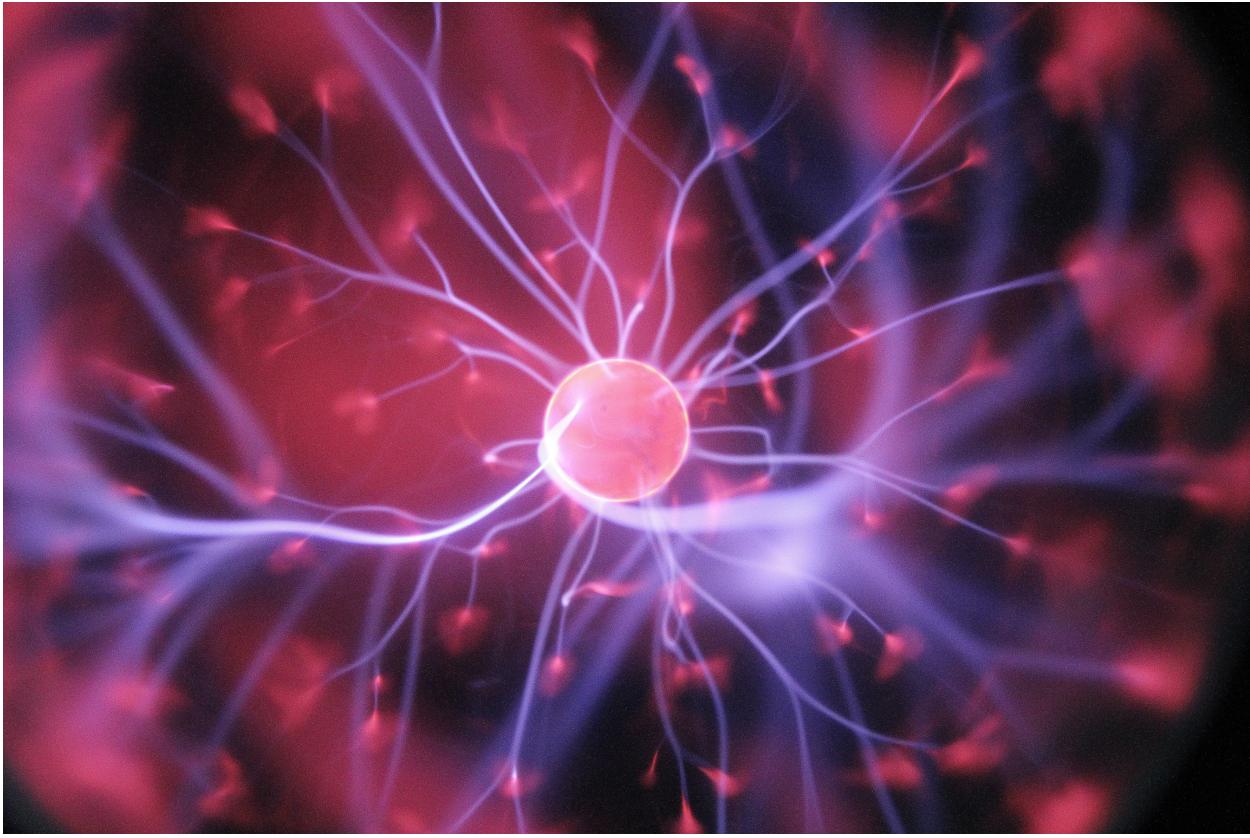
O pacote lavaan permite especificar uma família maior de modelos e é bastante popular para SEM em R. A sintaxe é:

```
>model <- c('
  F1 =~ X1 + X2 + X3
  F2 =~ Y1 + Y2 + Y3')
>lavaan (model,data,...)
```

Referências

Wright, S. (1921). "Correlation and causation". J. Agricultural Research. 20: 557–585.

<https://stats.stackexchange.com/questions/123063/is-there-any-good-reason-to-use-pca-instead-of-efa-also-can-pca-be-a-subst>
<https://steemit.com/steemstem/@dexterdev/linear-transformations-a-20-sbd-coding-contest-annoucement>



Capítulo 4 : Neurônios

Em março de 2016, o software AlphaGo venceu um mestre de Go. O feito é impressionante por se tratar de um jogo difícil de computar.

Inventado há mais de 2,500 anos, motivou avanços em matemática. Existem $2,08 * 10^{170}$ maneiras válidas de dispor as peças no tabuleiro. O polímata chinês Shen Kuo (1031–1095) chegou a um resultado próximo 10^{172} séculos atrás. Vale lembrar que o número de átomos no universo observável é de módicos 10^{80} .

No capítulo anterior, aprendemos uma formulação básica de modelo preditivo, a regressão linear simples. A seguir, estenderemos nosso leque de ferramentas para novas classes de relações, também incluindo mais informações na entrada de nossos modelos.

Mais do que isso, conheceremos a primeira máquina inteligente da história.

O perceptron de Rosenblatt

Frank Rosenblatt (1928 - 1971) nasceu e morreu em 11 de julho, mas esse não é o fato mais curioso da biografia deste psicólogo. Foi o responsável pelo desenvolvimento do primeiro neurônio artificial. Em suas palavras, o primeiro objeto não biológico a recriar uma organização do ambiente externo com significado.

It can tell the difference between a cat and a dog, although it wouldn't be able to tell whether the dog was to the left or right of the cat. Right now it is of no practical use, Dr. Rosenblatt conceded, but he said that one day it might be useful to send one into outer space to take in impressions for us. - New Yorker, December, 1958²²

O aparato reproduzia o entendimento da época sobre o funcionamento de um neurônio. O corpo recebe sinais de dendritos e, após processamentos ocultos, produz um output na forma de sinal elétrico pelo axônio. A primeira matematização viria do modelo de McCulloch & Pitts (“A Logical Calculus of the Ideas Immanent in Nervous Activity”, 1943).

Em 1949, Donald Hebb descreveu em seu clássico *The Organization of Behavior* um mecanismo plausível para a aprendizagem. Comumente expressa na máxima “Cells that fire together wire together” (células que disparam juntas, conectam-se entre si).

Com o objetivo de criar uma máquina que pudesse processar inputs diretamente do ambiente físico (e.g. luz e som), Rosenblatt concebeu extensão elegante do modelo em 1957 (“The Perceptron[do latim, *percipio, compreender*]—a perceiving and recognizing automaton. Report 85-460-1, Cornell Aeronautical Laboratory”). Composto de três partes: o sistema S (sensório); o sistema A (associação) e o sistema R (resposta). O neurônio “lógico” criado de McCulloch & Pitts foi modificado de maneira a processar inputs através de pesos antes da saída. A aprendizagem se dá pela modificação desses pesos.

Inicialmente, o perceptron foi simulado em um IBM 704 (também berço das linguagens FORTRAN e LISP). Em seguida, implementado como um dispositivo físico, batizado de Mark I Perceptron.²³ Um estudo mais profundo foi publicado por ele em 1962 (*Principles of neurodynamics*)

²²Ele consegue diferenciar um gato de um cachorro, ainda que não seja capaz de dizer se o cachorro estava à esquerda ou à direita do gato. No momento, não tem uso prático, Dr. Rosenblatt admitiu, porém disse que um dia pode ser útil para enviar um [aparato] ao espaço para capturar impressões para nós.

²³Mark I é um título comumente utilizado para a primeira versão de uma máquina.

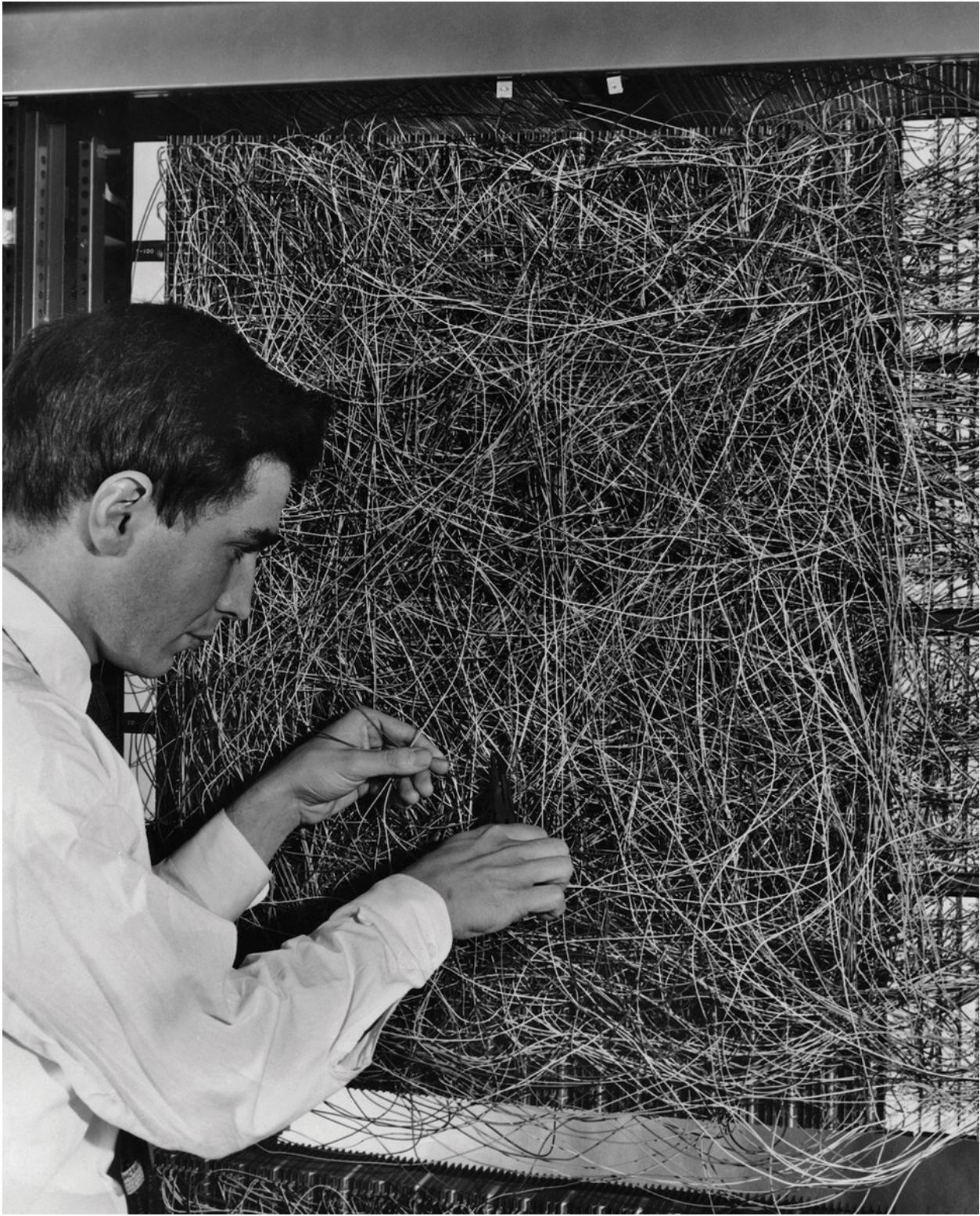


Figure 15: Frank Rosenblatt e Mark I.

