



Ciência de dados: Filosofia e aplicações.

Felipe Coelho Argolo

Página intencionalmente deixada em branco.

Versão 0.31: Introdução; Capítulo 0; Capítulo 1; Capítulo 2; Capítulo 3 (em progresso). 31 de Dezembro de 2018

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 fazer inferências com base em observações de cenários parecidos. Civilizações antigas, como os babilônios, usavam interpolação linear para estimar informações. Fazendo o censo populacional com intervalo de anos, estimavam o valor dos anos não medidos, supondo que eles eram medidas centrais daquelas ao seu redor. 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.

Objetivos

Este texto oferece uma introdução intuitiva ao campo, contextualizando-o epistemologicamente. O campo de aprendizagem estatística tem definição pouco estabelecida. Abrange aspectos de matemática pura e aplicada. Com uma perspectiva mais geral, a matemática pura desenvolve abstrações básicas, descrevendo o comportamento de números, probabilidades, funções e outras entidades. Veremos que progressos fundamentais foram feitos por nomes como De Moivre, Euler e Gauss.

Em matemática aplicada, especialistas estudam a relação dessas abstrações com fenômenos observáveis. Estas pessoas empregam métodos quantitativos a contextos restritos: por exemplo, James Clerk Maxwell deduziu (1860) a distribuição estatística e velocidade de partículas em um gás ideal, conhecida como distribuição de Maxwell–Boltzmann. Em estatística, veremos a descoberta da distribuição t para as estimativas de uma média por William Gosset.

São exemplos de campos que fazem uso extenso das ferramentas descritas: neurociências (modelos lineares em fmri), psicometria (análise fatorial), ecologia, biologia molecular (testes estatísticos), ciências clínicas (meta-análises e inferência causal), economia, marketing, algotrading.

Este texto introduz e fornece um guia para aplicações práticas destas ferramentas a fenômenos observáveis. É destinados aos profissionais e pesquisadores trabalhando na fronteira entre matemática aplicada e ciências naturais.

O primeiro capítulo ilustra como o racional hipotético-dedutivo funciona para estudar teorias científicas. Aborda a relação entre ciências empíricas e três abstrações matemáticas: a distribuição normal, a distribuição t e o teorema do limite central. O segundo capítulo aborda correlações e modelos preditivos lineares. Um framework frequencista e linguagem R são usados para demonstrações de exemplos e exercícios.

O terceiro capítulo apresenta um racional diferente para os procedimentos. Usando o conceito de holismo epistemológico (van Quine), reproduzidos as análises anteriores usando inferência bayesiana. Fazemos perguntas diferentes para obter informações de nossos dados. No capítulo quatro, o foco está em modelos classificatórios e na função logística. Usamos 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.

O quarto capítulo ilustra o uso de grafos/redes para a construção de modelos preditivos. Os exemplos são de Support Vector Machine e Redes Neurais. Modelos são construídos do zero (from scratch) para ilustrar dois mecanismos importantes de otimização (gradient descent e back propagation).

Sumário

Capítulo 0 - Ferramentas: programação com estatística básica

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

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

- Teorema do limite central e Distribuição normal
- Distribuição t
- Método hipotético-dedutivo e Testes de hipótese
- Valor p

Capítulo 2 - Sobre a natureza das relações

- Prelúdio: Quem precisa do valor p?
- Tamanho de efeito
- Correlações
- Coeficiente de correlação ρ de Pearson
- Regressão linear

Em construção:

Capítulo 3 - Contexto e inferência Bayesiana

- Intuições sobre distribuições probabilísticas
- Inferência Bayesiana para teste de diferenças e correlação linear
- Classificação
 - Regressão logística
 - Modelos hierárquicos
- Flexibilidade Bayesiana
 - Usando priors
 - O estimador Markov Chain Monte Carlo

Capítulo 4 - Redes neurais

- Support Vector Machines
- Gradient Descending
- Redes Neurais
 - Backpropagation
 - Deep learning (múltiplas camadas)

Capítulo 5 - Programação probabilística para contextos gerais

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

Capítulo 6 - 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

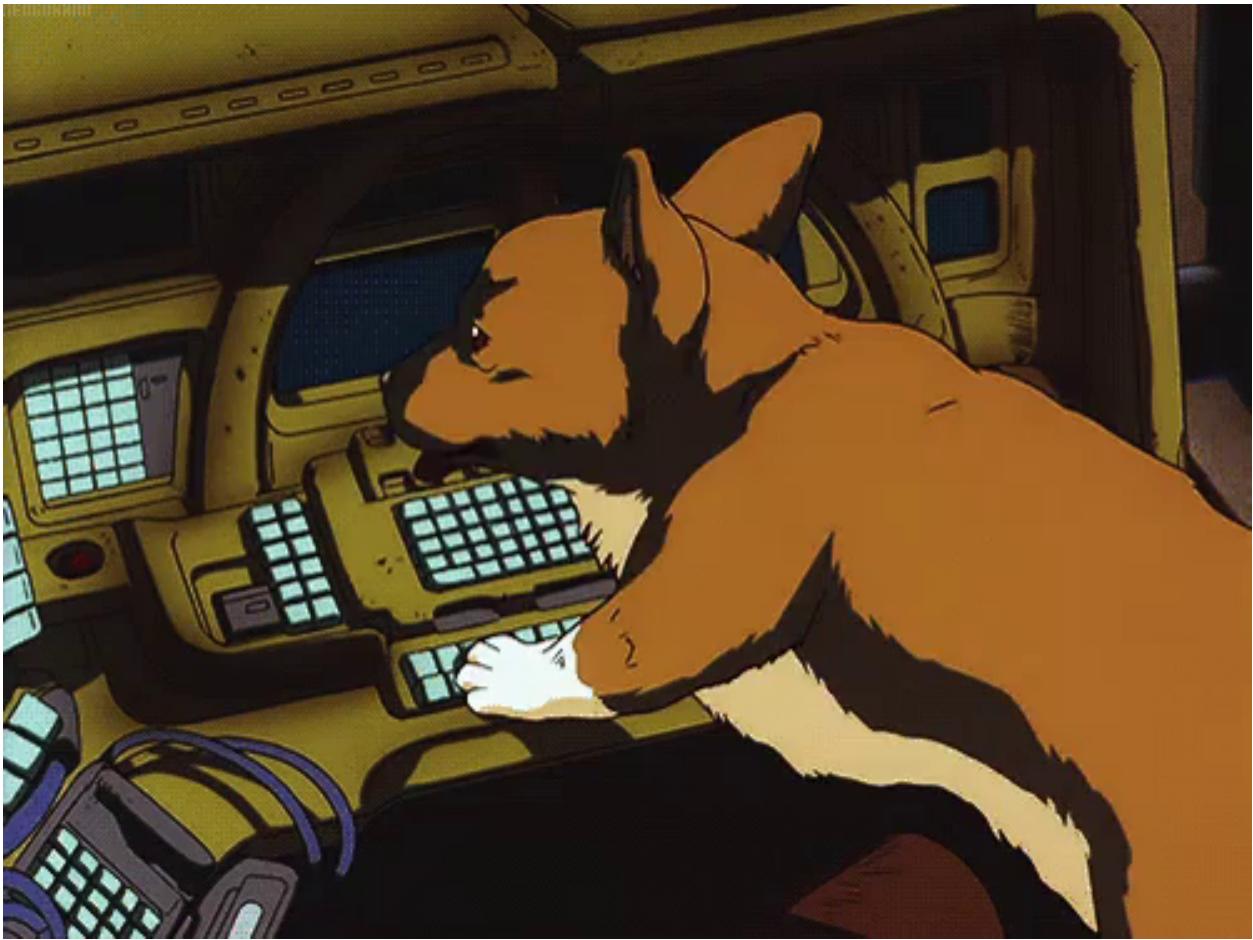
- Better Explained (<https://betterexplained.com/>)
- What is mathematics - Courant & Robbins
- Fundamentos da matemática elementar - Iezzi (Vol. 5)
- MOOCs sobre estatística básica usando R (e.g.: <https://www.coursera.org/specializations/statistics>)
- Cálculo Diferencial e Integral - Piskounov =)
- <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--EfW8dtjAuPoTuPcqmvOV53Fu>)

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 caminho 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 a 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 “ $\exp(\exp(3))$ ”. Usando parênteses, partimos da última computação. Usando o pipe (`%>%`), começamos com a primeira operação.

Notem que para usar um recurso da biblioteca magrittr, carregamos ela 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 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] [,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.

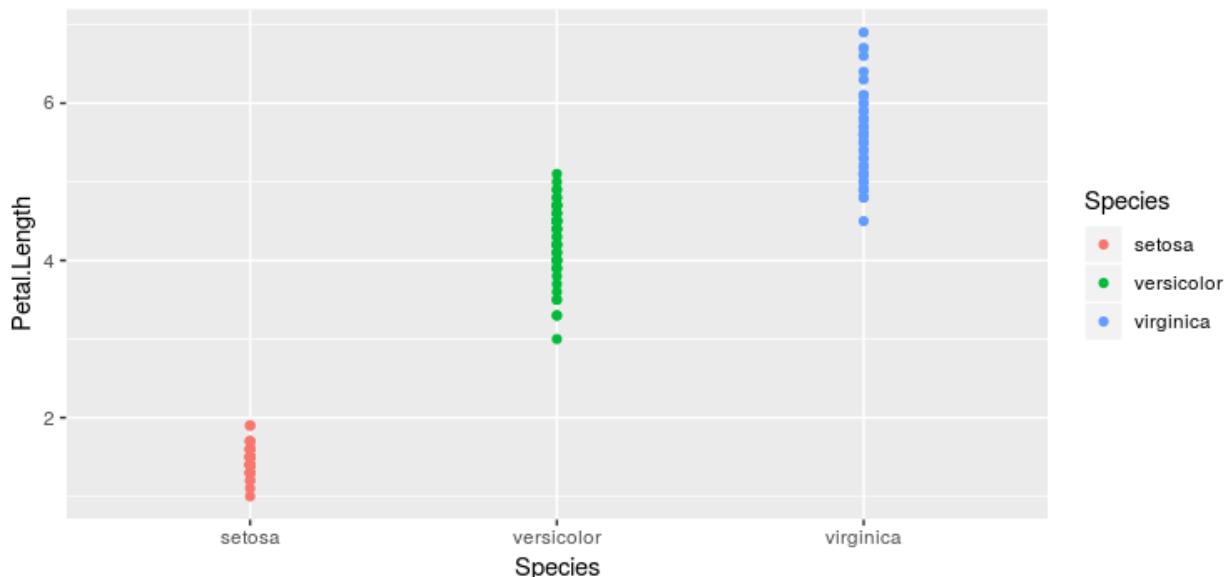
```
>head(sleep)
extra group ID
1 0.7    1 1
2 -1.6    1 2
3 -0.2    1 3
```

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.

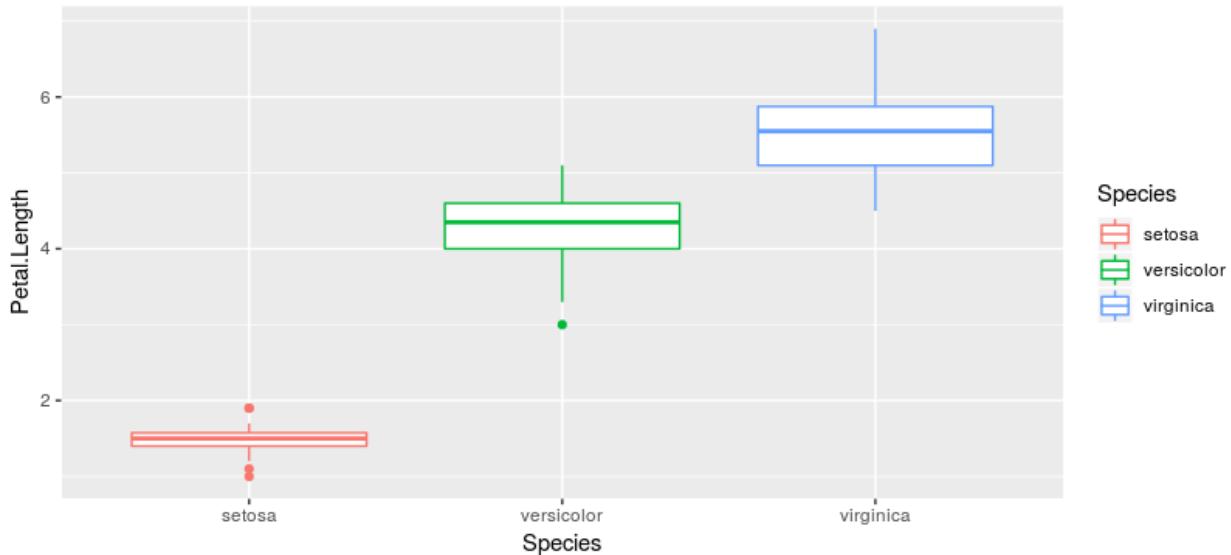
```
>library(ggplot2)
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+
  geom_point()
```



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()
```

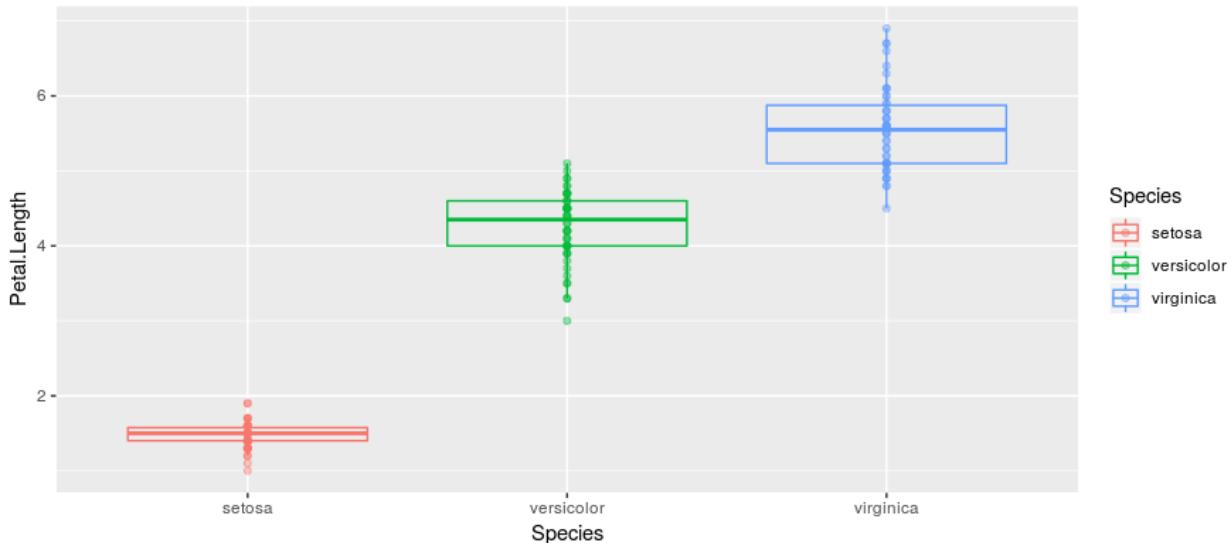
³Bertin, J. (1983),*Semiology of Graphics*, Madison, WI: University of Wisconsin Press



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:

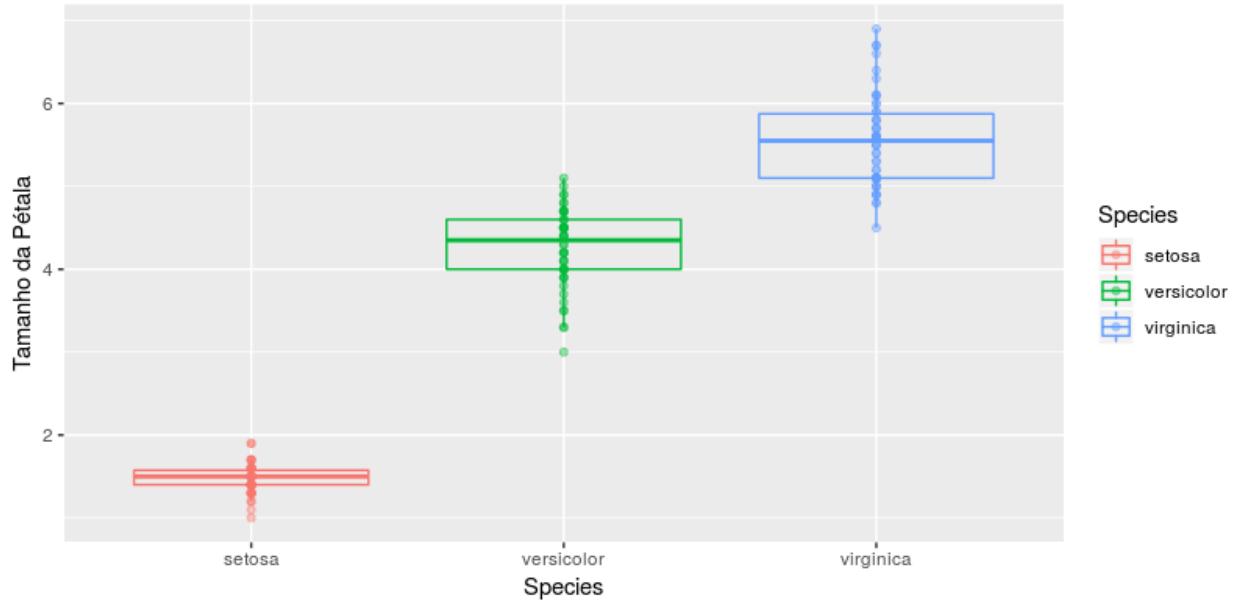
```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+  
  geom_point(alpha=0.4)+ # camada 1  
  geom_boxplot(alpha=0) # camada 2
```



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:

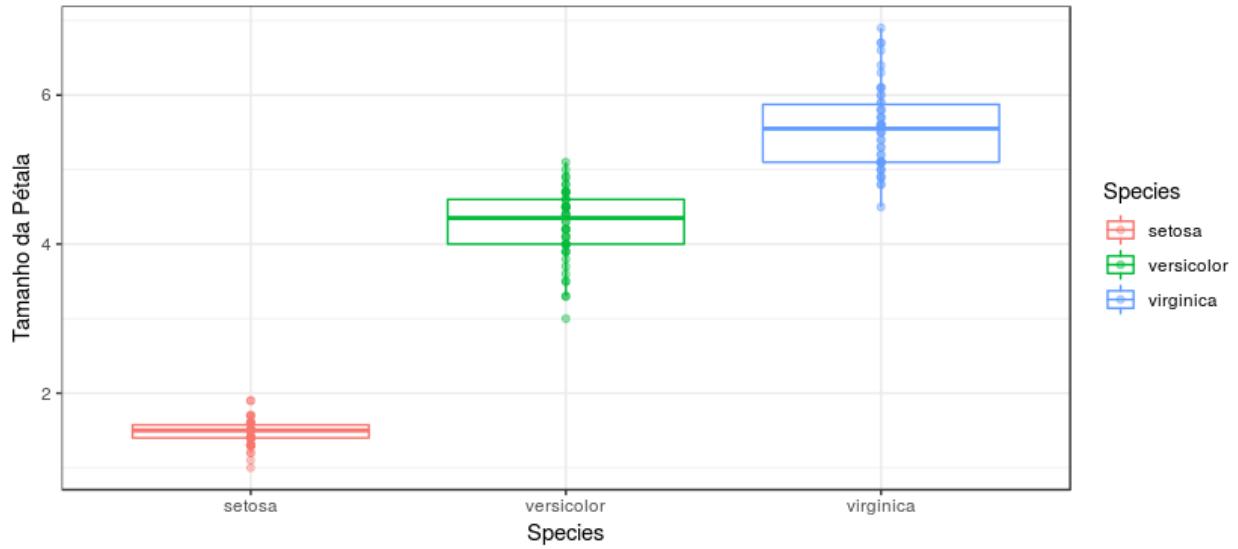
```
>ggplot(data=iris,aes(y=Petal.Length,x=Species,color=Species))+  
  geom_point(alpha=0.4)+ # camada 1  
  geom_boxplot(alpha=0)+ # camada 2
```

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

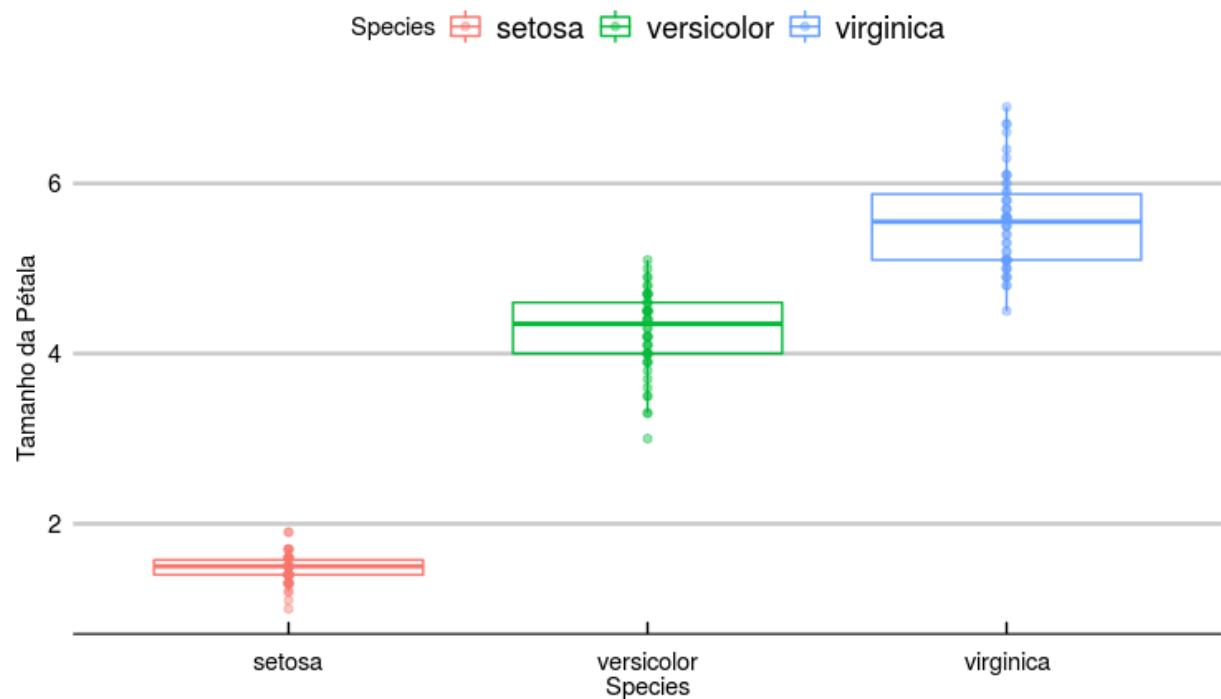


Ainda, existem temas prontos para mudar o estilo geral da imagem:

```
>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:

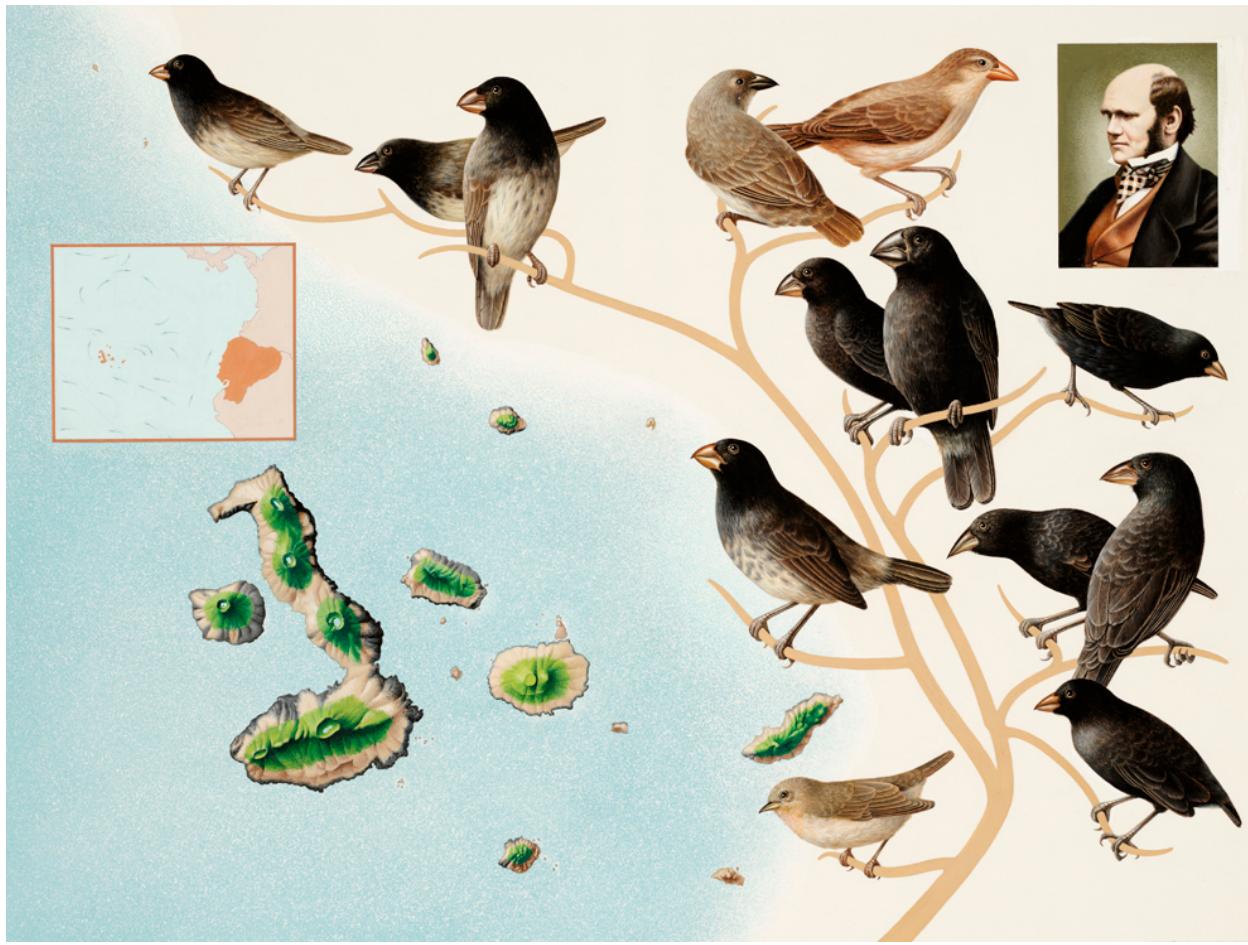
```
>map(.f = function(y) (y - mean(arg))^2, .x = arg) %>% # Define e aplica função
  unlist(.) %>% sum(.) / (length(arg) - 1) # Soma as distâncias e divide por n-1
```

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:rnorm(n=mean(c(3,2,1,16)))))
 - 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



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

Testes estatísticos e distribuições probabilísticas

Introdução

Charles Darwin observou que os pássaros fringilídeos 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.