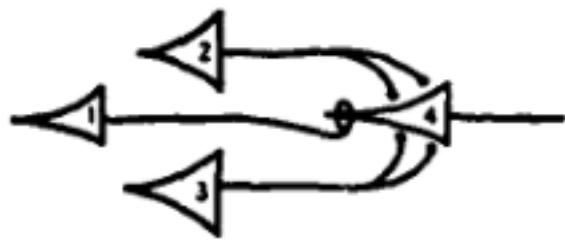


Figure 16: Diagrama de células lógicas em McCulloch & Pitts

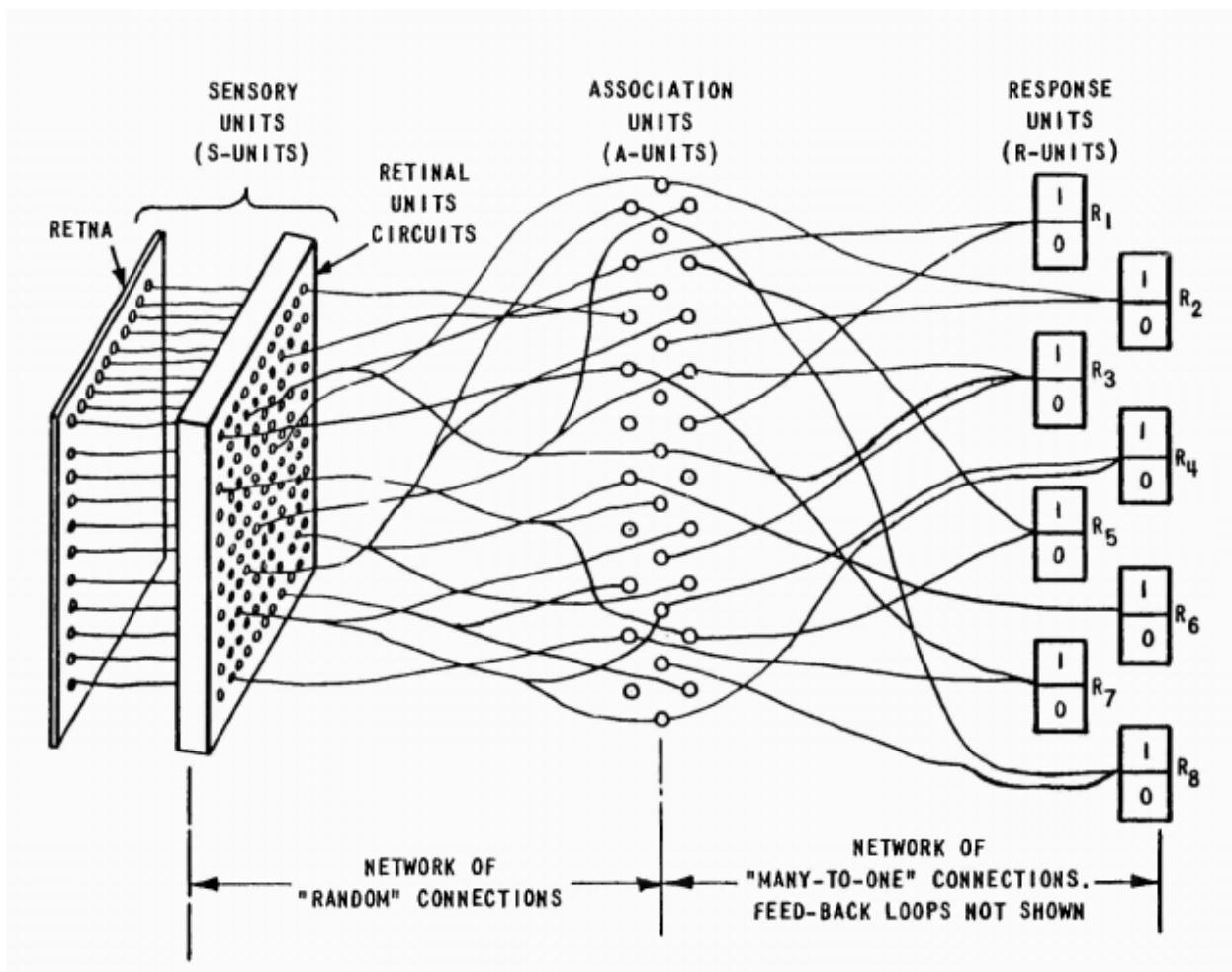


Figure I ORGANIZATION OF THE MARK I PERCEPTRON

Figure 17: Organização do Mark I, retirado de seu manual de uso original

Rosenblatt protagonizava calorosos debates sobre inteligência artificial na comunidade científica junto a Marvin Minsky, um amigo da adolescência. Em 1969, Minsky e um matemático (Seymour Papert) publicaram um livro centrado no Perceptron. (*Perceptrons: An Introduction to Computational Geometry*). Nele, provaram que o neurônio artificial era incapaz de resolver problemas não-lineares do tipo XOR. Para um problema eXclusive OR (OU eXclusivo) o neurônio deve disparar diante do estímulo A ou do estímulo B, porém não diante de ambos.

O impacto foi devastador sobre o otimismo vigente e se passou um período de 10 anos de baixíssima produção, conhecido como idade das trevas do conexionismo. A retomada dos neurônios artificiais aconteceu somente na década de 80. Infelizmente, Rosenblatt morreu prematuramente em 1972 num acidente de barco, não presenciando o renascimento dos perceptrons.

Sabendo das origens do modelo, é curioso que a maioria dos cursos introduzam perceptrons do ponto de vista puramente matemático, apontando a semelhança com neurônios como mera curiosidade. Pelo contrário, a inspiração em neurônios biológicos e posterior sucesso nas tarefas designadas fala em favor de um fantástico caso de sucesso para engenharia reversa.

A Natureza, através de evolução por seleção natural, é a verdadeira mãe desse algoritmo.

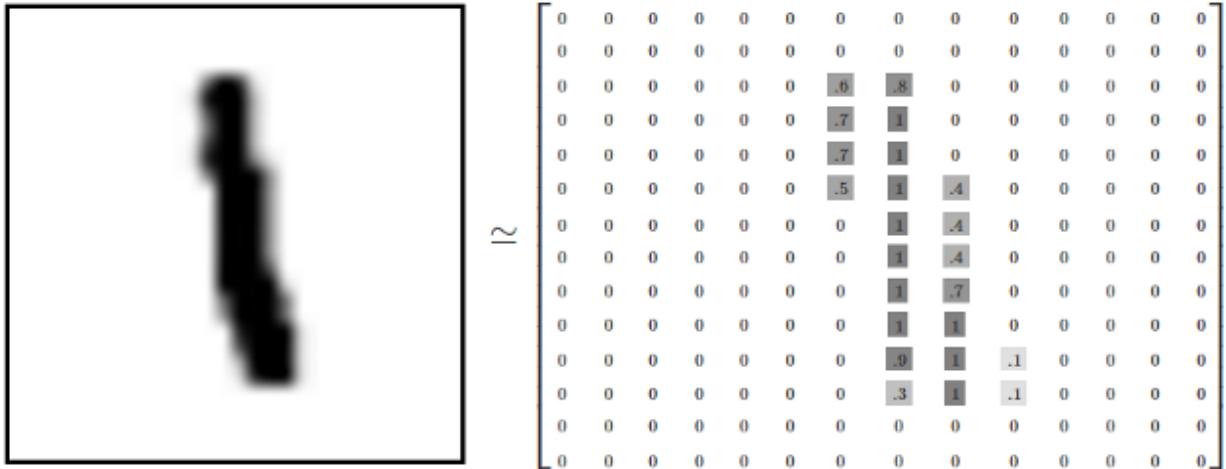
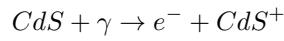


Figure 18: Exemplo de “1” em letra cursiva e sua representação numa matriz 2x2. <http://colah.github.io/posts/2014-10-Visualizing-MNIST/>

Criando neurônios

Mark I foi criado para reconhecimento visual, podendo ser considerado avô da visão computacional. Possuía um campo de entrada fotossensível de 20x20 (400) células de Sulfeto de Cádmio, as unidades S. Ao reagir com a luz, CdS emite um elétron:



Caso a célula seja ativada, envia o sinal eletrônico a uma unidade intermediária A. A unidade intermediária, por sua vez, transmite um sinal eletrônico à saída. A intensidade do sinal é regulada por sucessos prévios. Esse processo ficará mais claro com a implementação a seguir.

Imaginemos que a imagem acima tenha 10 pixels de altura e 10 de largura.

Para simplificação, 10 x 10 pixels em preto e branco (100 pixels com valores entre 0, preto, e 255, branco). Esses pixels podem ser esticados e vistos como uma matriz x de dimensão [100x1] com valores entre 0 e 255 em cada elemento.

Vamos simular uma imagem deste tamanho gerando uma matriz de dimensão 10x10 com 100 valores naturais aleatórios (entre 0 e 255) no R:

```
>set.seed(2600)
>my.image.data <- sample(0:255, 100, replace=T)
>x <- matrix(my.image.data, 10, 10)
```

	1	2	3	4	5	6	7	8	9	10
1	66	43	199	168	79	33	108	17	51	161
2	11	187	252	8	190	247	69	112	165	129
3	146	97	26	203	112	143	20	134	240	56
4	60	128	185	119	222	107	39	250	230	85
5	38	112	141	49	201	210	46	59	35	59
6	71	143	54	15	190	94	99	33	222	35
7	17	35	146	204	14	15	201	236	172	17
8	130	64	98	42	169	5	73	24	240	32
9	232	40	60	140	46	210	163	199	116	100
10	193	62	106	8	3	152	233	170	74	153

Eis a nossa imagem [10x10]. O computador lê os valores entre 0 (preto) e 255 (branco), dispondo para nós o sinal visual correspondente.

Em nosso exemplo hipotético, o classificador precisa saber se uma imagem apresentada corresponde à de um navio ou não.

Em regressão linear multipla, calculamos um peso β para cada variável. O racional aqui é parecido: ponderamos cada pixel por seus respectivos pesos. Em analogia, cada imagem é uma observação de 100 variáveis.

O neurônio deve disparar (output $y = 1$) caso seja um navio ou permanecer em repouso ($y = -1$) caso não seja.

Matematicamente, é uma multiplicação da matriz de valores da imagem x_j , de dimensão $[100 \times 1]$ por uma matriz $W_{[100 \times 1]}$ que traz os pesos (weights) estimados para cada pixel para cada classe. Então, forçamos o resultado para +1 ou -1 com uma função de ativação (ϕ).

$$y = \phi(W^T X)$$

Usaremos a função *Heaviside step*:

$$\phi(x) = \begin{cases} +1 & \text{se } x \geq 0 \\ -1 & \text{se } x < 0 \end{cases}$$

Em R:

```
library(magrittr)
# Heaviside
>phi_heavi <- function(x){ifelse(x >= 0, 1, -1)}
# Iniciando pesos com base em distribuição normal
>my_weights <- rnorm(100)/100
>w <- matrix(my_weights, 100, 1)
# Multiplicação usando o operador %*%
>as.vector(x) %*% w
# Score
[,1]
[1,] 20.19958
# Função de ativação usando %>% para encadeamento
>as.vector(x) %*% w %>% phi_heavi
```

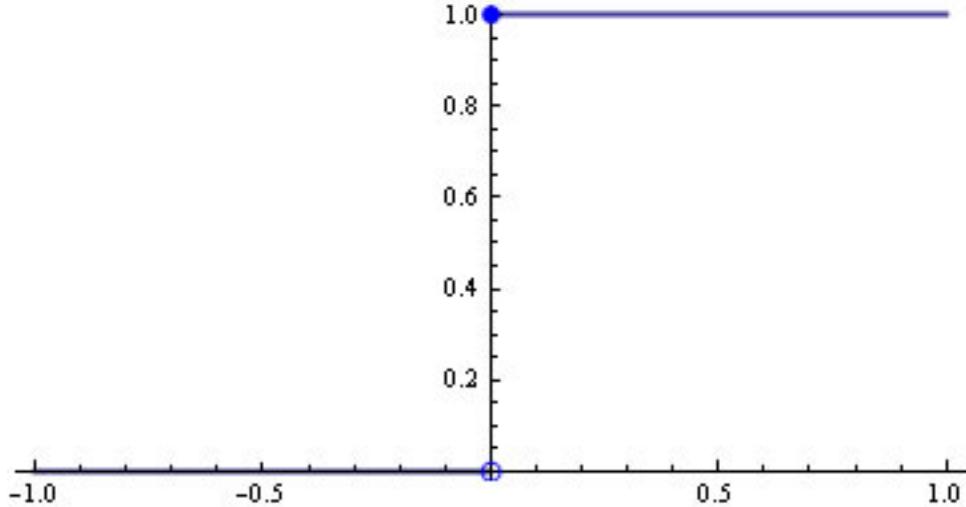


Figure 19: Heaviside step function

```
[,1]
[1,] 1
```

Para o exemplo acima, nosso neurônio com pesos aleatórios foi ativado para o estímulo aleatório x . Inicialmente, estabelecemos pesos aleatórios a partir de uma distribuição normal.

Então, o objetivo é observar as respostas corretas em várias imagens x_i e alterar os valores de W para que os scores maiores sejam os das classes corretas.

O processo de treino é bastante simples:

Seja $x_{i,j}$ o i -ésimo pixel da observação j . E w_0 o peso correspondente inicial, o peso atualizado, w' é:

$$w' = w_0 + \Delta w$$

Em que Δw indica o magnitude e o sentido da modicação no peso.