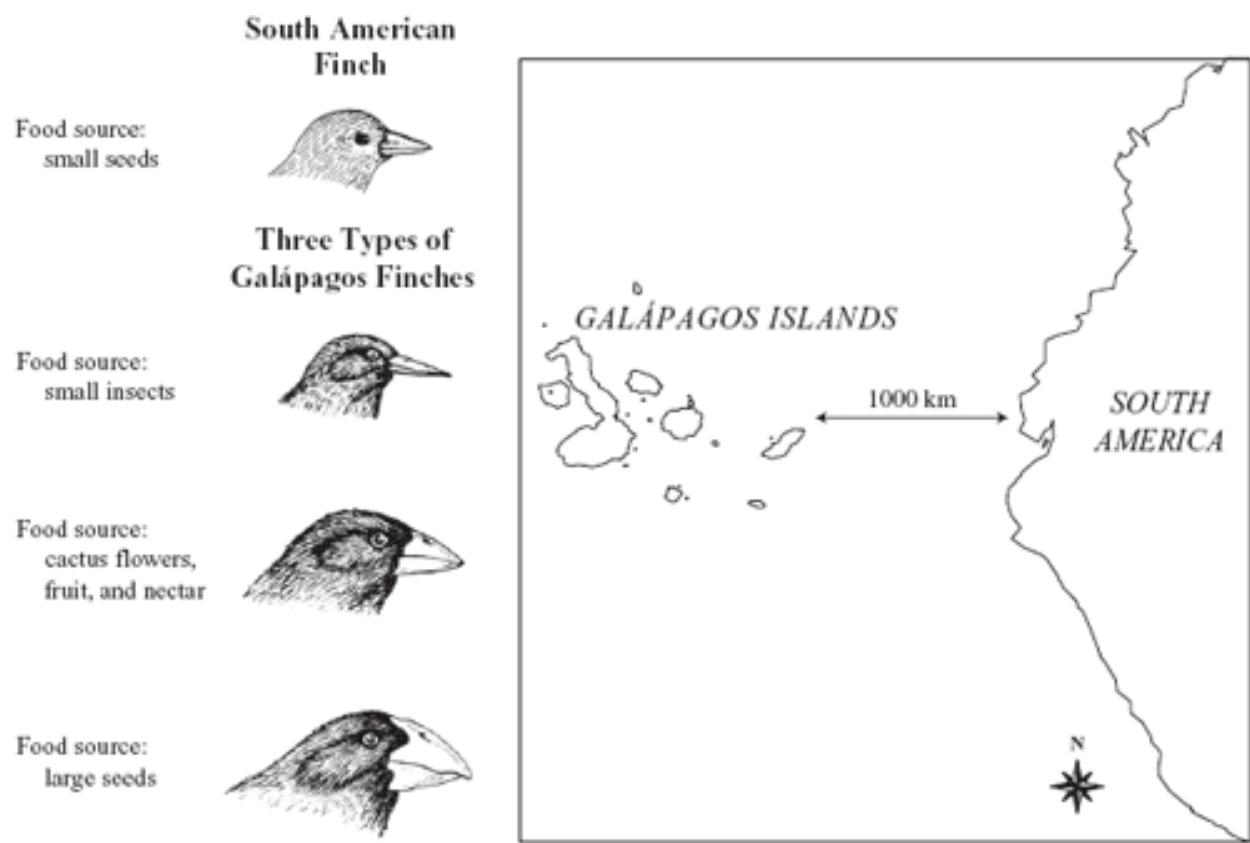


Figure 2: Figura 1. Fringilídeos em Galápagos.

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

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

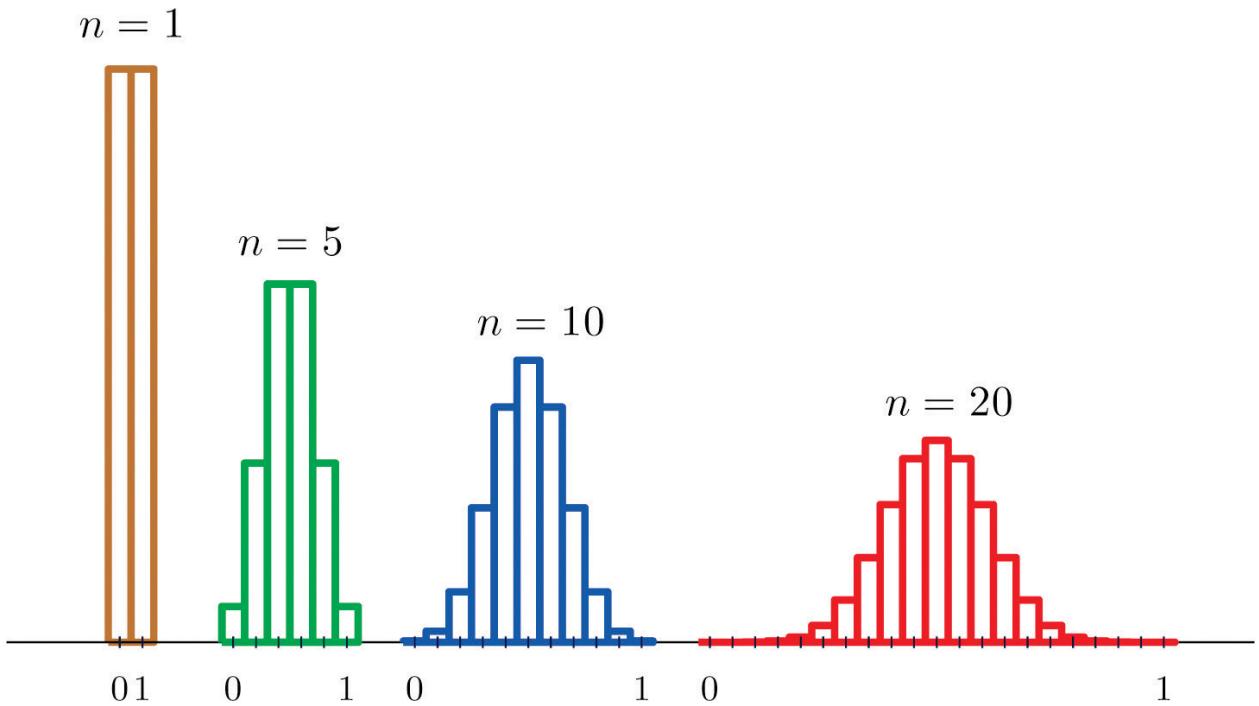


Figure 3: Figura 2. Distribuição binomial com $p = 0.5$ (e.g: lançamento de uma moeda). Para $n > 1$, valores extremos indicam resultados com apenas caras (cauda à esquerda, 0000...) ou coroas (cauda à direita, 1111...)

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 extraçalham 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 fringíldeos 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.

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 ($p(\text{car}) = p(\text{cor}) = 0.5$), aproximam-se de uma curva suave (contínua) à medida em que o n aumenta.

Para $n = 1$, temos uma distribuição uniforme $P(1) = P(0) = 0.5$. Para um número maior de lançamentos, os resultados mais frequentes são números parecidos de caras (0s) e coroas (1s).

Para $n = 10$, é muito mais provável obter um número de caras próximo a 5 (centro das curvas) que um

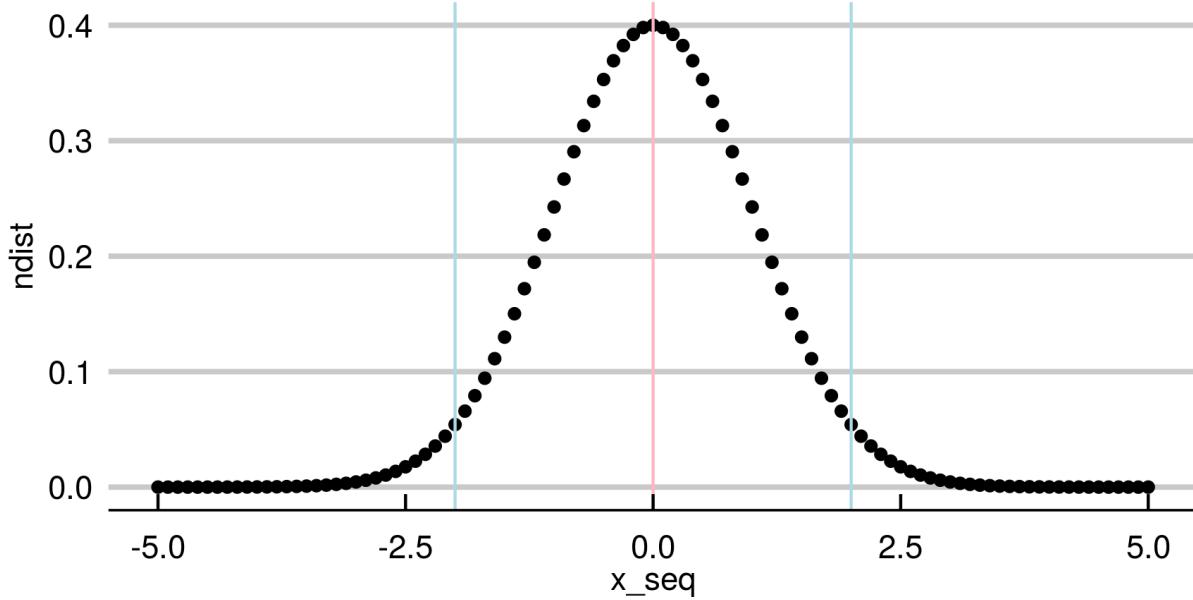


Figure 4: Figura 3: Distribuição normal (gaussiana), cujo formato lembra o de um sino

resultado com 9 ou 10 lançamentos iguais. É possível demonstrar que aumentar o valor de n faz que 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. Mas qual curva? Parecia o de um número elevado aos valores de uma quadrática negativa: $P(x) = ?^{-x^2}$. Lembre-se de que a função quadrática $f(x) = x^2$ correspondente geometricamente a uma figura convexa, uma parábola com ponto central inferior e extremidades tendendo ao infinito. Sua versão negativa, $f(x) = -x^2$ é a figura espelhada, com um ponto central superior e extremidade tendendo a valores negativamente infinitos. Ao exponenciarmos um número a $-x^2$, temos $P(x) = ?^{-x^2}$, com ponto máximo no centro e extremidades tendendo a zero. Entenderemos o porquê.

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 misterioso em questão é o número de Euler ($e \sim 2.72\dots$), que será melhor explicado no capítulo 3. 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)
```

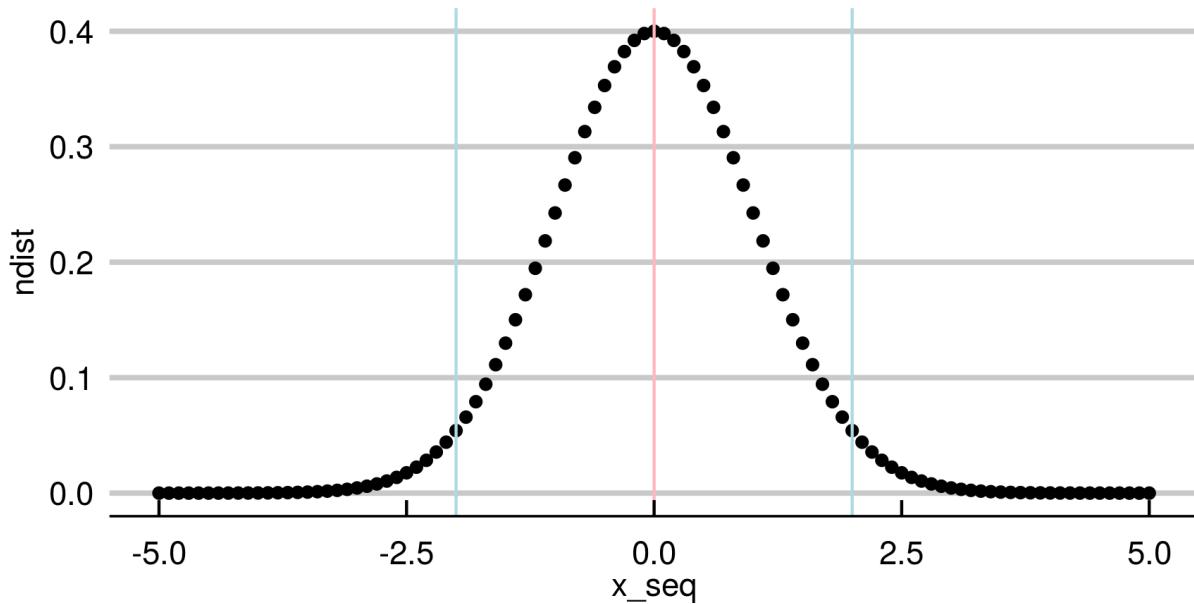
Em seguida, obter valores no intervalo $[-5, 5]$ e plotá-los:

```
>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.