ACEITEMOS, por enquanto, a fórmula:

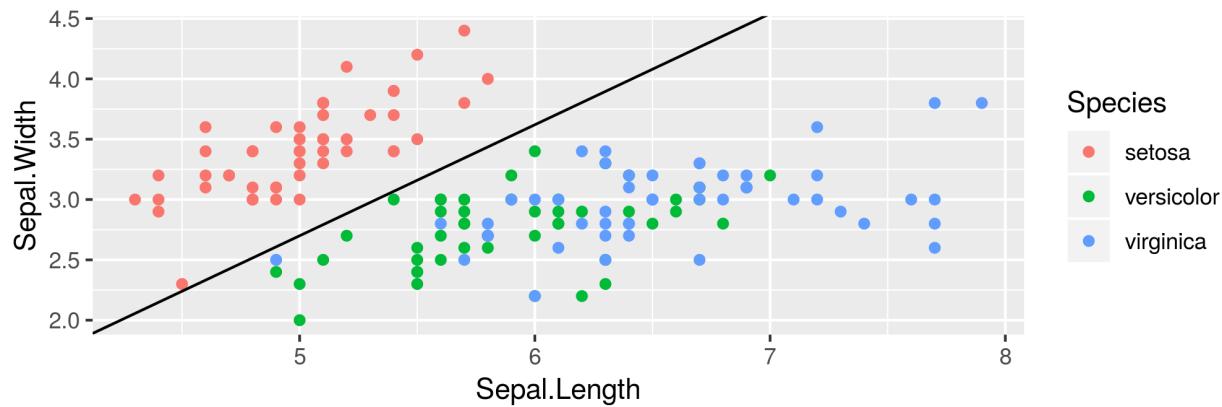
$$\Delta w_i = \eta(score_j - output_j)x_i$$

Em que $x_{i,j}$ é o valor do i -ésimo pixel, w_i é o i -ésimo peso e η uma constante chamada *tasa de aprendizagem* (learning rate), que determina o tamanho dos incrementos feitos pelo algoritmo. Mostraremos a derivação dessa equação a seguir.

Se os dados são linearmente separáveis, o algoritmo converge com um número suficiente de exemplos.

Assim, funciona para separar flores *setosa* de outra classe, mas não teríamos bons resultados separando *virginica* de *versicolor*.

```
>ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width,color=Species))+
  geom_point()+
  geom_abline(slope = 0.92,intercept = -1.9)
```



Auto MaRk I

Usando as abstrações acima, codificamos nosso perceptron em R, o Auto MaRk I.

Argumentos: Exemplos (x, vetor de números reais) e estados esperados (y, disparar = 1 vs. não disparar = -1) devem ter mesmo tamanho.

Eta: Número especificando constante de aprendizagem.

Auto MaRk I inicializa um peso aleatório para cada entrada e, numa ordem aleatória, percorre os exemplos atualizando os pesos.

```
library(magrittr)
>mark_i <- function(x, y, eta) {
  # inicializa pesos randomicos de distribuicao normal
  w <- rnorm(dim(x)[2]) # numero de pesos = numero de colunas em x
  ypreds <- rep(0, dim(x)[1]) # inicializa predicoes em 0
  # Processa as observacoes em x de forma aleatoria
  for (i in sample(1:length(y), replace=F)) {
    # predicao
    ypred <- sum(w * as.numeric(x[i, ])) %>% phi_heavi
    # update em w
    delta_w <- eta * (y[i] - ypred) * as.numeric(x[i, ])
    #nota: x[i,] sera multiplicado como matriz (dot product)
    w <- w + delta_w
    ypreds[i] <- ypred # salva predicao atual
  }
  print(paste("Weights: ",w))
  return(ypreds)
}
```

Vamos testá-lo para o problema proposto, separando flores *setosa* de *versicolor*. Preparação de dados:

```
>train_df <- iris[1:100, c(1, 2, 5)]
>train_df[, 4] <- -1
>train_df[train_df[, 3] == "setosa", 4] <- 1
>names(train_df) <- c("s.len", "s.wid", "species", "target")
>head(train_df)
  s.len s.wid species target
1  5.1   3.5   setosa     1
2  4.9   3.0   setosa     1
3  4.7   3.2   setosa     1
4  4.6   3.1   setosa     1
5  5.0   3.6   setosa     1
6  5.4   3.9   setosa     1
> train_df[60:65,]
  s.len s.wid   species target
60  5.2   2.7 versicolor    -1
61  5.0   2.0 versicolor    -1
62  5.9   3.0 versicolor    -1
63  6.0   2.2 versicolor    -1
64  6.1   2.9 versicolor    -1
65  5.6   2.9 versicolor    -1
>x_features <- train_df[, c(1, 2)]
>y_target <- train_df[, 4]
```

E então, podemos ativá-lo:

```

>y_preds <- mark_i(x_features, y_target, 0.002)
[1] "Weights:  0.309434266643425"
[2] "Weights: -0.426791898133975"
> table(y_preds,train_df$target)
y_preds -1  1
-1 19  1
 1 31 49
> y_preds
 [1]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
[25]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
[49]  1  1 -1  1  1 -1  1 -1 -1  1  1  1 -1  1 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
[73]  1 -1 -1  1 -1 -1  1  1 -1  1 -1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
[97] -1  1  1 -1

```

Usando $\eta = 0.002$, obtivemos 41% de acurácia (classificações corretas). Podemos modificar a taxa de aprendizagem. Com $\eta = 0.05$, aumentamos para 59%. Com $\eta = 0.1$, temos 62%. Um bom valor é 0.01, com 77%. **Nada mau!** Codificamos nosso neurônio *do zero*, usando algumas matrizes, pesos aleatórios e um algoritmo sequencial de operações e atualização de pesos. Com isso, atingimos uma acurácia razoável.

```

> y_preds <- mark_i(x_features, y_target, 0.05)
[1] "Weights: -1.12748454064396"
[2] "Weights: 1.35197455996465"
> table(y_preds,train_df$target)
y_preds -1  1
-1 30 21
 1 20 29

> y_preds <- mark_i(x_features, y_target, 0.1)
[1] "Weights: -2.08944843785222"
[2] "Weights: 3.35800090343738"
> table(y_preds,train_df$target)
y_preds -1  1
-1 36 14
 1 14 36

> y_preds <- mark_i(x_features, y_target, 0.01)
[1] "Weights: -0.250410476080629"
[2] "Weights: 0.447470183281492"
> table(y_preds,train_df$target)
y_preds -1  1
-1 43 16
 1  7 34

```

Chamamos η de hiperparâmetro. A escolha de valores para hiperparâmetros é um dos desafios em aprendizagem estatística. Uma maneira trivial é testar muitos valores possíveis e observar o desempenho, entretanto isso não é exequível para grandes volumes de dados e/ou muitos parâmetros. Existem diversos processos heurísticos e algoritmos para encontrar valores ótimos.

Uma forma popular para otimizar o treinamento é particionar o dataset em pedaços e apresentar os particionamentos (epochs) repetidas vezes ao classificador ou acumular os erros de epochs ao invés de exemplos individuais. Assim, calculamos erros agregados e evitamos mínimos locais.

Para evitar andar em círculos, avançamos por mais tempo em uma direção antes de recalcular a rota.



Deep learning



Intuições

Com o aprendizado através de exemplos, otimizamos nosso classificador (mudando pesos W) para minimizar a perda usando aproximações. Assim como estendemos modelos simples do capítulo 2 usando grafos no capítulo 3, aqui vamos aplicar o mesmo conceito e imaginar relações entre neurônios.

Os dados são apresentados aos perceptrons na linha de frente. O output, porém, não é o resultado dessa primeira operação: esse valor é usado como input para neurônios da próxima camada.

Assim, conseguimos implementar transformações adequadas (rotações, torções, escalonamentos, dobrar) em sequência, de maneira que abstrações complexas possam ser capturadas.

Going Deep

As versões reais da maioria dos conceitos criados por seres humanos não são idênticas umas às outras. Em outras palavras, não existe um conjunto rígido de regras para classificarmos a maior parte das entidades ao nosso redor. Muitas entidades são diferentes, porém similares o suficiente para pertencer a uma mesma categoria.



Todos são naturalmente reconhecidos como felinos, mas apresentam variações de tamanho, cor e proporção em todo o corpo. Esse é um problema interessante e antigo, mais conhecido na ideia de entes platônicos, os quais capturam a essência de um conceito.

Alguns filósofos contemporâneos acreditam que abstrações humanas são instâncias de um conceito mais genérico: mapas biológicos contidos em redes neuronais (Paul Churchland, Plato's Camera).

Esses mapas estão associados de forma hierarquizada. Numerosos padrões em níveis inferiores e um número menor em camadas superiores.

No caso da visão, neurônios superficiais captam pontos luminosos. O padrão de ativação sensorial enviado ao córtex visual primário é o primeiro mapa, que é torcido e filtrado caminho cima.

Neurônios intermediários possuem configurações que identificam características simples: olhos e subcomponentes da face. Por fim, temos camadas mais profundas, ligadas a abstrações.

Deduzindo superfícies

Um classificador deve capturar essa estrutura abstrata a partir de modelos matemáticos tratáveis. Para examinarmos esse aspecto, usemos um exemplo. O gráfico abaixo representa milhares de amostras com: (1) a curva diária natural de um hormônio (em vermelho) e a curva sob uso de esteroides anabolizantes (azul).

Como hipotéticos membros de uma comitê atlético, nosso objetivo aqui é, dada uma amostra, saber se o atleta está sob efeito de esteroides. Quando experimentamos, normalmente haverá ruídos (erros) na medida e receberemos medições imprecisas da curva. Variações na dieta daquele dia, micções, sudorese, stress e outros fatores.

Usamos o tempo (t , eixo horizontal) e nível hormonal (β , eixo vertical).

Um modelo bastante popular para classificações é o de regressão logística. Nele, estimamos probabilidade para um evento com base nas probabilidades de uma função sigmoide. Temos uma probabilidade (valor entre 0 e 1) definida por:

$$P(h, \beta) = \frac{1}{1 + e^{-(i+t*h+\beta*y+\epsilon)}}$$

ϵ representa o erro e i é uma constante.

Em uma linha de R:

```
>logist.fit <- glm(type_dic ~ beta + tempo, family=binomial,data=inv.ds)
```

A vantagem de usar essa modelagem é que temos uma relação direta entre o inverso dessa função (P^{-1}), “logito”) e a combinação linear dos nossos parâmetros:

$$\text{logit}(P(x)) = i + t * x + \beta * y + \epsilon$$

Em outras palavras, o processo de estimativa é parecido com o da regressão linear, que é facilmente tratável. Outra consequência é que assumimos que a distinção entre classes (com base no logito, log odds) pode ser dada por um limite. Este tem uma relação linear com nossas variáveis. Estimamos a magnitude e o sentido dessas relações pelos parâmetros da regressão.



Figure 20: Resposta a estímulos visuais em V1 de *Macaca fascicularis* <http://www.jneurosci.org/content/32/40/13971>

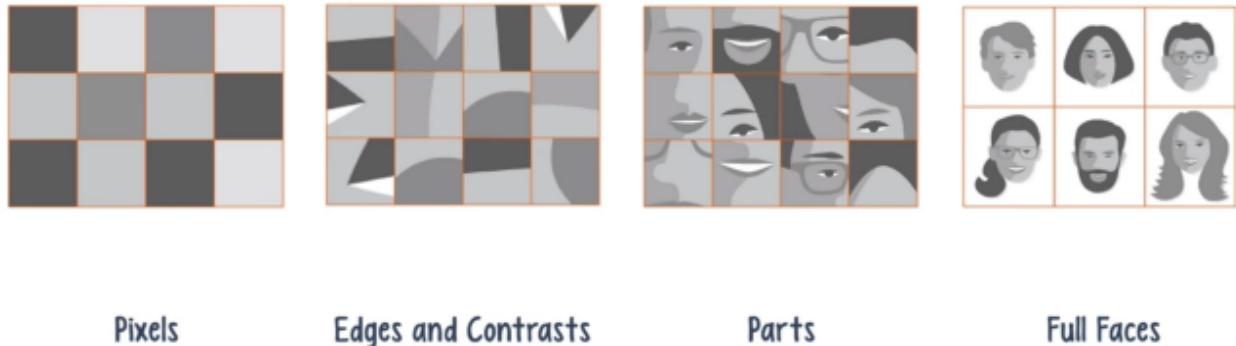
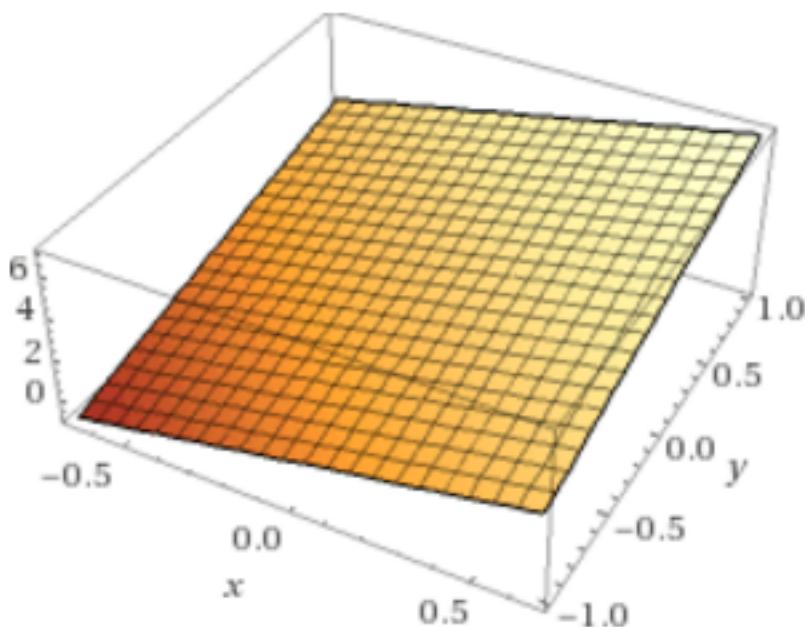


Figure 21: Retirado de: <https://www.youtube.com/watch?v=SeyIg6ArS4Y>



Podemos imaginar que o log odds (z , eixo vertical) cresce linearmente com uma combinação de duas variáveis (x e y). Notem que a superfície definida pelo nossa equação/modelo é um [hiper]plano²⁴ dado por $z = 3 + 3x + 2y$. Estimamos qual seria a posição na reta dada por aquela medida e usamos um limite de decisão (decision boundary) linear. Voltando ao nosso exemplo, seria difícil capturar as diferenças usando esta estratégia.