É evidente que $-x^2$ sempre retorna valores 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 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 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:

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

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.

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\_\(ggplot2\).html

```

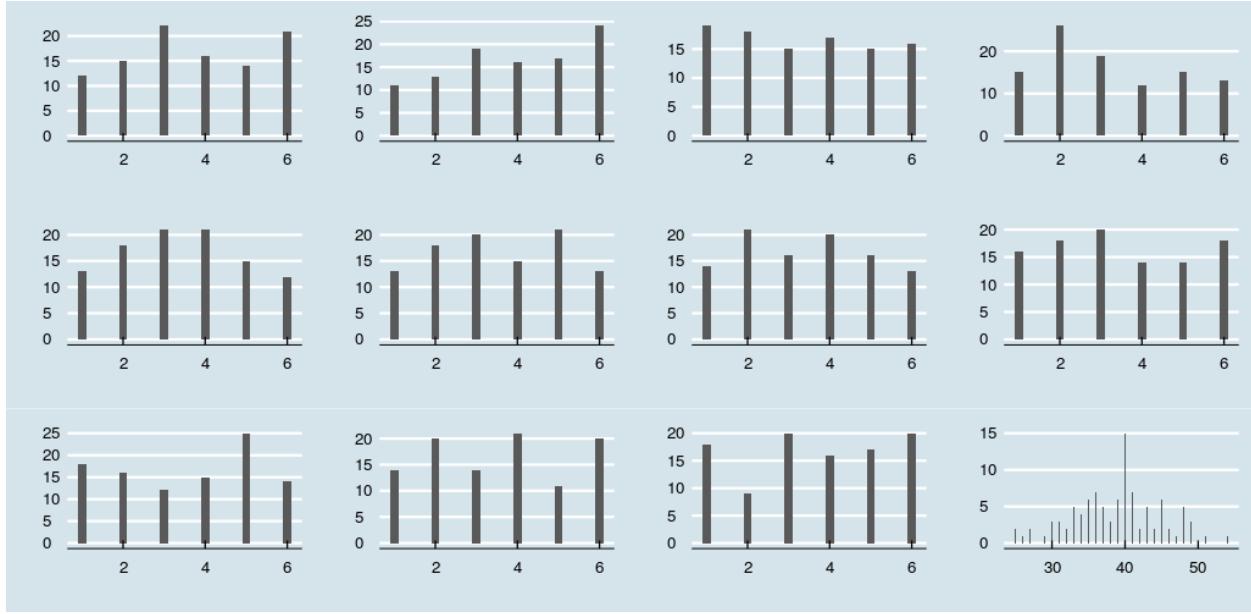


Figure 5: Figura 2 . Soma de amostras ($n=100$) de 11 distribuições uniformes correspondentes ao lançamento de dados honestos de 6 faces. O resultado está na célula inferior à direita.

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.⁶ Este resultado tem uma sutil importância 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 simplificações de fenômenos complexos. 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.

⁶Prova formal em <http://www.cs.toronto.edu/~yuvalf/CLT.pdf>



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). Os fenômenos observáveis em nosso universo são naturalmente complexos. 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.

A descoberta das equações que regem esses mecanismos de convergência foi um grande avanço para as ciências experimentais.

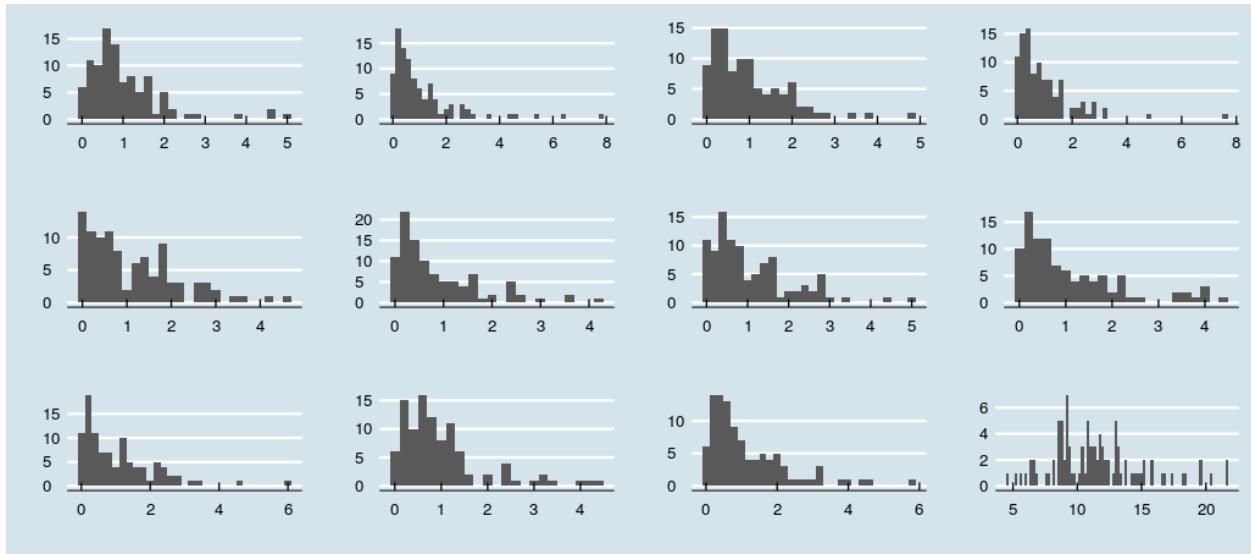


Figure 6: Figura 3. Soma de amostras ($n=100$) de 11 distribuições gama. O resultado está na célula inferior à direita. Função de densidade de probabilidade para distribuição gama: $f(x) = 1/\Gamma(\alpha) * \beta^\alpha * x^{\alpha-1} * e^{-\beta x}$, com $\alpha = \beta = 1$

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.

Darwins's Finches

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)+
```