Acima, um neurônio sigmoide, que equivale à regressão logística. É como o plano anterior, mas visto de cima, dividimos ele em duas regiões para classificação. Por que? O classificador linear otimiza suas respostas levando em conta apenas o valor absoluto da medida hormonal. Isto é, valores acima de um limite serão considerados doping, não considerando horário. Matematicamente, o coeficiente para o tempo foi ajustado em 0. Mudar isso tornaria a reta divisória inclinada em relação ao eixo x , piorando a classificação.

Podemos verificar isso diretamente através dos parâmetros estimados em nosso modelo de regressão.

```
> summary(logist.fit)
Call: glm(formula = type_dic ~ beta + tempo, family = binomial, data = inv.ds)
Coefficients:
(Intercept)      beta      tempo
```

²⁴Um hiperplano é a generalização de plano (curvatura zero) para quaisquer dimensões. O hiperplano é um espaço de $n - 1$ em um espaço n dimensões. A reta é um hiperplano em duas dimensões, o plano é um hiperplano em 3 dimensões. Para dimensões mais altas, a visualização é menos simples. Um hiperplano divide os espaço em duas partes.

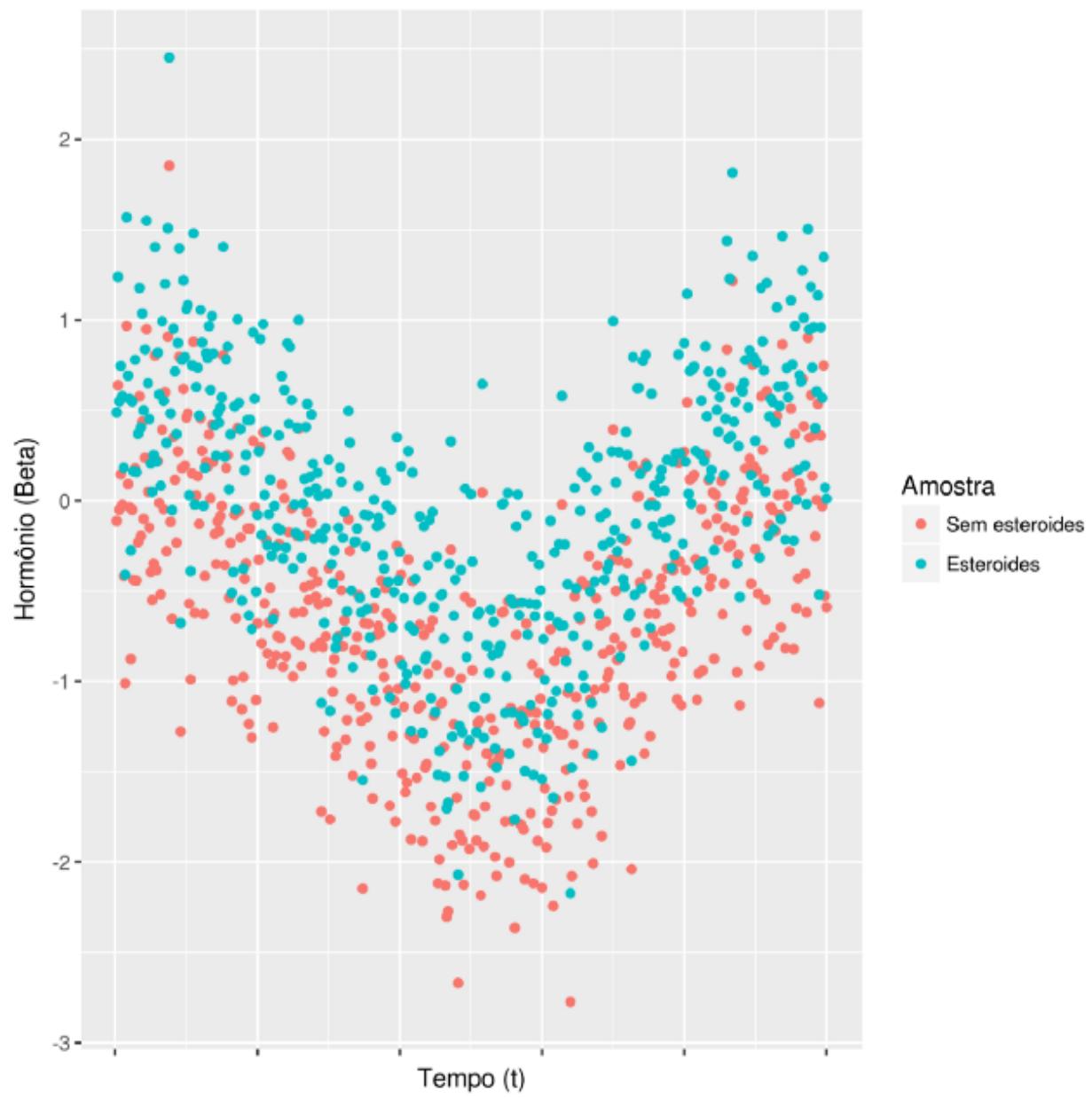


Figure 22: Exemplo inspirado no texto de Chris Olah (<http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/>)

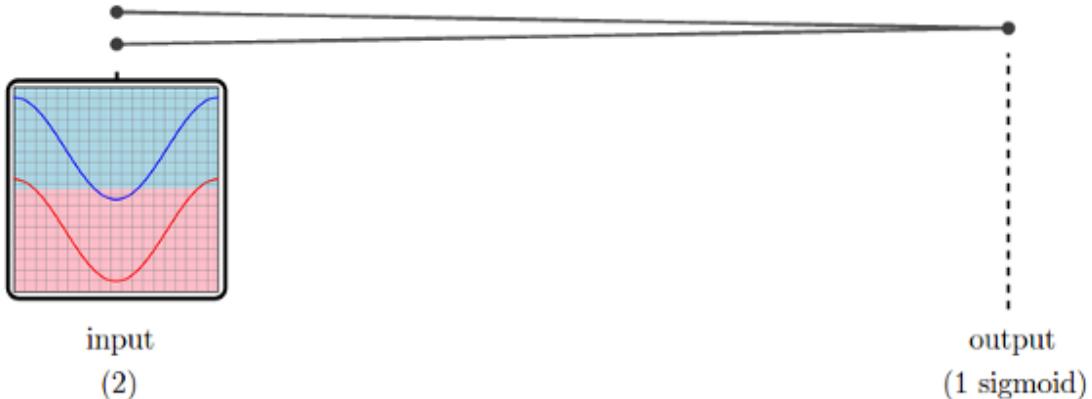


Figure 23: <http://colah.github.io/posts/2015-01-Visualizing-Representations/>

```

-0.8752803  -3.6195723  -0.0001221 # Próximo a zero
Degrees of Freedom: 999 Total (i.e. Null);  997 Residual
Null Deviance:      1386
Residual Deviance: 774.4  AIC: 780.4
> prob=predict(logist.fit,type=c("response"))
> inv.ds$prob=prob
> curve <- roc(type_dic ~ prob, data = inv.ds)
> curve

Call:
roc.formula(formula = type_dic ~ prob, data = inv.ds)
Data: prob in 500 controls (type_dic 0) < 500 cases (type_dic 1).
Area under the curve: 0.8767

```

Quem poderá nos ajudar?

A solução é introduzir termos polinomiais de grau mais alto ($x^2, x^3 \dots$), interações ou usar funções mais complexas. Aí corremos o risco de realizar sobre ajuste. Deixar o sinal dos confundir e fazer um modelo complexo que não funciona em novos exemplos.

E o que acontece se conectarmos classificadores simples hierarquicamente?

A resposta de uma unidade é usada como a entrada de outras. Quando processamos o sinal em etapas, cada camada modifica os dados para as camadas posteriores, transformando e filtrando/dando forma.

As camadas intermediárias permitem a transformação gradual do sinal, e o sistema acerta usando apenas dois classificadores simples (sigmoids). No exemplo acima, temos uma camada de 2 neurônios entre o input e o output.

Agora, a primeira camada (hidden) modifica a entrada com duas unidades sigmoids e a segunda camada pode classificar corretamente usando apenas uma reta, algo que era impossível antes. Em tese, esse modelo pode capturar melhor as características que geraram os dados (flutuação hormonal ao longo do dia).

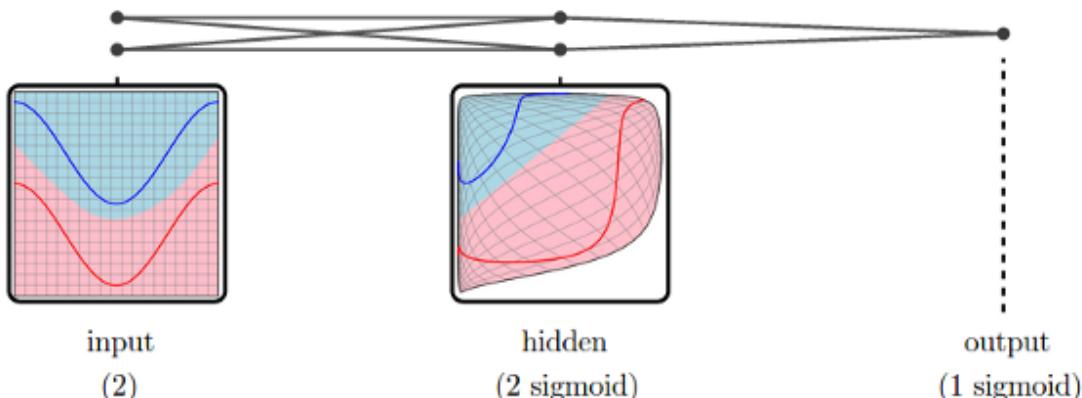


Figure 24: <http://colah.github.io/posts/2015-01-Visualizing-Representations/>

Neurônios

Notem que o diagrama acima lembra uma rede neural. Esse tipo de classificador foi inspirado na organização microscópica de neurônios reais e acredita-se que seu funcionamento seja de alguma forma análogo. A arquitetura de redes convolucionais (convolutional neural networks), estado da arte em reconhecimento de imagens, foi inspirada no córtex visual de mamíferos²⁵. Outros modelos bio inspirados (Spiking neural networks, LTSMs...) apresentam desempenhos inéditos para tarefas complexas e pouco estruturadas, como reconhecimento de voz e tradução de textos. A teoria mais aceita é de que o maquinário neural dos animais foi desenhado por processos evolutivos, como a seleção natural. Assim, apresenta coloridas formas de complexidade a depender da tarefa desempenhada.

Como podemos ver, as redes biológicas são complexas, com até dezenas de bilhões de unidades de processamento paralelas conectadas. Zona destacada possui grafo isomorfo ao descrito no texto.

Nos modelos profundos (deep) de reconhecimento de rosto, neurônios de camadas superficiais capturam bordas, ângulos e vértices, camadas intermediárias detectam presença de olhos, boca, nariz. Por fim, camadas ao final da arquitetura decidem se é um rosto ou não e a quem ele pertence.

Eficiência e aplicações

Podemos demonstrar formalmente que uma rede neural com apenas uma camada interna é capaz de aproximar qualquer função. A prova não é lá essas coisas, já que, no fundo o que fazemos é criar uma tabela de consulta (lookup table) para os valores de entrada e saída usando os neurônios. Na prática, é difícil obter boas performances. Tão difícil que as redes neurais também passaram décadas esquecidas. Se você rodar o modelo abaixo, baseado no nosso exemplo, verá que a acurácia é próxima da regressão logística. É necessário algum conhecimento e tempo para afinar os detalhes.

Normalmente, depende da qualidade e da quantidade dos dados. O boom veio com a descoberta de topologias de rede especificamente boas para certas tarefas (e.g. LSTM para linguagem natural, Conv Nets para visão computacional).