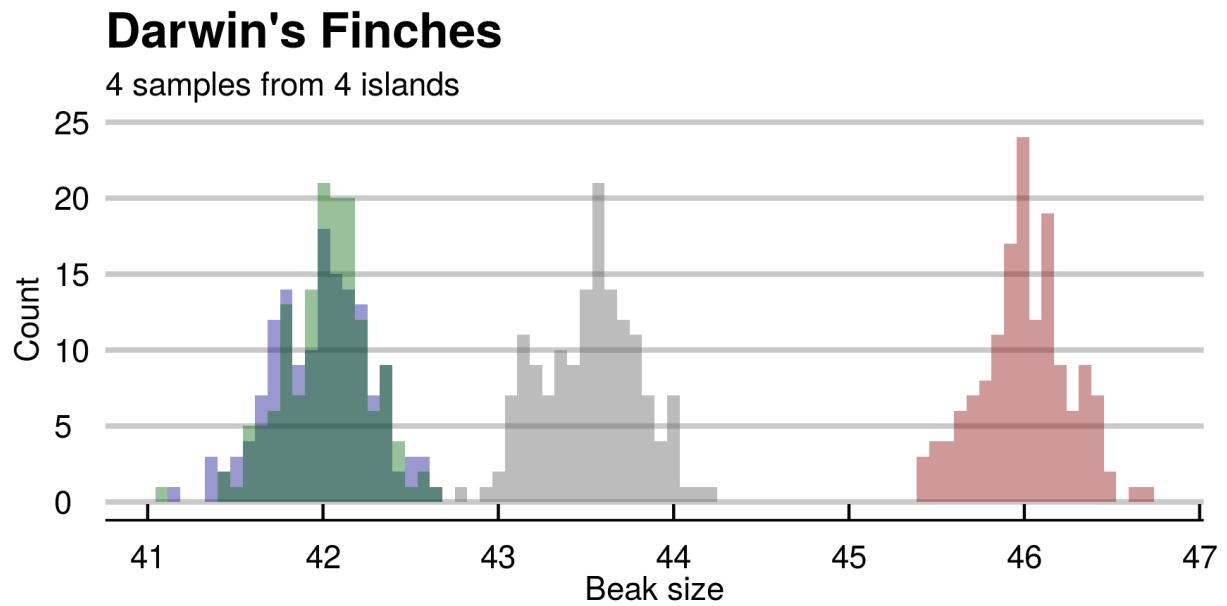


Figure 7: Figura 4. Distribuição das medidas de bicos em populações simuladas para genes com efeito aditivo.

```
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*), dado pelo 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:

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

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

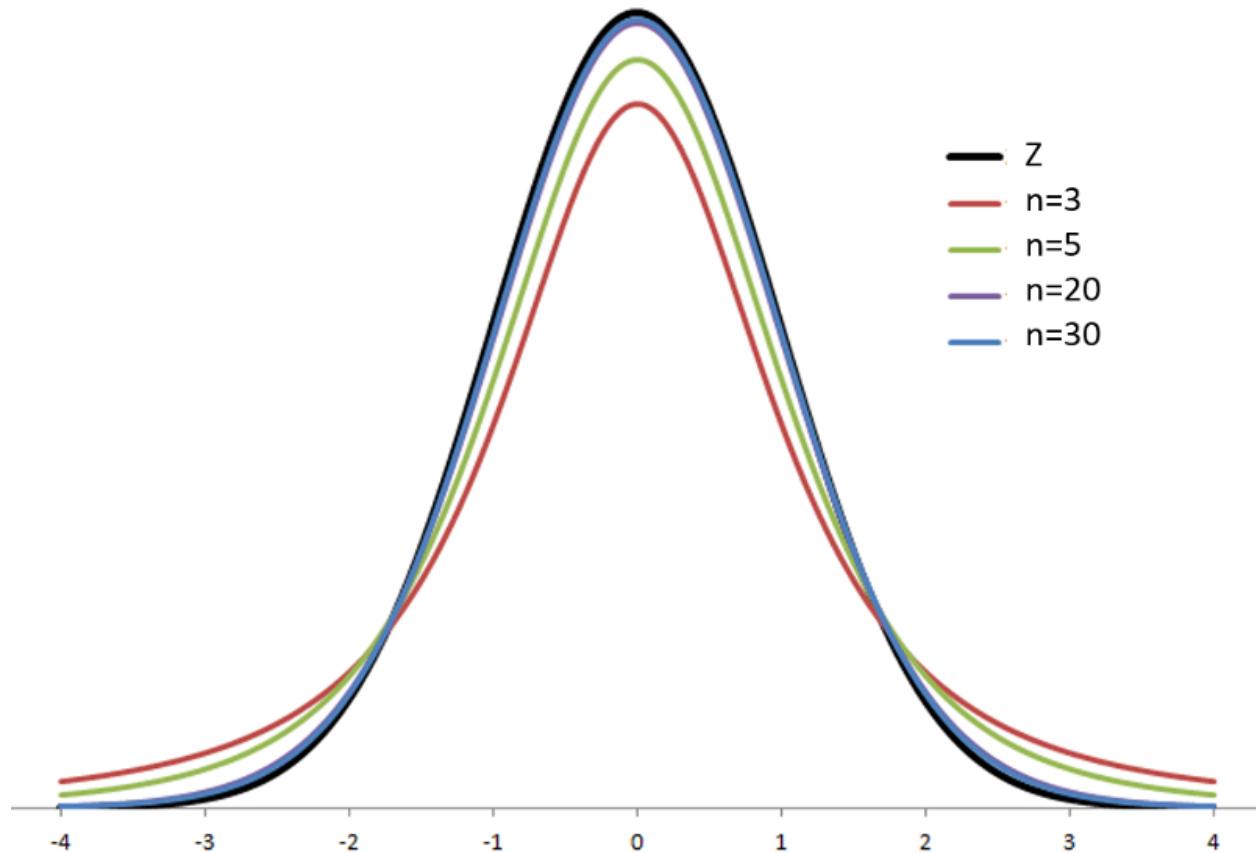
⁸O valor [da estatística z em uma curva normal] para o qual $p = 0.05$, ou 1 em 20, é de 1.96 ou aproximadamente 2; é conveniente pegar esse ponto como um limite ao julgar quando um desvio deve ser considerado significante ou não.

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

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 Beta 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.

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.⁹ 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. $\mu_{bico} = 40\text{mm}$; $\mu_{sonoadicional} = 0\text{horas}$), calculamos as probabilidades de nossa estimativa.

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?

⁹https://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’.

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.

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

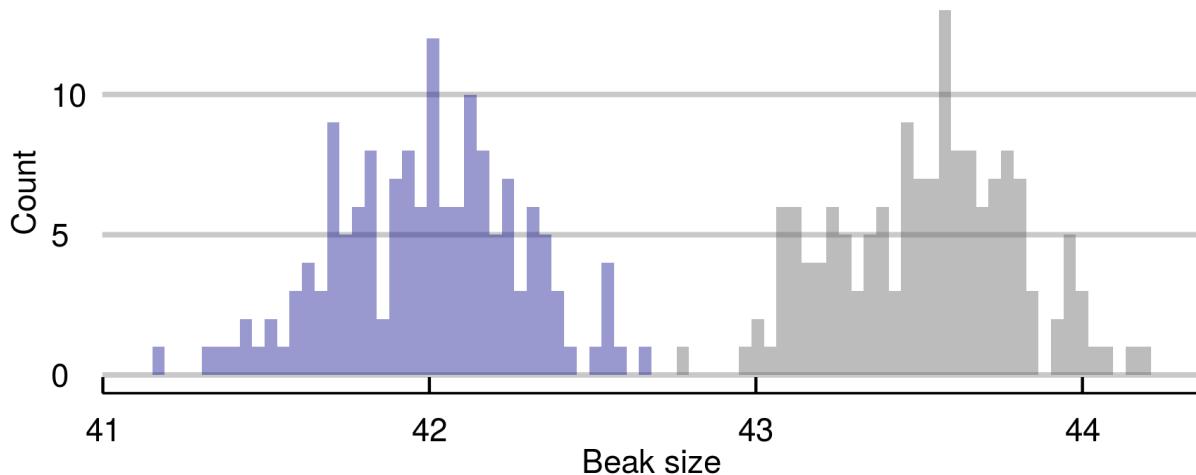
$$\sigma_{pooled} = \sqrt{\frac{(n_1 - 1)\sigma_1^2 + (n_2 - 1)\sigma_2^2}{(n_1 - 1) + (n_2 - 1)}}$$

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	

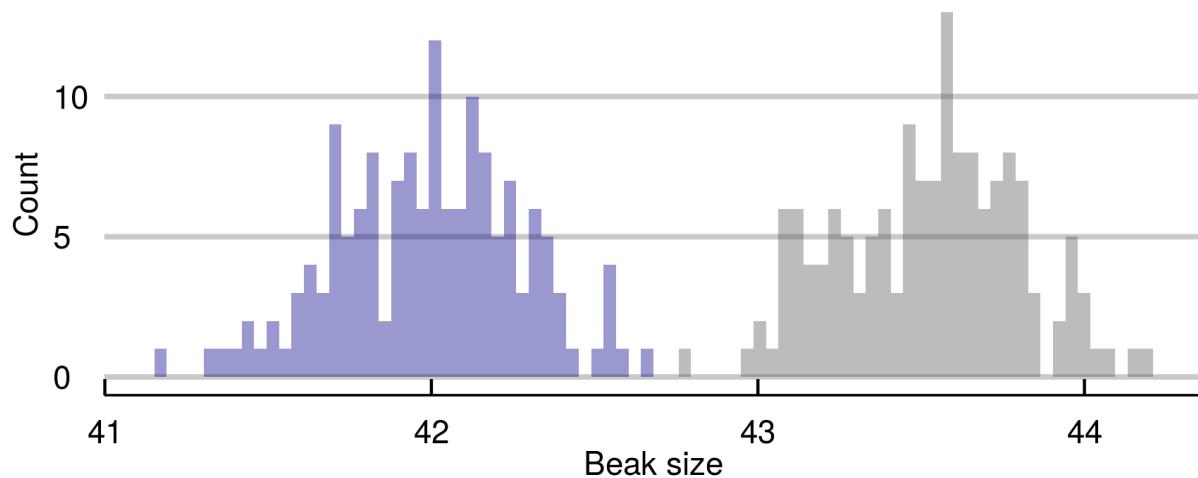
Report example

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	

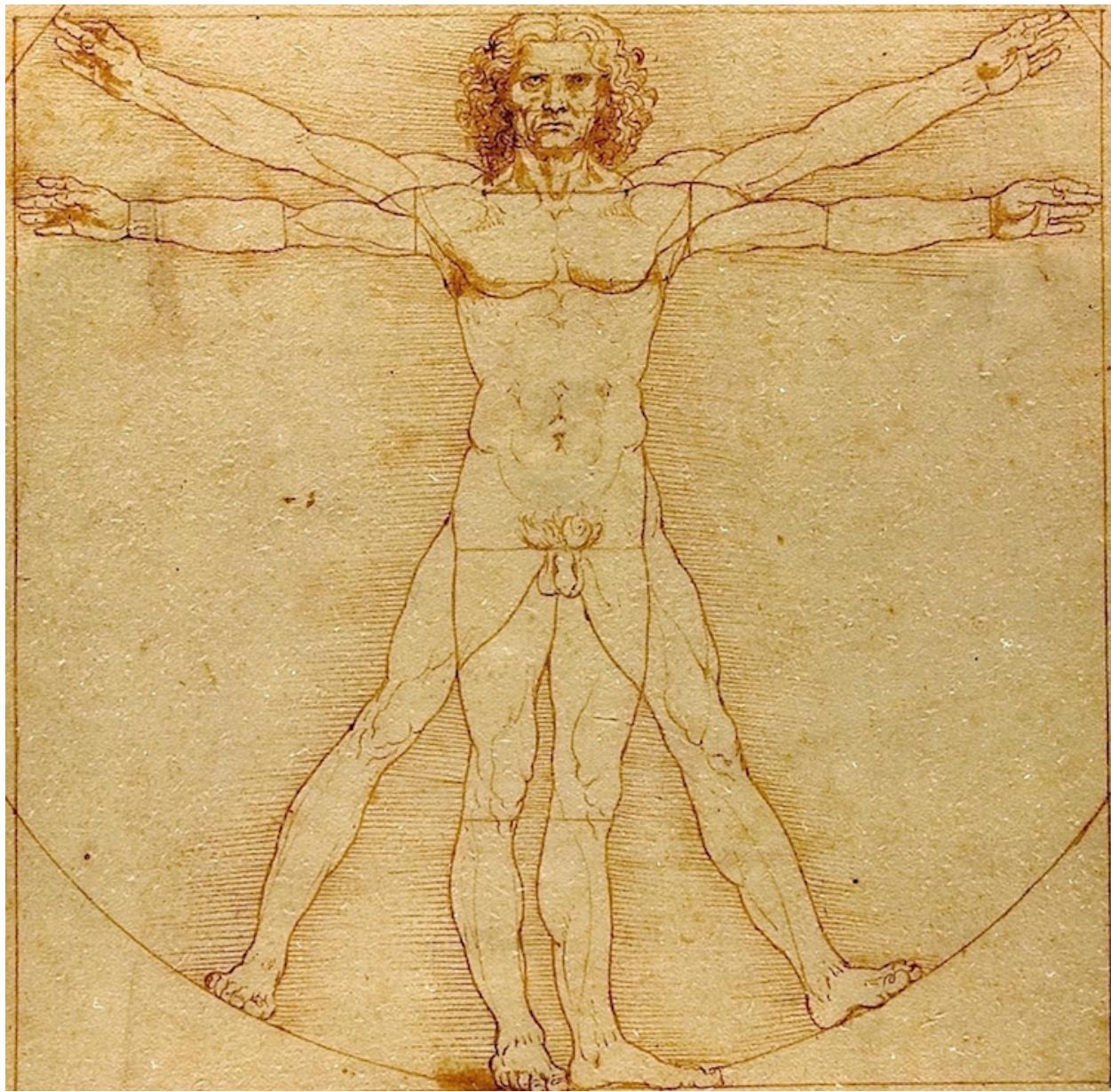
Darwin's Finches

Samples A and B



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, σ/\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:

$$F = ma$$

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.

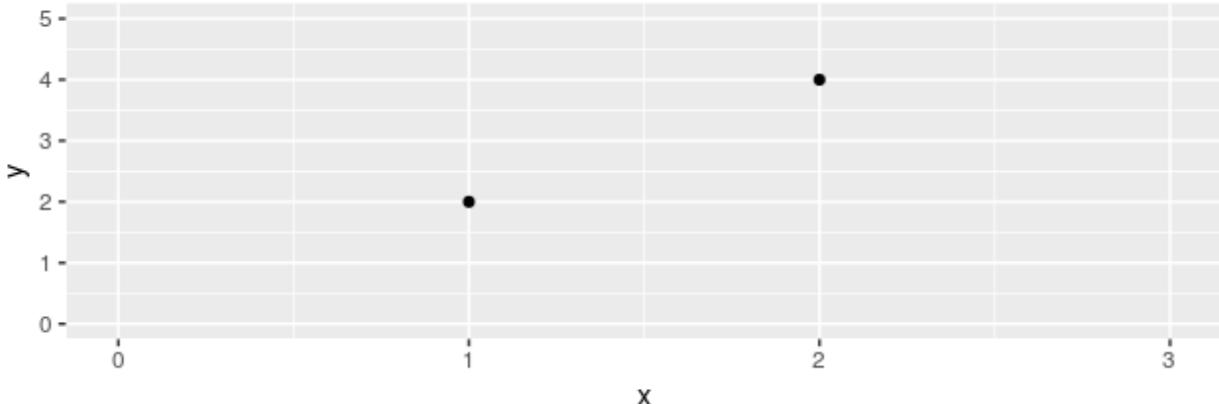
Isto é, valores maiores de massa correspondem a um aumento proporcional em energia. O valor de c^2 expressa essa proporção constante.

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

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.

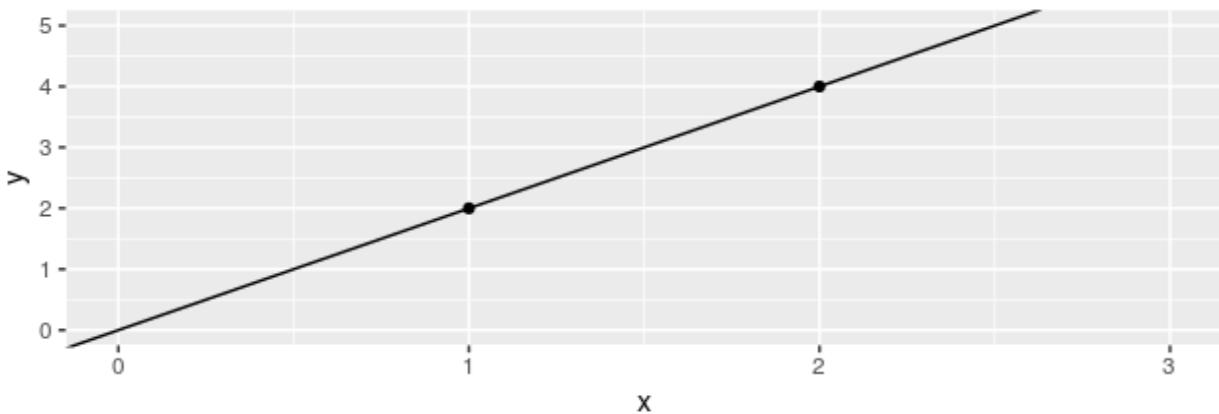
```
>library(ggplot2)
>ggplot() +
  geom_point(mapping=aes(x=1,y=2)) +
  geom_point(mapping=aes(x=2,y=4)) +
  xlim(0,3) + ylim(0,5)
```



$$y = \beta * x$$

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

```
>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 $F = ma$ 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 aleatórios¹⁰.

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

¹⁰A 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.