Em outras palavras, modelar uma rede neural para problemas inéditos pode ser algo desafiador.

O código a seguir mostra como implementar uma rede com a topologia descrita usando a lib **deepnet**. O modelo tem um funcionamento ligeiramente diferente (deep belief networks), porém não nos preocupemos com isso no momento.

²⁵(<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1557912/>)

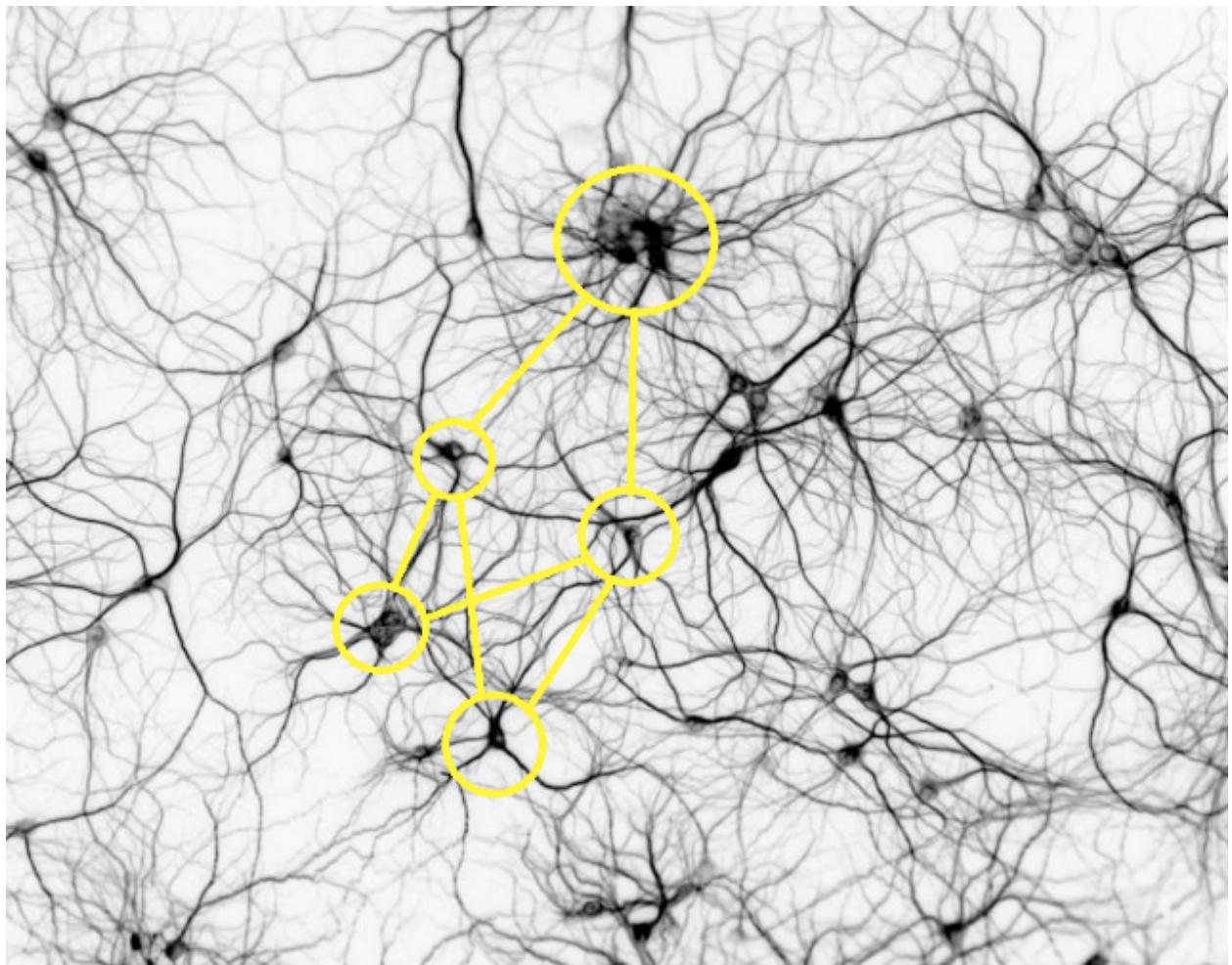


Figure 25: Modificado de <http://www.rzagabe.com/2014/11/03/an-introduction-to-artificial-neural-networks.html>

```

# Neural Net para o exemplo
# Processo para gerar dados em arquivos auxiliares ao livro(/aux)
>library(deepnet)
>inv.ds$tempo.norm <- normalize(inv.ds$tempo)
>deep.log.dbn <- dbn.dnn.train(
  x=as.matrix(inv.ds[,c("beta","tempo.norm")]),
  y=as.numeric(as.character(inv.ds$type_dic)),
  hidden = c(2), activationfun = "sigm",
  learningrate=2.65, momentum=0.85, learningrate_scale=1,
  output = "sigm", numepochs=3, batchsize= 11)
  (...)

begin to train dbn .....
training layer 1 rbm ...
dbn has been trained.

begin to train deep nn .....
deep nn has been trained.

>inv.ds$deep.test <- nn.predict(deep.log.dbn,
  x=as.matrix(inv.ds[,c("beta","tempo")]))


>curve <- roc(type_dic ~ deep.test, data=inv.ds)
>plot(curve)
>curve
Call:
roc.formula(formula = type_dic ~ deep.test, data = inv.ds)
Data: deep.test in 1000 controls (type_dic 0) < 1000 cases (type_dic 1).
Area under the curve: 0.6671

```

As redes neurais passaram algum tempo esquecidas, até que algumas reviravoltas²⁶ permitiram o treinamento eficaz dessas redes. Algoritmos para melhorar o treinamento, assim como arquiteturas econômicas ou especialmente boas em determinadas tarefas. Além disso, o uso de processadores gráficos (GPU), desenhados para as operações de álgebra linear que discutimos (com matrizes) permitiu treinar em um volume maior de dados.

Gradient Descent para o Perceptron

Até o momento, ilustramos intuições e aplicações básicas, porém o grande desafio de modelos com muitos parâmetros está em orquestrar o treinamento conjunto de diferentes nodos.

Usamos os pesos com uma fórmula contendo taxa de aprendizagem (η) e outros parâmetros: a função de erro entre score desejado($score_j$) e output $E = d(score_j, output_j)$; valor da entrada (x_i).

$$w'_i = w_i + \Delta w_i$$

Δw_i pode ser obtido usando o conceito de Gradient Descent. Intuitivamente, calculamos a inclinação local e caminhamos no sentido oposto ao mais íngreme. O valor de η governa o tamanho dos passos.

Levando em conta cada j -ésima observação, definimos uma função de perda L expressando a soma dos erros nos n exemplos e minimizamos ela.

$$\min(L) = \min \sum_j^n E(score_j, output_j))$$

²⁶(<http://people.idsia.ch/~juergen/who-invented-backpropagation.html>)

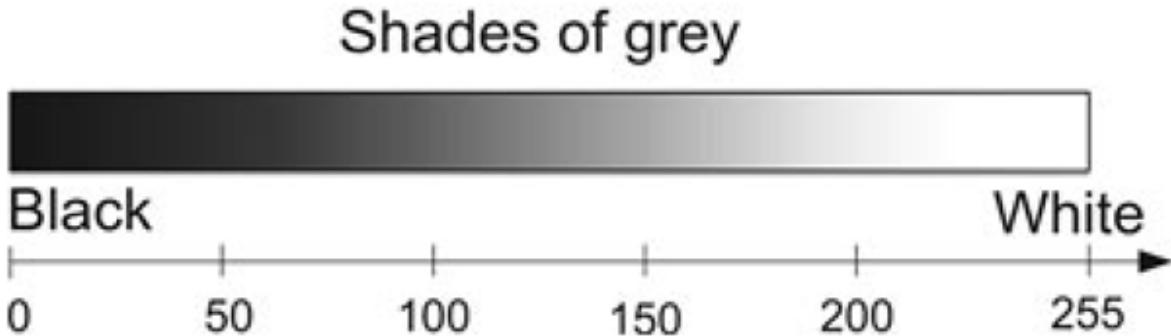
Calculamos o valor dos pesos atuais e percorremos o espaço em direção a um valor mínimo local. Se a superfície L for convexa, acharemos uma solução ótima com o número suficiente de passos.

Usaremos para nossa função de erro a distância euclidiana entre score desejado e output. O score desejado é a resposta ótima e o output é um produto entre pesos e entrada:

$$E = d_{eucl.}(score_j, output_j) = (score_j - output_j)^2$$

Notem que o processo envolve implementar uma função de erro entre resultados da rede e um espaço virtual de scores ótimos. O sucesso do treinamento depende de uma correspondência entre a função de distância escolhida e a distância real no espaço em que os dados foram gerados. Não sabemos se isso reflete a realidade. No exemplo, cada pixel reflete um sinal de 0 a 255.

A figura abaixo mostra a correspondência entre valores da medida e escala visual.



A intuição para sensibilidade à luz pode ser percebida num intervalo contínuo entre incidência total de luz (valores extremos de branco, medida: 255) e ausência total (valores extremos de preto, medida: 0). Supondo que podemos atribuir um rótulo a cada tom de cinza e que esse conjunto é ordenável pela *clareza*, dizemos que há isomorfismo de ordem entre os conjuntos.

Isso implica que a distância euclidiana deve funcionar em nossas medidas como nos números reais \mathbf{R} . Resta saber se a projeção das observações é linearmente separável. É intuitivo para seres humanos saber quais problemas serão separáveis: basta imaginar a tarefa de diferenciar tipos de imagens com uma regua numa tela em preto e branco.

Para descobrir o valor mínimo de L , vamos encontrar polos através de derivadas parciais. Ou, seu equivalente para funções de múltiplas variáveis (espaços multidimensionais), o gradiente(∇). É o produto escalar das derivadas parciais daquela função.

Para cada observação x_j , a derivada parcial da função de perda em relação a um peso w_i expressa a taxa de variação no erro global em função daquele peso. $\frac{d}{dw_i} L(w_i) = \frac{d}{dw_i} \frac{1}{n} \sum_j nE(score_j, output_j)$

Sabemos então se devemos ajustar o peso para cima ou para baixo, assim com a magnitude do passo. Algebricamente, modificaremos w seguindo o inverso do gradiente. A taxa de aprendizagem é um hiperparâmetro que regula artificialmente o tamanho desse passo:

$$\begin{aligned} \Delta w_i &= -\eta \frac{dL}{dw_i} \\ &= -\eta \frac{d}{dw_i} \frac{1}{n} \sum_j E(score_j, output_j) \end{aligned}$$

Lembrando que o erro é dado pela distância euclidiana:

$$= -\eta \frac{d}{dw_i} \frac{1}{n} \sum_j^n (score_j - output_j)^2$$

Fazemos $f(x) = (score_j - output_j)$ e $g(x) = x^2$, de maneira que

$$L = \frac{1}{n} \sum_j^n E(score_j, output_j) = (g \circ f)$$

$$= \frac{1}{n} \sum_j^n (score_j - output_j)^2$$

Podemos resolver $\frac{d}{dw_i} L$ aplicando a regra de cadeia $(g \circ f)' = (g' \circ f)f'$ e a ‘regra do tombo’ para derivadas de polinômios ($\frac{d}{dx}(x^n) = nx^{n-1}$).

Então,

$$f' = \frac{d}{dw_i} (score_j - output_j)$$

O output é dado pelo produto escalar entre pesos w_j e entradas x_j :

$$f' = \frac{d}{dw_i} (score_j - w_j \cdot x_j)$$

O score desejado não depende dos pesos, portanto a primeira derivativa é 0.

$$\begin{aligned} f' &= 0 - \frac{d}{dw_i} w_j \cdot x_j \\ &= -\frac{d}{dw_i} \sum_{i,j}^n w_{i,j} * x_{i,j} \\ &= -\frac{d}{dw_i} (w_0 * x_0 + \dots + w_i * x_i + w_n * x_n) \end{aligned}$$

Os termos não dependentes de w_i também são zerados e ficamos com o primeiro termo da soma:

$$f' = -\frac{d}{dw_i} w_i x_i$$

A função a ser derivada agora descreve uma relação linear (polinômio de grau 1) em w_i e temos:

$$f' = (-x_{i,j})$$

Sabendo f' , buscamos o outro termo em $(g \circ f)'$:

$$(g \circ f) = (score_j - output_j)^{2-1}$$

$$(g' \circ f) = 2(score_j - output_j)^{2-1}$$

$$= 2(score_j - output_j)$$

Por fim, a derivada parcial da função de perda para o i-ésimo peso w_i é:

$$\frac{dL}{dw_i} = \sum_j^n \frac{d}{dw_i} (score_j - output_j)^2$$

$$= \sum_{i,j}^n 2(score_j - w_j \cdot x_j)(-x_{i,j})$$

Para simplificar a expressão e estabelecer o tamanho dos incrementos sobre o pesos, escalamos a derivada parcial por uma constante, dada por $-\frac{1}{2}\eta_0$:

$$-\frac{1}{2} * \eta_0 \frac{dL}{dw_i} = -\frac{1}{2} \eta_0 * 2(score_j - output_j)(-x_j)$$

$$\Delta w_i = \eta_0 \sum_j^n (score_j - w \cdot x)(x_j)$$

E η_0 é um [hiper]parâmetro que simplifico a equação e define o tamanho dos incrementos usados.