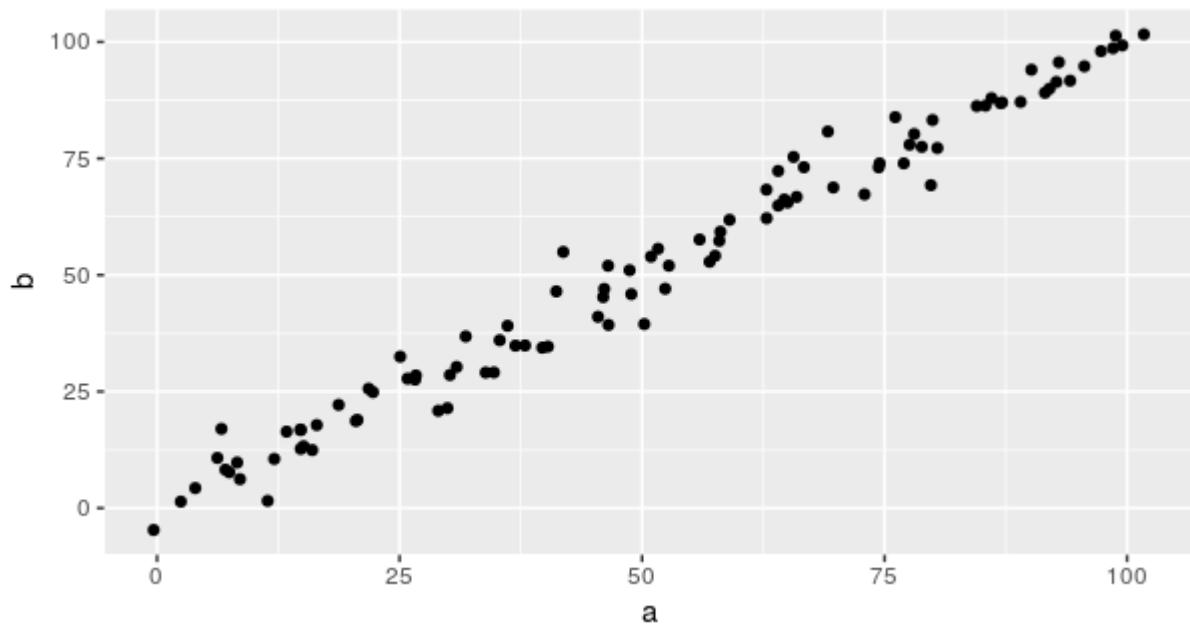
```

>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 garantidamente¹¹ 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.

¹¹Inequação de Cauchy-Schwarz

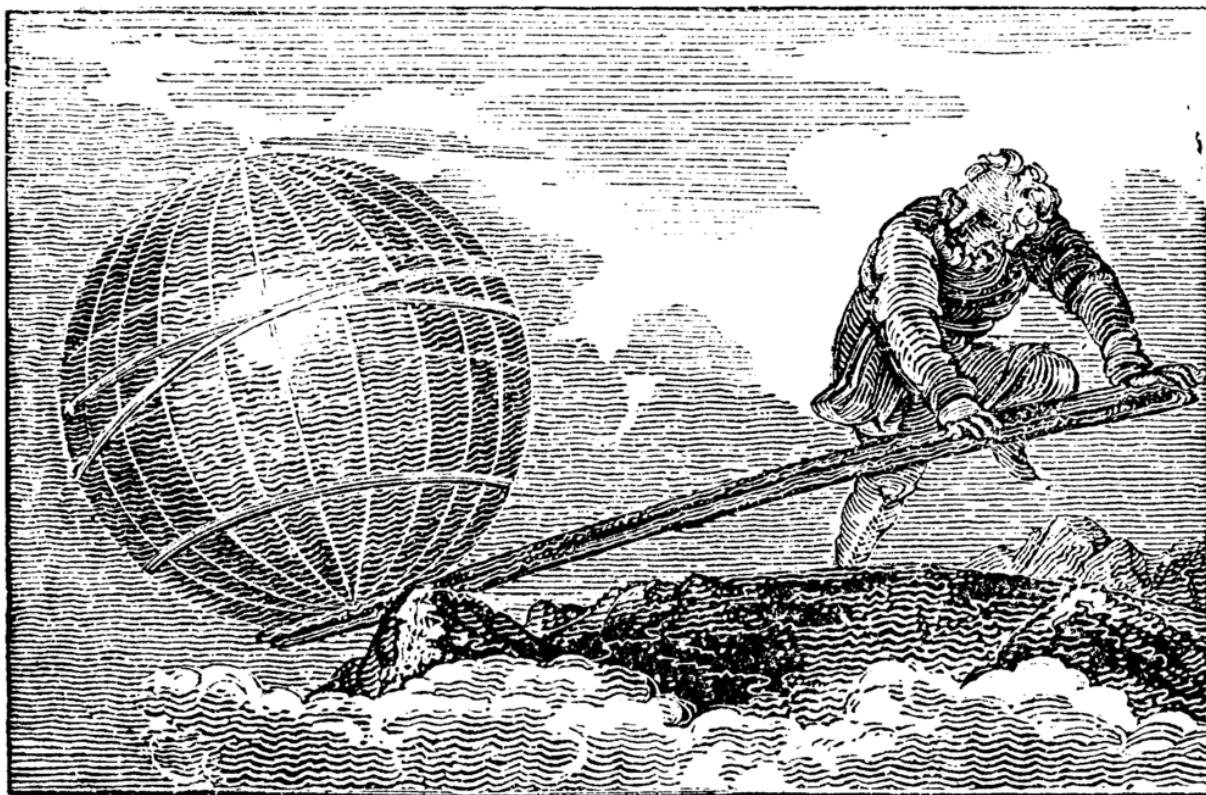


Figure 9: Dê-me um ponto de apoio e eu moverei a Terra

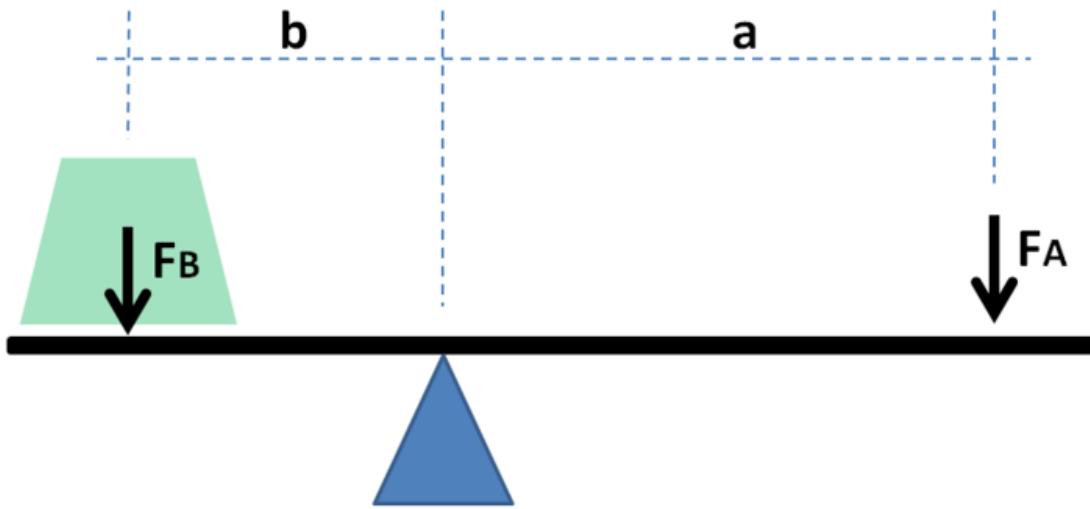
Um breve mergulho na física: Momentos

12

Para adquirir uma intuição sobre o coeficiente, é interessante resgatar o conceito físico, originalmente descoberto 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*.

¹²Pappus 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

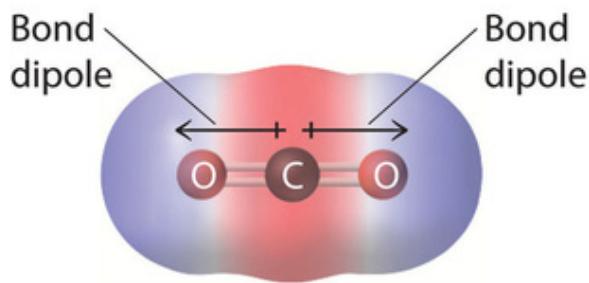
No caso da alavanca, vimos que cada força aplicada sobre o objeto está associada a um momento(torque). Sabemos que a gravidade atua sobre cada pedaço com massa compondo o todo. Podemos então calcular o momento resultante somando os momentos de todos os N pontos. Seja F_i a função descrevendo a força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

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

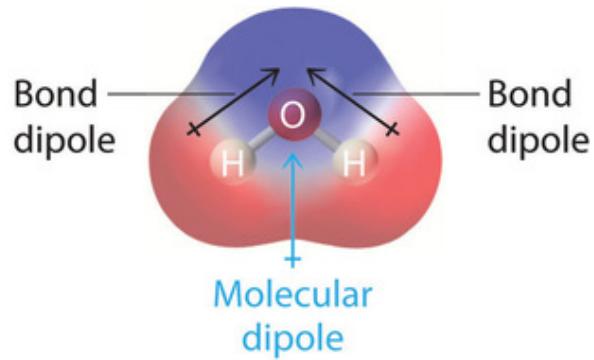
Um sistema, como o pássaro apoiado sobre o dedo, está em equilíbrio quando a soma dos momentos em relação ao ponto fixo é zero. Para cargas elétricas, o sistema é apolar quando o momento é zero. Na figura abaixo, vemos como a molécula de CO_2 é apolar, enquanto a molécula de água é polar:



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.

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 .

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 rotações¹³. O n -ésimo momento $M_n = \sum_{i=1}^N m_i d_i^n$

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 massa total, aqui ele representa a probabilidade total (1).

O **primeiro momento**, correspondente ao centro de massa na mecânica estática, é a **média**.

O **segundo momento**, correspondente ao momento inercial, 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:

$$\mu_0 = \int_{-\infty}^{\infty} d(x, x_0)^0 f(x) dx = \int_{-\infty}^{\infty} f(x) dx = 1$$

¹³Perceba que os termos d_i^2 estariam presentes nas área de um círculo com centro idêntico ao do objeto e raio igual à distância entre o centro: πd^2 . <https://physics.stackexchange.com/a/371165/218274>

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 $E[X]$.

Segundo momento:

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

. Como vimos anteriormente, a soma dos quadrados dos desvios, nossa variância, $\sigma^2 = E[(x - \mu)^2]$.

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

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 antropometria ¹⁴:

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 desvio-padrão maior que a média, a outra também será 1 desvio-padrão 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:

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

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.

¹⁴Francis Galton's account of the invention of correlation. Stephen M. Stigler. Statistical Science. 1989, Vol. 4, No. 2, 73-86.

Assim, podemos ter variáveis altamente correlacionadas positivamente 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, normalizando¹⁵ 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.

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, accesei 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

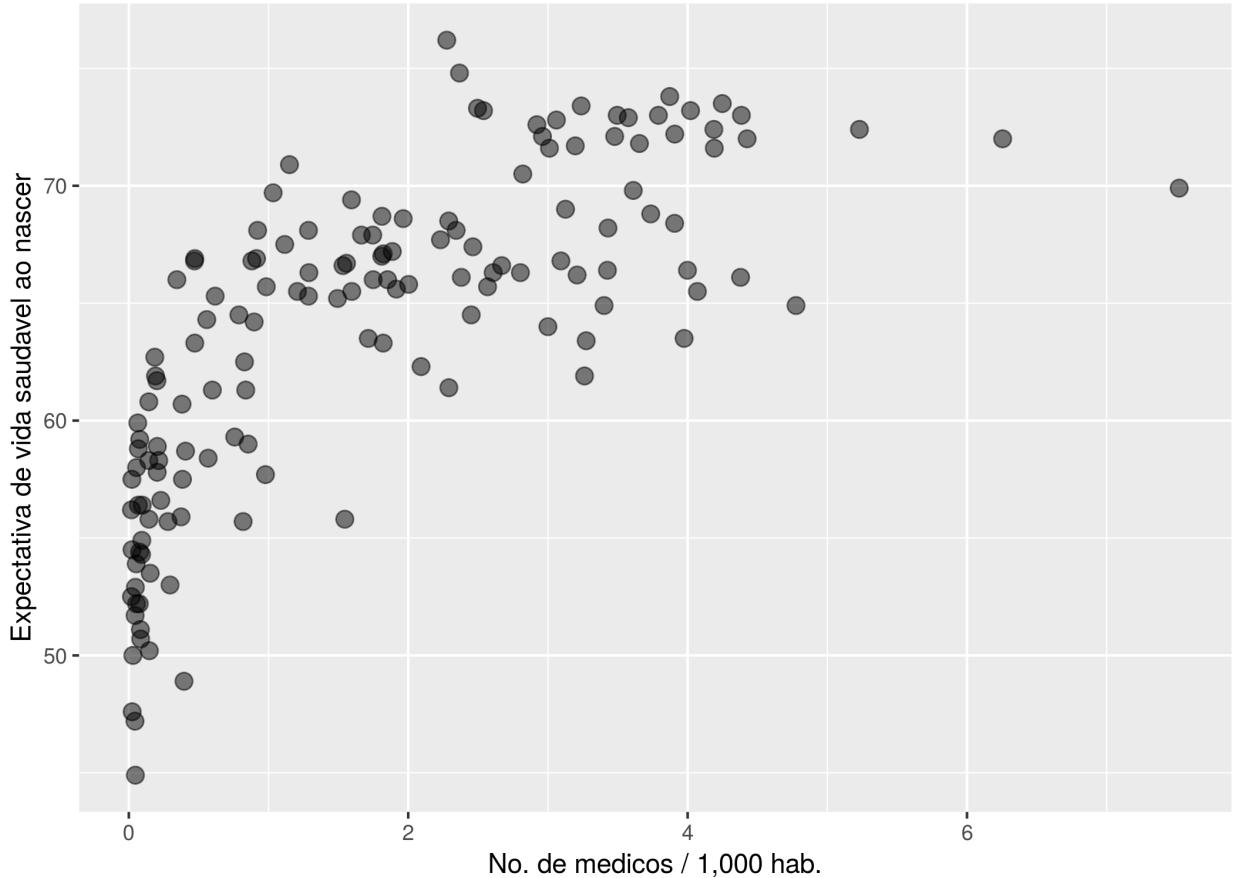
>who_df <- read.csv("data/who_lifeexpect.csv", skip=2)
>who_df$hale <- who_df$X2016
>uni_df <- left_join(worldbank_df[,c("Country", "n_docs")],
                      who_df[,c("Country", "hale")], by="Country")
```

¹⁵Aqui, normalização tem o sentido de ajustar a escala das medidas. Não confundir com transformações para que os dados passem a ter distribuição gaussiana.

```

>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 Nº 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:

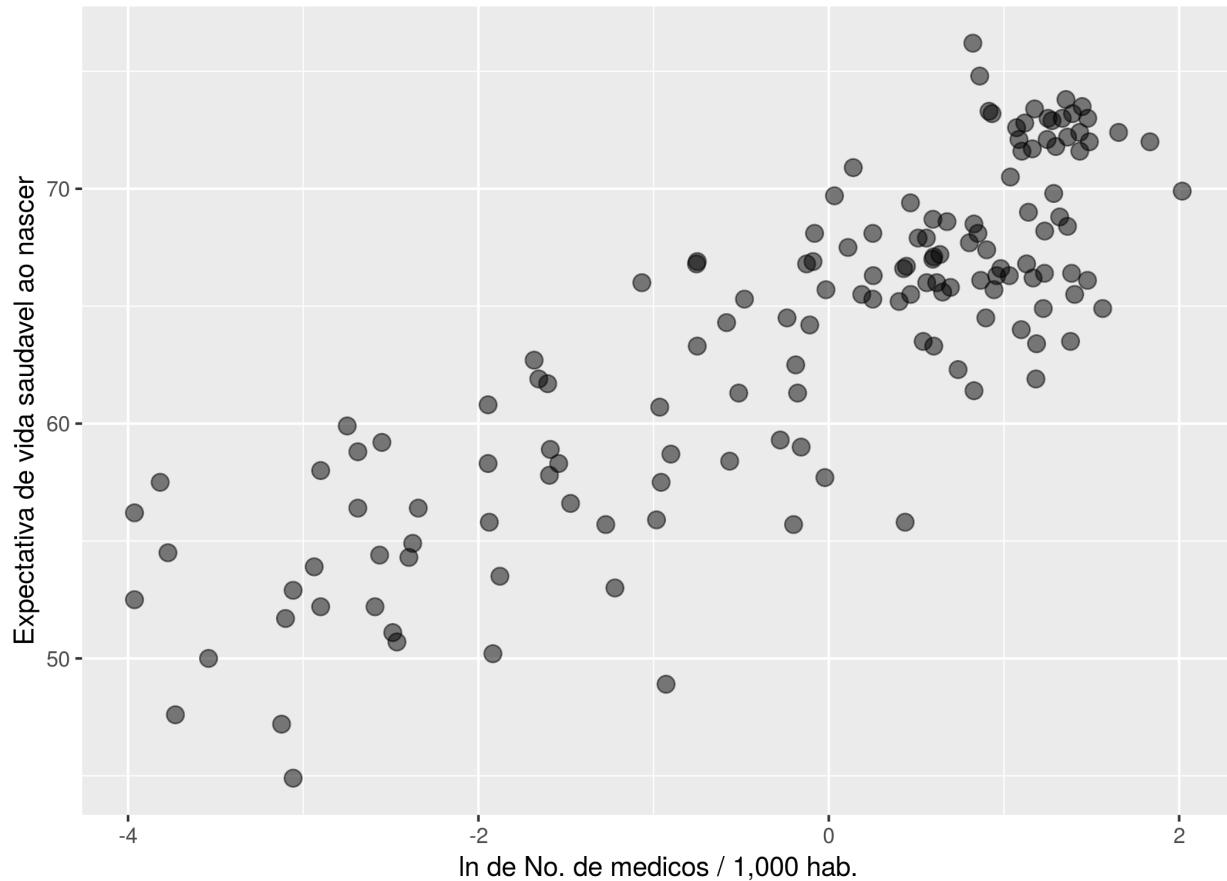
Se $y = \log(x)$, fazemos a substituição $x' = \log(x)$ para obtermos $y = x'$.

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 médicos / 1,000 hab.")+
  ylab("Expectativa de vida saudável 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 ($\rho \sim 0.841$).

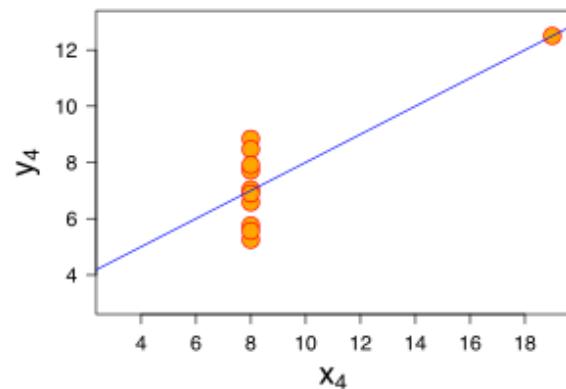
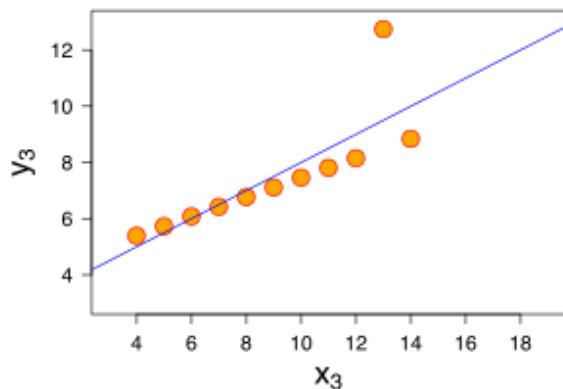
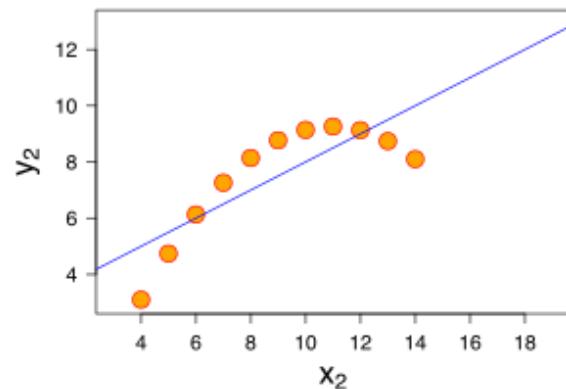
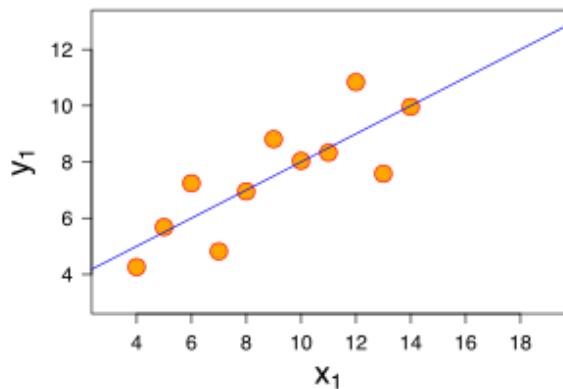
O valor p é baixo ($p < 0.001$) considerando a hipótese nula H_0 de $\rho = 0$. Concluímos então que há uma relação linear significativa de forte magnitude entre o logaritmo do número de médicos e a expectativa de vida dos países em nossa amostra.

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 relação 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.

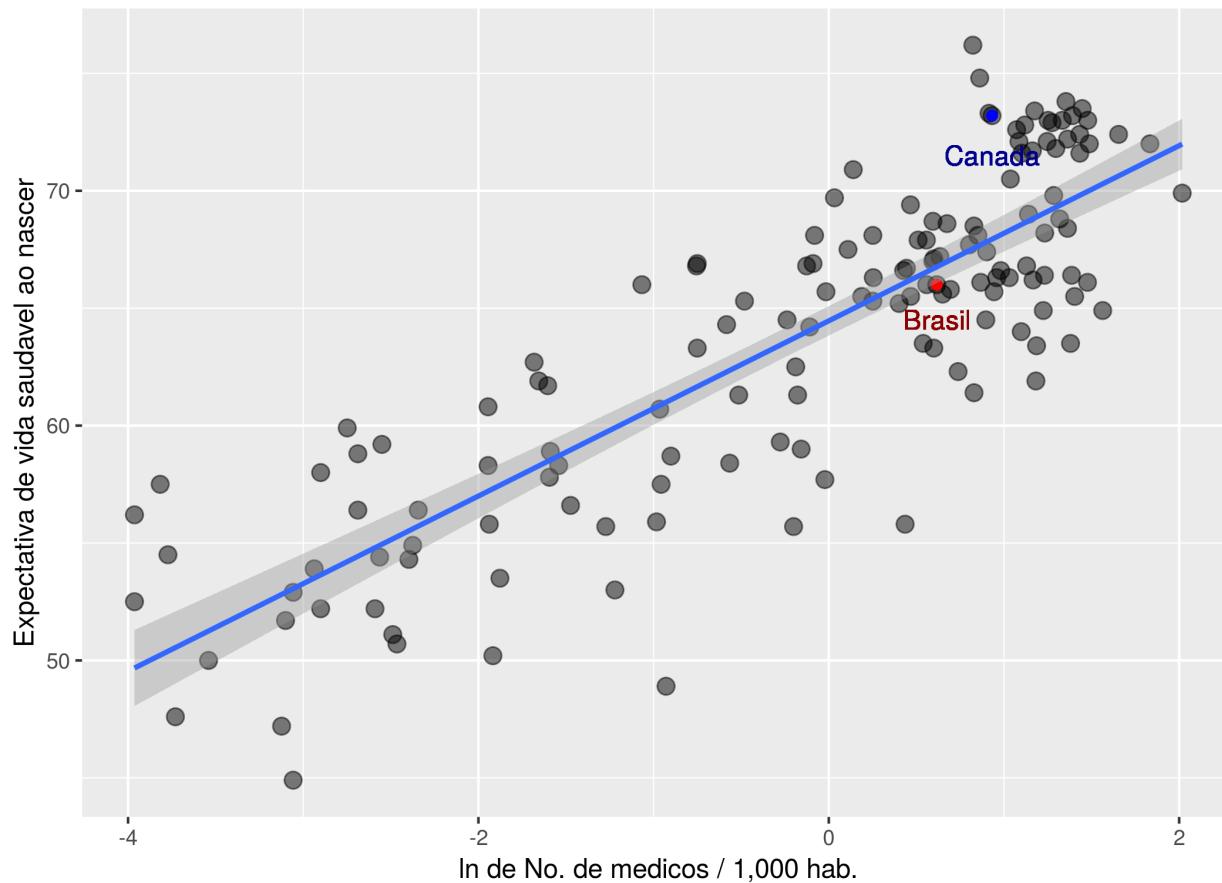


Previsões

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 médicos / 1,000 hab.")+
  ylab("Expectativa de vida saudável ao nascer")
```



Vemos que o Brasil está bastante próximo do esperado para o número de médicos¹⁶. 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. O ministro da saúde de um país em desenvolvimento passa a ter métricas acessíveis para seus objetivos.

¹⁶É 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:

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

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).

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

Podemos adotar como função de verossimilhança (*likelihood function*) para os valores y_i uma distribuição de probabilidades normal cuja média é dada pela reta $\mu_{yi} = \beta_0 + \beta_1 * x_i$.

Assim, a probabilidade de cada valor y_i é dada de acordo com o desvio para o valor previsto pela reta.

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

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

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 logaritmica 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(likelihood) são os mesmos que maximizam a o logaritmo da função de verossimilhança(log-likelihood).

Com algum esforço¹⁷, encontramos fórmulas fechadas:

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

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

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

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 predição dados.