Como implementamos antes no Auto MaRK I.

```
(...)
ypred <- sum(w * as.numeric(x[i, ])) %>% phi_heavi
delta_w <- eta * (y[i] - ypred) * as.numeric(x[i, ]) #-----
w <- w + delta_w
(...)
```

Backpropagation

Uma vez que o texto é sobre deep learning, precisamos falar de backpropagation. É o conceito de propagar gradientes da função perda ao longo da rede de maneira a atualizar cada nodo de maneira única.

Como vimos, podemos encarar a rede neural como uma sequência de funções plugadas. Algebraicamente, se o primeiro nodo é $q(x, y)$, o neurônio f que recebe sua saída como input tem valor $f(q(x, y))$ ou $f \circ q$.

Exemplo

Neurônio de input: $q(x, y) = 3x + 2y$

Segundo neurônio: $f(z) = z^2$

Output final: $f(q(x, y)) = q^2 = (3x + 2y)^2$

Podemos calcular o efeito de mudanças inter nodos com a regra de cadeia funções compostas. Isto é, podemos obter o gradiente de erro no nodo de hierarquia mais alta (f), com respeito a uma das variáveis de entrada (x) na hierarquia mais baixa. A operação é computacionalmente barata, bastando multiplicar as derivadas parciais dos erros em cada parte.

$$\frac{df}{dx} = \frac{df}{dq} \frac{dq}{dx}$$

É possível calcular de forma recursiva, portanto local e paralela, ao longo das camadas. Fazendo o mesmo acima para df/dy , teremos os valores de df/dx e df/dy que é precisamente nosso gradiente em f .

```
# Valor duplo (x,y) para inputs
>x <- 1
>y <- 3
q <- 3*x + 2*y # primeira camada
f <- q^2 # segunda camada
```

```

# Backprop - Mudanças em hierarquia superior
# dadas por entradas de camadas inferiores
dfdq <- 2*q # derivada de  $x^2$ ; variação de  $f$  em função de  $q$ 
dqdx <- 3 # Derivada de  $3x$ ; variação de  $q$  em função de  $x$ 
dqdy <- 2 # Derivada de  $2x$ ; variação de  $q$  em função de  $y$ 
# Obter gradiente de  $f(x,y)$  multiplicando as parciais
dfdx = dfdq*dqdx
dfdy = dfdq*dqdy
grad = c(dfdx,dfdy)
> grad
[1] 24 16

```

Usando essa lógica, calculamos os gradientes para a função de erro e treinamos o modelo.

Podemos então implementar nossa rede neural, Mark II.

Mark II

Nossa rede terá um perceptron de entrada com dimensão igual à do input. Entretanto, acrescentamos um peso a mais, que corresponderá a um intercepto.

Note que

$$y = w_0 + w_1x_1 + w_2x_2$$

é o mesmo que

$$y = 1 * w_0 + w_1x_1 + w_2x_2$$

Assim, adicionamos um peso w_0 e também forçamos uma dimensão a mais no input, que sempre terá valor 1. Chamamos esse artifício de adicionar um intercepto (*bias*) de *bias trick*. Ajuda a estabelecer um valor basal para o output, facilitando a convergência.

```

library(magrittr)
library(ggplot2)
set.seed(2600)

mark_ii <- function(x, y, eta, reps=1) {

  # inicializa pesos randomicos de distribuicao normal
  w1 <- rnorm(n = (dim(x)[2]+1)) %>% as.matrix # numero de pesos = numero de colunas em x + bias

```

Em seguida, neurônios da camada intermediária, dois, cada um com dois pesos.

```

w21 <- rnorm(2) %>% as.matrix
w22 <- rnorm(2) %>% as.matrix

```

Zeramos as previsões e iniciamos os loops de treinamento. Para a rede neural, precisamos de muitos exemplos de exposição, então embutimos em Mark II um parâmetro (*reps*) responsável por repetir o processo de treinamento com o dataset.

A rigor o melhor seria partitionar o dataset em fragmentos menores para cada epoch, mas vamos manter as coisas simples.

```

ypreds <- rep(0,dim(x)[1]) # inicializa predicoes em 0
yerrors <- rep(0,dim(x)[1]) # inicializa predicoes em 0
for (j in 1:reps){
  print(paste("This is training epoch:",j))
  print(paste("Current weights:",w1,w21,w21))
}

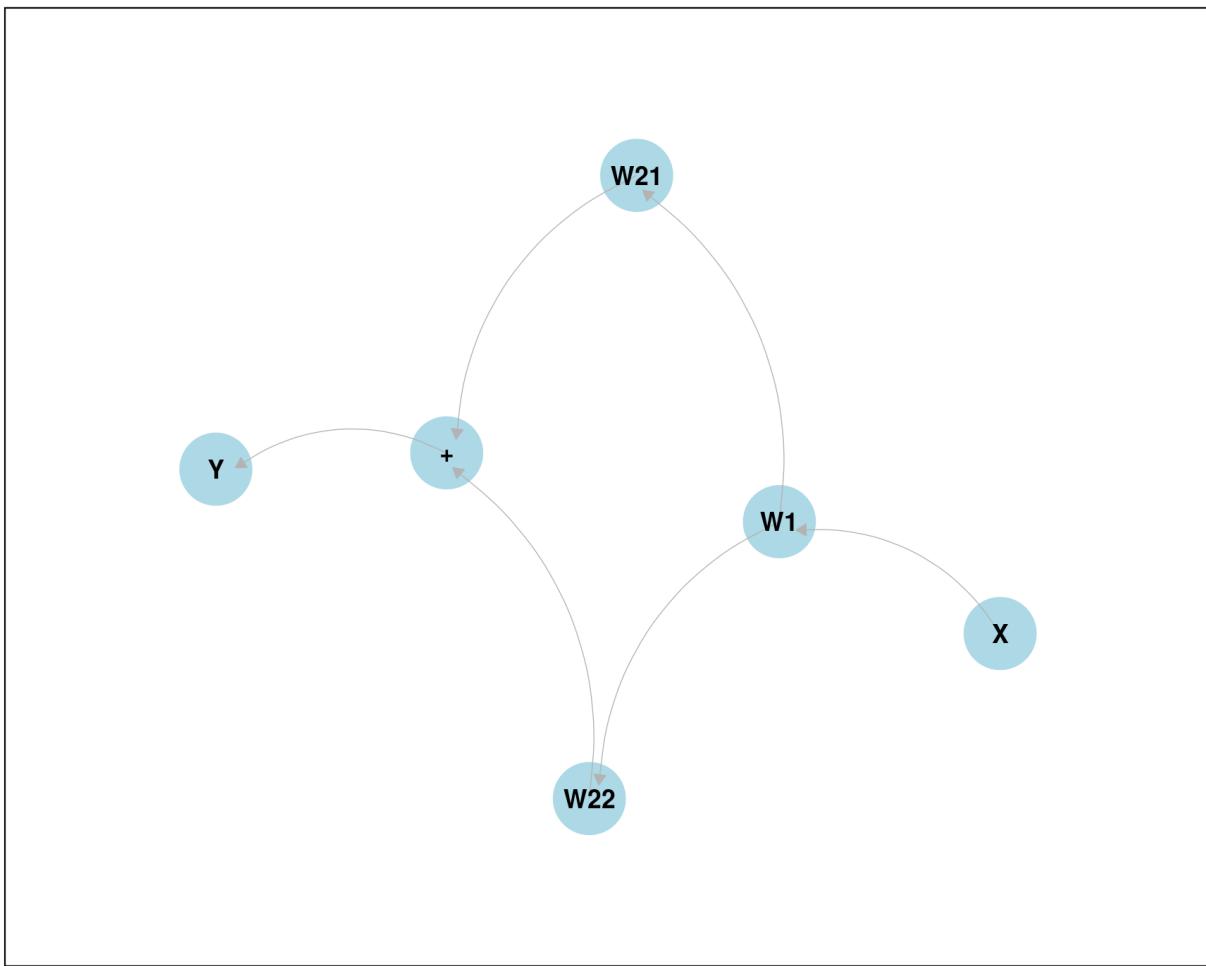
```

Predições: a primeira camada soma o produto de seus pesos pela entrada e pela unidade (*bias trick*). Os neurônios da segunda camada somam o produto de seus pesos pelo output. O output final é a soma dos outputs na camada intermediária.

```
# Processa as observações em x de forma aleatoria
for (i in sample(1:length(y), replace=F)) {
  # predicao
  ypred1 <- sum(w1 %*% c(as.numeric(x[i, ]), 1))

  ypred21 <- sum(w21 %*% as.numeric(ypred1))
  ypred22 <- sum(w22 %*% as.numeric(ypred1))

  out <- sum(ypred21, ypred22)
```



Agora, as regras de atualização dos pesos seguindo derivações com regra de cadeia. Para os neurônios intermediários, temos: $\frac{d}{dw_{21}}$ e $\frac{d}{dw_{21}}$ de $(target - (pred22 + pred21))^2$.

$$\frac{d}{dw_{21}} (target - (pred22 + pred21))^2$$

Aplicando a regra de cadeia e sabendo que a predição do segundo neurônio W_{22} não depende dos pesos em W_{21} :

$$\begin{aligned}
&= 2(\text{target} - (\text{pred22} + \text{pred21})) * \frac{d}{dw_{21}}(-1)(\text{pred22} + \text{pred21}) \\
&= 2(\text{target} - (\text{pred22} + \text{pred21})) * \frac{d}{dw_{21}}(-1)(w_{21} * \text{ypred1})
\end{aligned}$$

Que é a derivada para os pesos do perceptron:

$$= 2(\text{target} - (\text{pred22} + \text{pred21})) * (\text{ypred1})(-1)$$

Entretanto, calcular os pesos de w_1 em função da saída requer um pouco mais:

$$\begin{aligned}
&\frac{d}{dw_1}(\text{target} - (\text{pred22} + \text{pred21}))^2 \\
&= 2(\text{target} - (\text{pred22} + \text{pred21})) * \frac{d}{dw_1}(-1)(\text{pred22} + \text{pred21}) \\
&= 2(\text{target} - (\text{pred22} + \text{pred21})) * \frac{d}{dw_1}(-1)(\sum w_{22} \sum w_1 x + \sum w_{21} \sum w_1 x)
\end{aligned}$$

Usando a derivada de somas e verificando que os termos não relacionados ao w_1 avaliado somem:

$$2(\text{target} - (\text{pred22} + \text{pred21})) * (-1)(\sum w_{22}x + \sum w_{21}x)$$

```

# update em w . Eta ja ajustado para 1/2*eta
delta_w22 <- eta * (-1) * (y[i] - (ypred21 + ypred22)) * ypred1
delta_w21 <- eta * (-1) * (y[i] - (ypred21 + ypred22)) * ypred1
delta_w1 <- eta * (y[i] - (ypred21 + ypred22)) * -1 *
  (sum(w21 %*% c(as.numeric(x[i,]),1)) + sum(w22 %*% c(as.numeric(x[i,]),1)))

w1 <- w1 - delta_w1
w21 <- w21 - delta_w21
w22 <- w22 - delta_w22
ypreds[i] <- out # salva predicao21 atual
yerrors[i] <- ypreds[i] - y[i]
}
print(paste("Mean squared error:", mean((yerrors)^2)))
}
return(ypreds)
}

```

Então, podemos testá-lo em um dataset.

```

>train_df <- iris[, c(1, 2, 3)]
>names(train_df) <- c("s.len", "s.wid", "p.len")
>head(train_df)
>train_df[60:65,]

>x_features <- train_df[, c(1, 2)]
>y_target <- train_df[, 3]

# Convergência boa
>mark_ii_preds <- mark_ii(x = x_features, y = y_target,
                           eta=0.000001, reps = 40)

```

```

[1] "This is training epoch: 1"
[1] "Current weights: -0.45050790019773 -0.0197893400687895 -0.0197893400687895"
[2] "Current weights: 0.150011803623929 2.13458518518008 2.13458518518008"
[3] "Current weights: 1.48235899015804 -0.0197893400687895 -0.0197893400687895"
[1] "Mean squared error: 1133.22204821886"
(...)

[1] "This is training epoch: 2"
[1] "Current weights: -0.67126807499406 -0.0609395311239563 -0.0609395311239563"
[2] "Current weights: -0.0707483711724013 2.09343499412492 2.09343499412492"
[3] "Current weights: 1.26159881536171 -0.0609395311239563 -0.0609395311239563"
[1] "Mean squared error: 176.747586724131"
(...)

[1] "This is training epoch: 4"
[1] "Current weights: -0.791488817323548 -0.0700721883119202 -0.0700721883119202"
[2] "Current weights: -0.19096911350189 2.08430233693696 2.08430233693696"
[3] "Current weights: 1.14137807303222 -0.0700721883119202 -0.0700721883119202"
[1] "Mean squared error: 7.32496712284895"
[1] "This is training epoch: 5"
[1] "Current weights: -0.805708526415977 -0.0705118739404967 -0.0705118739404967"
[2] "Current weights: -0.205188822594319 2.08386265130838 2.08386265130838"
[3] "Current weights: 1.12715836393979 -0.0705118739404967 -0.0705118739404967"
[1] "Mean squared error: 3.31246116798174"
(...)
[1] "Mean squared error: 2.50706426321967"
(...)
[1] "Mean squared error: 2.50638029884829"
(...)
[1] "Mean squared error: 2.50640582517322"

```

Podemos observar o modelo convergindo à medida em que os pesos se estabilizam e nossa medida de erro cai. Usando o η acima, a rede costuma convergir com correlação $\rho \sim 0.60$ entre previsões e dados originais.

```

>acc_data <- data.frame(y_preds=mark_ii_preds,
                           y_targs=y_target)

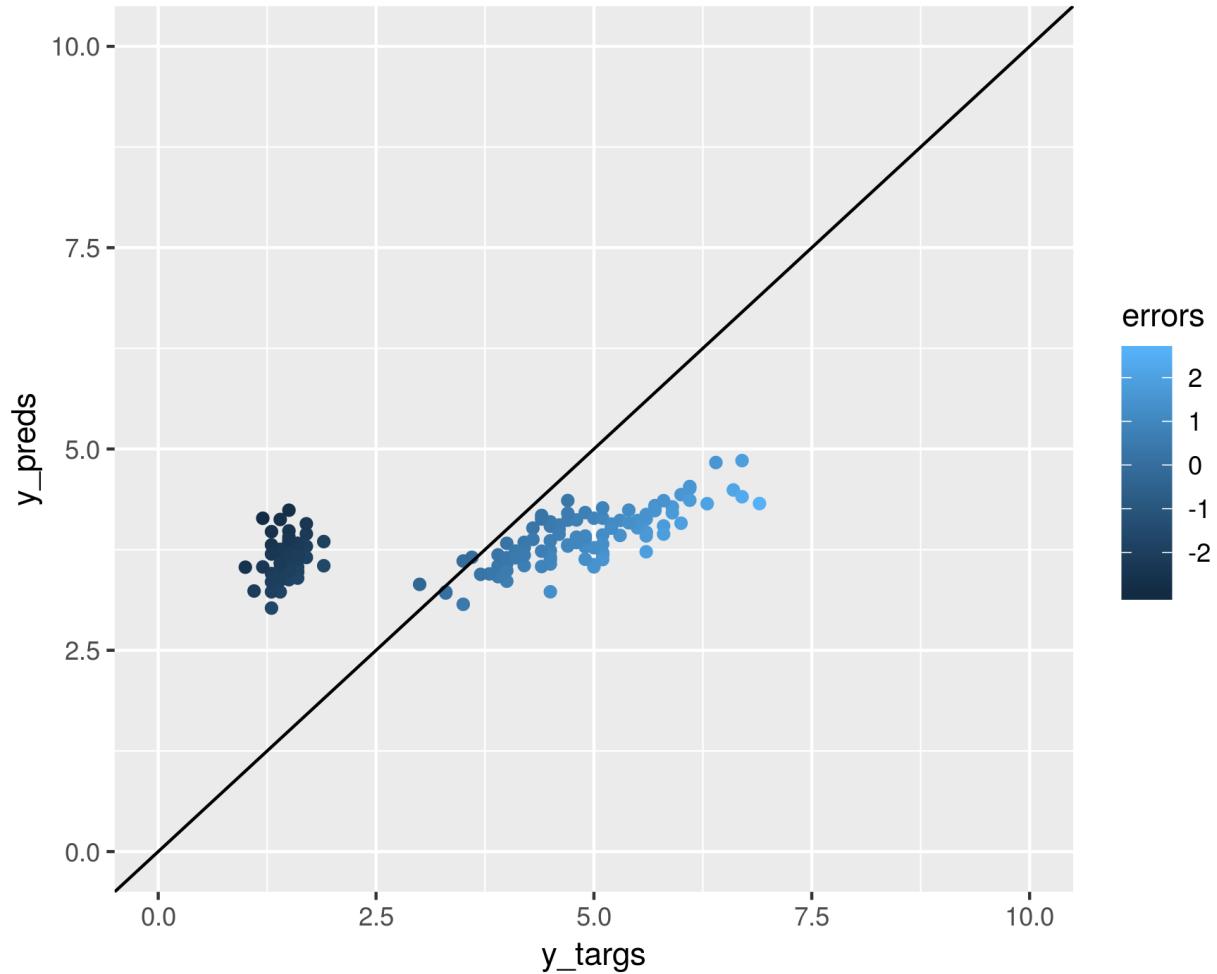
>acc_data$errors <- y_target - mark_ii_preds
>cor.test(acc_data$y_preds, acc_data$y_targs)

Pearson's product-moment correlation

data: acc_data$y_preds and acc_data$y_targs
t = 8.9717, df = 148, p-value = 1.203e-15
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
0.4788098 0.6883163
sample estimates:
cor
0.5935271

>ggplot(acc_data,aes(y=y_preds,x=y_targs,color=errors))+ 
  geom_point() + xlim(0,10) + ylim(0,10) +
  geom_abline(slope = 1,intercept = 0)

```



Referências

Para uma história completa sobre redes neurais: J. Schmidhuber. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61, p 85–117, 2015. (Based on 2014 TR with 88 pages and 888 references, with PDF & LATEX source & complete public BIBTEX file).

<http://web.csulb.edu/~cwallis/artificialn/History.htm> https://sebastianraschka.com/Articles/2015_singlelayer_neurons.html <https://rpubs.com/FaiHas/197581>

Início dos textos em construção

Capítulo 5 : Contexto e Inferência Bayesiana

Probabilidades

Uma abordagem da matemática aplicada que tem se popularizado é o de Inferência Bayesiana. Os procedimentos anteriores são usualmente descritos como frequencistas.

Ainda que a informação final de ambas (frequencista e bayesiana) possa convergir, a perspectiva muda radicalmente.

Por princípio, partimos de um ponto diferente.

Ao invés de usar probabilidades para testar hipóteses sobre *parâmetros*, modelamos os valores deles diretamente.

Um parâmetro é um símbolo, uma aproximação para uma ideia (*para*, “perto”, *metron*, “medida”). Em geral, usamos parâmetros para representar fenômenos que se comportam como números (e.g: existem elementos que podem ser ordenados por alguma noção de tamanho e estes podem ser somados e/ou multiplicados). Exemplos de parâmetros que podem ser estimados experimentalmente: capacidade ventilatória, altura média, IMC.

O médico liga para confirmar uma consulta marcada: “(…) confirma às 20:00 ou vai se atrasar?”. O horário de ‘20:00’ é uma estimativa pontual, porém sabemos que existe a chance de chegarmos 19:55 ou 20:05. Um frequencista responderia: “Confirmo. Diante do trânsito, posso atrasar. Porém, um atraso maior que 10 minutos é muito improvável ($p < 0,05$)”.

Um bayesiano responderia: “Confirmo. Diante do trânsito, é mais provável que eu chegue entre 20:00 e 20:05.” Trabalhamos com incertezas o tempo inteiro. Inferência Bayesiana desvia da modelagem probabilística de hipóteses; o objetivo é modelagem direta das distribuições dos parâmetros.

Em abordagem frequencistas, visamos a probabilidade de uma hipótese que versa sobre os valores de algum parâmetro. No teste t para duas amostras, definimos a hipótese nula em função das médias (μ) e outros parâmetros (σ, df). $H_0 : \mu_{amostra_1} > \mu_{amostra_2}$. Em abordagem bayesiana, estimamos diretamente as distribuições de $\mu_{amostra_1}$ e $\mu_{amostra_2}$ e podemos fazer inferências a partir delas.

Muitos métodos científicos: Feyerabend, Carnap e Quine

No primeiro capítulo, entramos em contato com o método hipotético-dedutivo e a falseabilidade como critério de demarcação científica. Apesar de dominante, esse racional possui vulnerabilidades interessantes. Entenderemos melhor argumentos contrários e propostas alternativas através de três filósofos do século XX. Esse é um momento conveniente, uma vez que tiramos os holofotes das hipóteses.

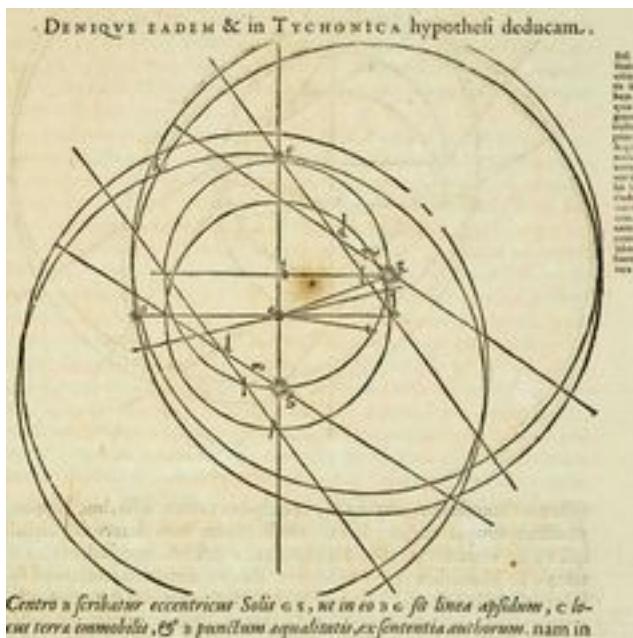
Paul Feyerabend (1924 - 1994)

Conhecido pela personalidade ímpar e por ideias radicais, Paul Feyerabend, em *Contra o Método*(1975), argumenta que boa parte dos avanços significativos aconteceram fora do método científico.

Crenças pessoais e detalhes biográficos são responsáveis por mudanças em nosso conhecimento. Mais que isso, usar falsificabilidade (como vimos no *Capítulo I* sobre K. Popper) e o método hipotético-dedutivo teriam nos feito rejeitar o heliocentrismo e outras ideias chave. Na verdade, o sistema geocêntrico (Terra no centro do sistema) de Ptolomeu era mais acurado (!) que o de Copérnico (Sol ao centro) usando um mesmo número de parâmetros para cálculos das órbitas. O sistema copernicano estava mais próximo da realidade como entendida hoje, porém o estágio intermediário de concepção teórica era ‘pior’²⁷.

Além de menos acurado, era mais complexo em alguns aspectos, incluindo mais epicíclos. A Revolução Copernicana somente consolidou a mudança de paradigma com contribuições subsequentes de de Tycho Brahe, Kepler, Galileo e Newton, cerca de 1 século depois.

²⁷Stanley E. Babb, “Accuracy of Planetary Theories, Particularly for Mars”, *Isis*, Sep. 1977, pp. 426



Centra sfericarum eccentricarum Solis &c. ut in ea > e. sit linea apsidum, et loca terra immobilitatis, & punctum aequalitatis ex centro eiusdem. nam in

Diante das incongruências entre um método e as inevitáveis imprevisibilidades da empreitada humana em conhecer o Universo, Feyerabend propõe o *anarquismo epistêmico* sob o mote “*Anything goes*” (‘Vale tudo’). Isto é, quaisquer recursos são válidos na tentativa e atacar um problema.

É tentador pensar que, dada a profundidade do trabalho, a defesa de uma postura tão contundente é obviamente uma aplicação dos preceitos defendidos no livro como necessários para disseminar uma ideia. Outros filósofos nos ajudam a conceber uma ciência não pautada num método hipotético-dedutivo de maneira mais construtiva.

Rudolph Carnap (1891 - 1970)

Carnap, do Círculo de Viena, também contrapôs Popper. Em “Testability and Meaning” (1936-7), argumenta que falsificabilidade não difere de verificacionismo. Envolve a testagem de cada assertiva em si, um problema que outros também endereçaram.

Diante de resultados inesperados em um experimento, o procedimento automático para um cientista envolve checar a integridade das condições desenhadas. Verificar a composição da amostra, os métodos de coleta, mecanismos de perda, critérios de exclusão e inclusão, premissas da análise. Isso não é desonestade intelectual: são fatores menores reais e facilmente abordáveis que podem ter invalidado a teoria de base. O mesmo se dá para técnicas de análise e conceptualização de construtos.

O cuidado com esses pontos é deseável e desnuda o inevitável calcanhar de Aquiles da falsificabilidade. É impossível refutar uma hipótese/assertiva de maneira isolada. Cada procedimento experimental ou lógico envolve a interdependência entre os símbolos usados.

Willard van Orman Quine (1908 – 2000)

Uma escola filosófica parte do problema acima. A tese de Duhem-Quine postula que é impossível testar qualquer hipótese científica, uma vez que sempre há premissas aceitas como verdade.

Em ‘*Os dois dogmas do empiricismo*’, Quine considera as proposições e as relações lógicas entre elas apenas um sistema, que só pode ser estudado em conjunto. Os exercícios ilustrados no volume anterior testa a adequação dos dados à família de distribuições t. Também assume que tamanhos dos bicos são mensuráveis usando números e que estes podem ser comparados com valores de outras amostras.

A princípio, essas declarações parecem triviais. Entretanto, considerando os fatores humanos da ciência, a mudança de lentes é significativa. Abandonando o esquema de testagem de hipóteses como eixo, o *valor p* deixa de ter papel central na narrativa. Integra um conjunto de informações maior sobre os parâmetros examinados.

Discutivelmente, abordar um problema dessa maneira é historicamente mais frutífero. As contribuições mais contundentes são advindas de cientistas dedicados a estudar um contexto ou problema como um todo. É raro, talvez inédito, que um grupo operando de forma sistemática com o método hipotético-dedutivo tenha obtido avanços consistentes.

Estimar livremente os parâmetros de que falamos é muito mais intuitivo que adequar uma ideia aos procedimentos de testagem de hipóteses.

Bayesian estimation

Para a abordagem anterior, ao fazer um test t, calculamos a estatística t correspondente às diferenças encontradas e então a probabilidade de obter valores iguais ou mais extremos.

Agora, faremos algo mais simples e intuitivo. Vamos estimar um parâmetro: a diferença entre os grupos. Na verdade, valores prováveis dela. Todas as inferências subsequentes serão derivadas da distribuição produzida por nosso procedimento.

Novamente, usaremos 30 observações retiradas de amostras de distribuição normal ($\mu_a = 0; \mu_b = 0.6; \sigma_a = \sigma_b = 1$) normais. Usando a library BEST, é possível usar inferência bayesiana para responder nossa pergunta (“Como é a distribuição da diferença entre A e B (...)?”). Aplicamos o estimado sobre as amostras A e B e, em seguida, plotamos as distribuições.

```
>library(ggthemes)
>library(rstan)
>library(reshape2)
>library(BEST)
>library(ggplot2)
>options(mc.cores = parallel::detectCores() - 1)
>a <- rnorm(n = 30, sd = 1, mean = 0)
```

```

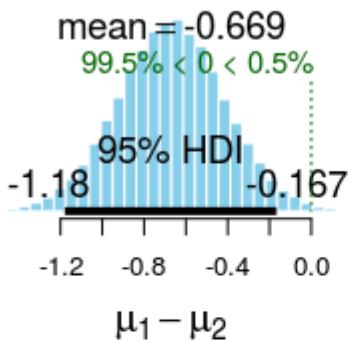
>b <- rnorm(n = 30, sd = 1, mean = 0.6)

# BEST
>BESTout <- BESTmcmc(a, b)

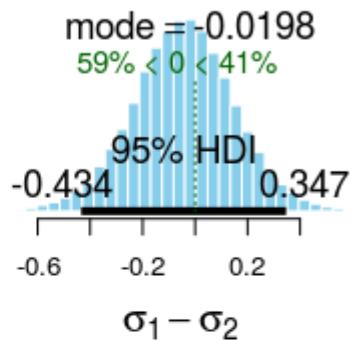
### BEST plots
>par(mfrow=c(2,2))
>sapply(c("mean", "sd", "effect", "nu"), function(p) plot(BESTout, which=p))
>layout(1)

```

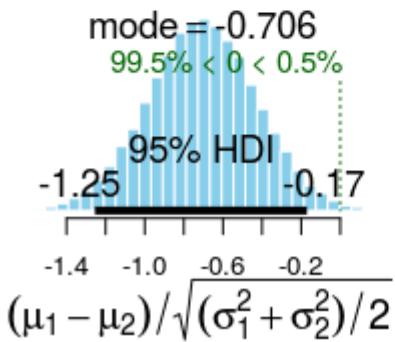
Difference of Means



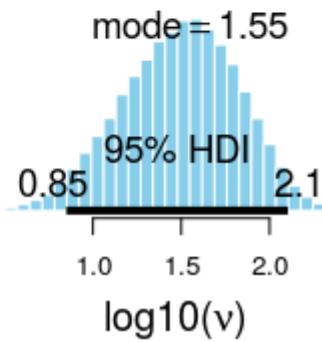
Difference of Std. Dev.s



Effect Size



Normality



A distribuição no canto superior esquerdo corresponde à nossa estimativa da diferença entre A e B. Com ela, podemos fazer estimativas pontuais ($diff_{\mu_a \mu_b} = -0.669$). O intervalo apontado como 95% HDI (High density interval) contém 95% da distribuição.

Por trás das cortinas

Obviamente, vamos entender a arte permitindo estimar essas distribuições. A flexibilidade e o poder dos modelos bayesianos permite lidar com uma série de problemas dificilmente tratáveis de outra forma. Entretanto, é fácil cair em armadilhas ou esbarrar em dificuldades durante o processo.

Nesse framework, lidamos com distribuições. É extremamente importante entender os componentes envolvidos para não cometer erros importantes.

MODIFIED BAYES' THEOREM:

$$P(H|x) = P(H) \times \left(1 + P(C) \times \left(\frac{P(x|H)}{P(x)} - 1 \right) \right)$$

H: HYPOTHESIS

X: OBSERVATION

P(H): PRIOR PROBABILITY THAT H IS TRUE

P(x): PRIOR PROBABILITY OF OBSERVING X

P(C): PROBABILITY THAT YOU'RE USING
BAYESIAN STATISTICS CORRECTLY

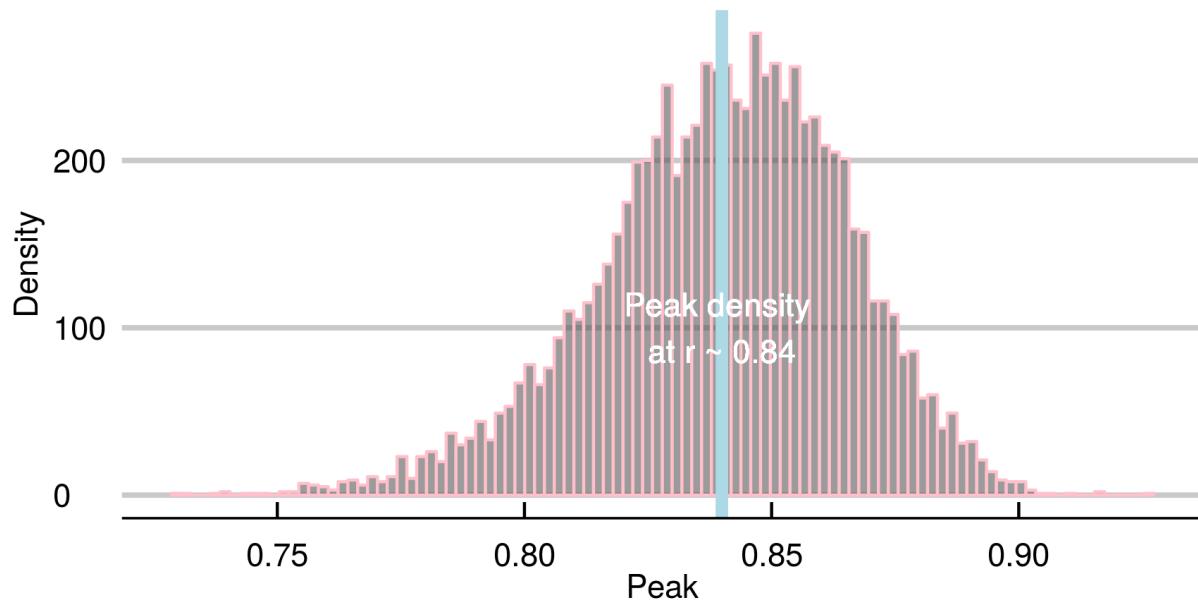
Figure 26: <https://xkcd.com/2059/>

Então, seguindo o exercício anterior, precisamos especificar que consideramos as duas amostras vindo de distribuições t com médias $\mu_1 e \mu_2$ e desvios-padrão idênticos, $\sigma_a = \sigma_b$.

@ Teorema de Bayes: posterior = Likelihood function x prior / evidence @ Posterior para correlação @ Posterior para diferença entre médias @ Posterior para correlação

```
>sample_data <- list(y_1=a,y_2=b,N=length(a))
>fit <- rstan::stan(file="aux/best.stan",
  data=sample_data,
  iter=3000, warmup=100, chains = 6)

>muDiff <- extract(fit, par='mudiff')$mudiff
>means <- lapply(list(a,b), mean)
>sample_diff <- diff(as.numeric(means)) # observed in data
>ggplot(as.data.frame(muDiff),aes(x=muDiff))+ 
  geom_histogram(alpha=0.6,color="pink")+
  geom_vline(xintercept=-sample_diff,
             color="light blue",size=2)+ # line for observed difference
  xlab("Difference of Means")+ylab("Frequency")+
  geom_text(label="Sample difference",
            color="white",x=mean(muDiff),y=500)+
  theme_economist_white(gray_bg = F)
```

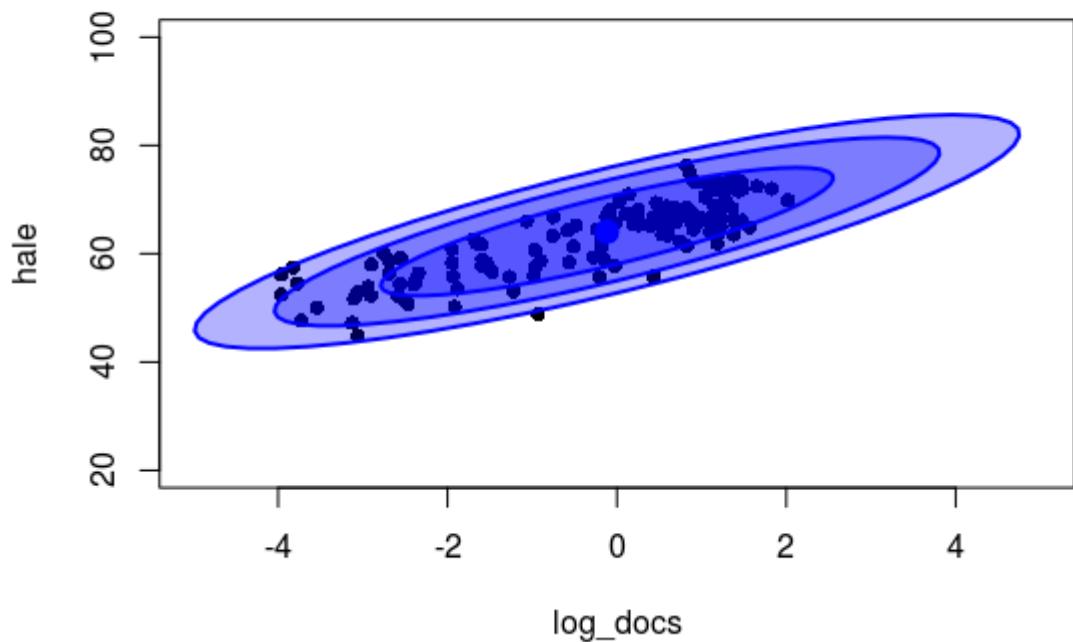


@Correlacao

```
>fit <- rstan::stan(file="aux/corr-docs.stan",
  data=uni_df,
  iter=3000, warmup=100, chains = 6)
```

Ellipse:

```
>x.rand = extract(fit, c("x_rand"))[[1]]
>plot(uni_df[,c("log_docs","hale")],
  xlim=c(-5,5), ylim=c(20, 100), pch=16)
>dataEllipse(x.rand, levels = c(0.75,0.95,0.99),
  fill=T, plot.points = FALSE)
```



- Classificação
 - Regressão logística @ Numero de euler
 - Modelos hierárquicos
- Flexibilidade Bayesiana
 - Usando priors
 - O estimador Markov Chain Monte Carlo