¹⁷Detalhes 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>

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 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 reais. 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("")+
  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. 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:
```

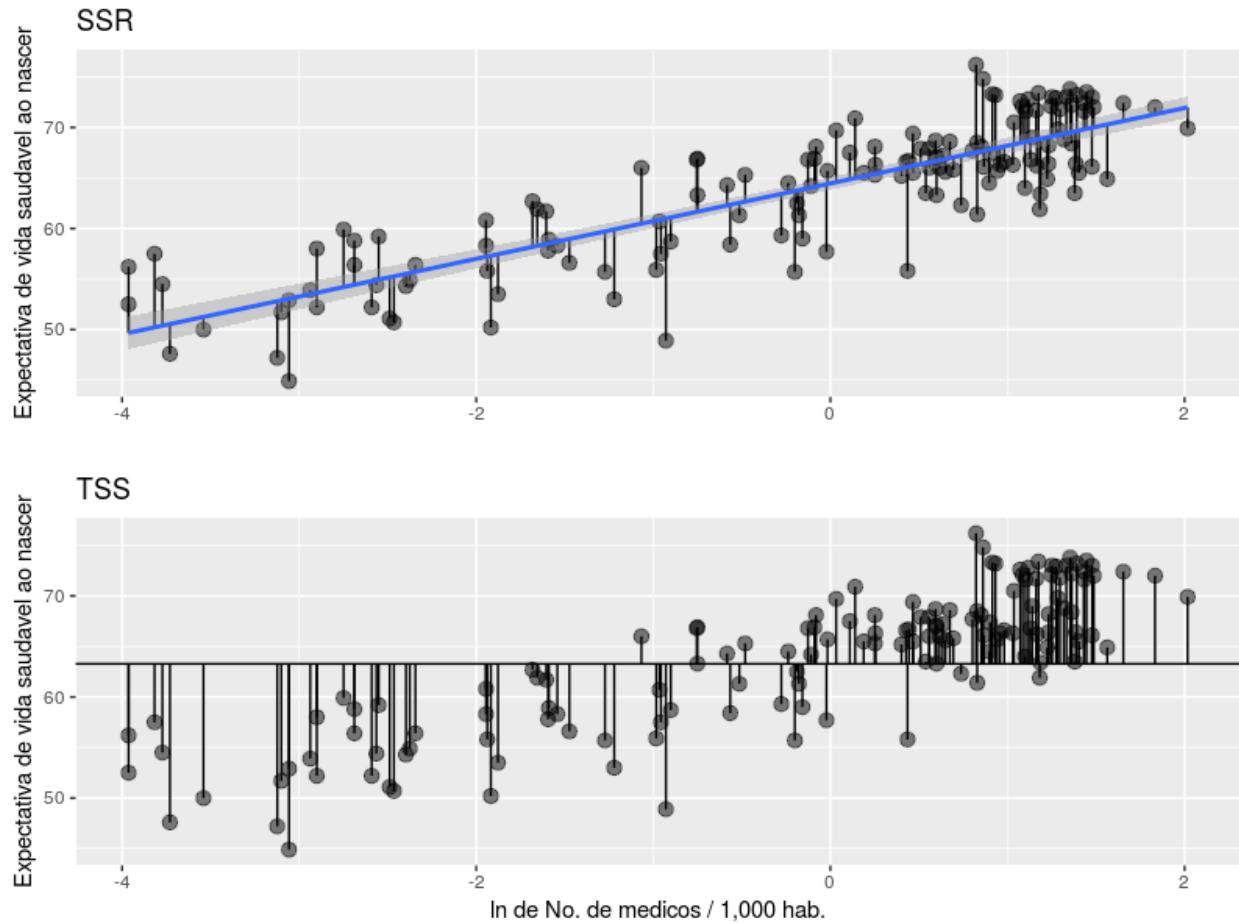


Figure 11: O quadrado da distância entre um ponto e a reta corresponde a um resíduo. Obtemos SSR e TSS somando todos os resíduos nas figuras superior e inferior, respectivamente.

```

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-assumptions.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 Spearman e o teste U de Mann Whitney.

Ranks e o ρ 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.

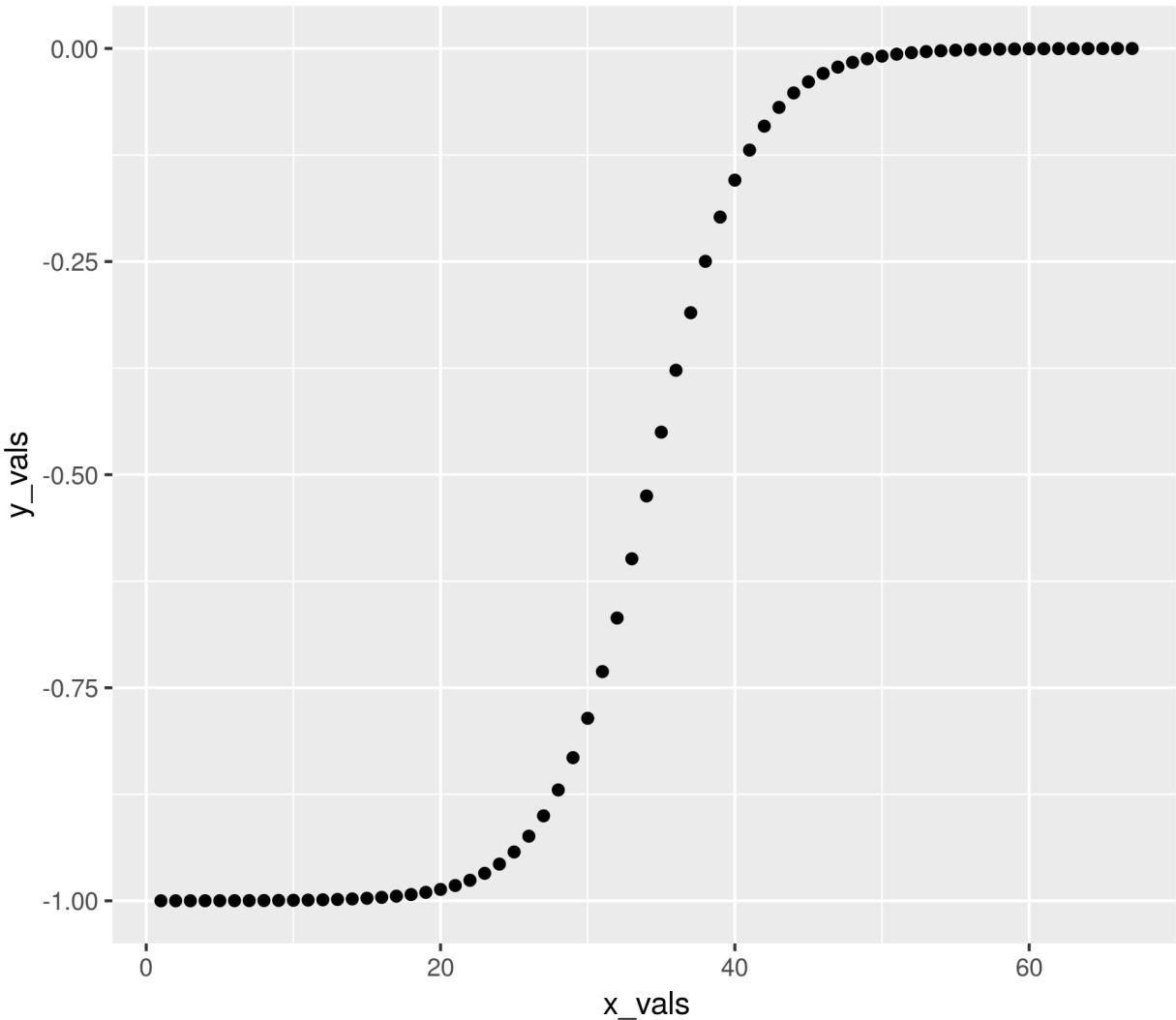
Uma simples solução para esse problema é ranquear os valores. Assim, os itens do conjunto são tratados pela sua posição em relação a outros itens, de forma independente dos valores associados. Exemplo:

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

O *rho* 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:

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



O coeficiente de Pearson é $\rho \sim 0.936^{18}$:

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

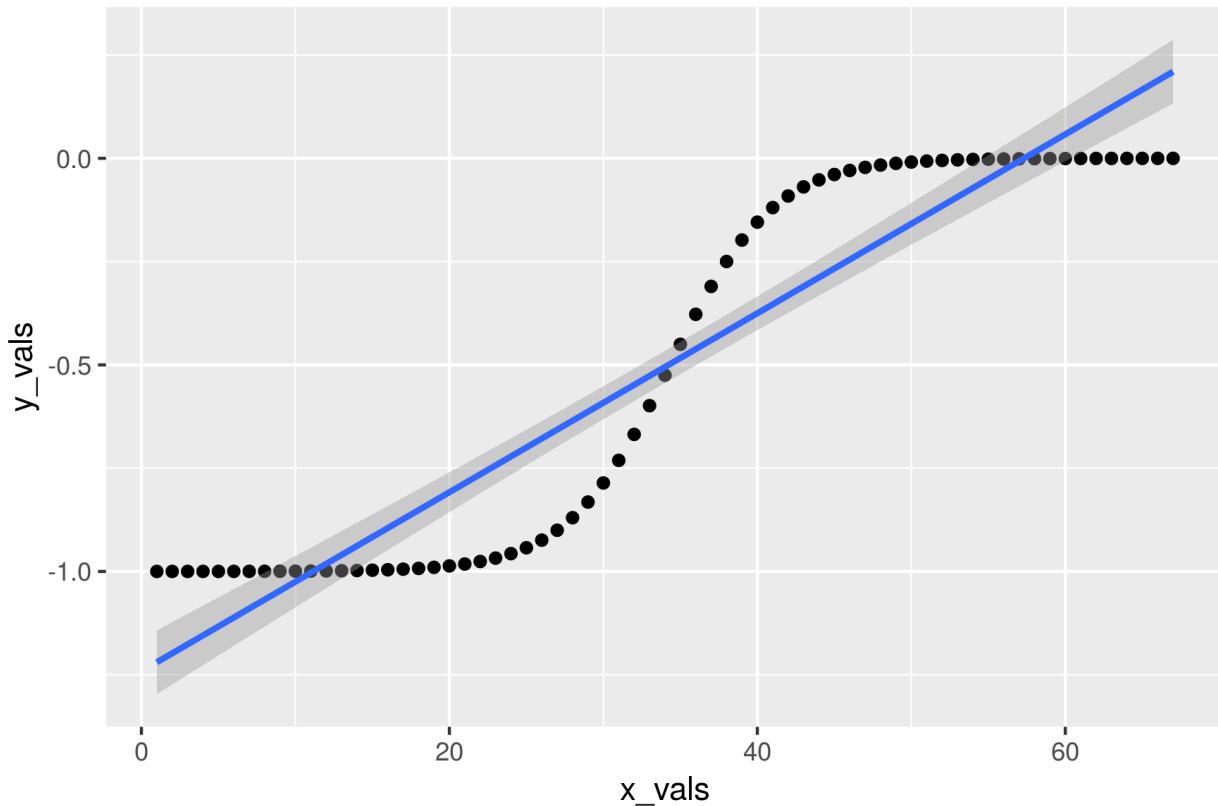
Pearson's product-moment correlation

data: sig_data$y_vals and sig_data$x_vals
t = 21.462, df = 65, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8978064 0.9603803
sample estimates:
      cor
0.9361287

>ggplot(sig_data,aes(x=x_vals,y=y_vals))+
```

¹⁸Como observamos no gráfico, a correlação linear não é tão alta. O coeficiente se aproxima de 1 $\rho \sim 0.936$ 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).

```
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:

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

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.

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

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.

Capítulo 3 : Contexto e Inferência Bayesiana (DRAFT)

@ Intuicao de cenários possíveis vs. Axiomatizacao de Kolmogorov
@ AIC/BIC

Muitos métodos científicos: Feyerabend, Carnap e Quine

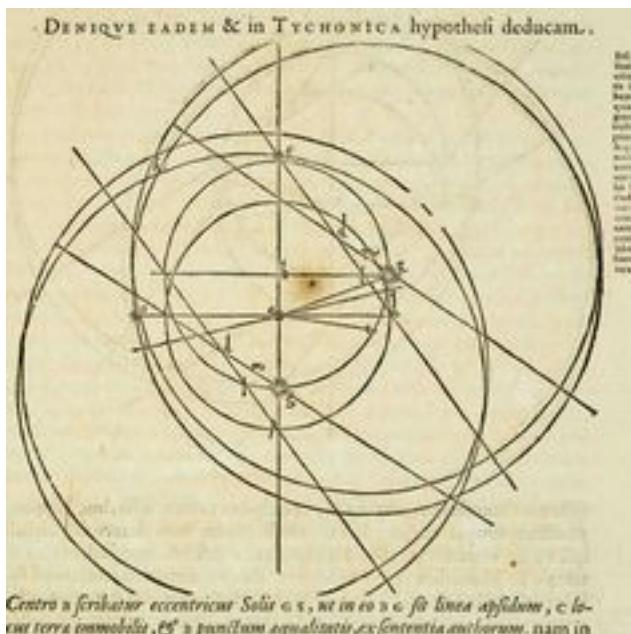
No primeiro capítulo, entramos em contato com o método hipotético-dedutivo e a falseabilidade como critério de demarcação científica. Apesar dominante, esse racional possui vulnerabilidades interessantes. Entenderemos melhor argumentos contrários e propostas alternativas através de três filósofos do século XX.

Feyerabend

Conhecido pela personalidade ímpar e por suas ideias radicais, Paul Feyerabend é notável por conceber o *anarquismo epistêmico*. Em sua obra mais proeminente, *Contra o Método*(1975), argumenta que boa parte dos avanços científicos significativos aconteceram fora do método.

O autor destaca o fato de que as previsões do modelo heliocêntrico proposto por Galileo tinha previsões verificáveis piores que os antecessores. Além disso, fazia uso de abstrações mais complexas. Ainda assim, verificamos posteriormente que o Sol é o objeto mais massivo do sistema solar, ocupando um dos epicentros¹⁹.

Sendo assim, os astrônomos contratados pela igreja à época estavam certos em dizer que o trabalho de Galileo era falso. Ratzinger deveria julgar se a Igreja se retratar quanto ao caso de Galileo (e se comprometer com a verdade) ou manter a postura (e assumir que o melhor que podemos fazer depende dos conhecimentos disponíveis).



Feyerabend argumenta que outros fatores, especialmente idiossincrasias, detalhes biográficos, são responsáveis por mudanças significativas em nosso conhecimento.

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.

¹⁹Hoje, sabemos que há uma interação de muitos corpos celestes. As órbitas são mais complexas, porém o sol continua sendo o atrator mais forte.

Mais que isso, que usar falsificabilidade e o método hipotético-dedutivo teriam nos feito rejeitar o heliocentrismo e outras ideias chave.

De fato, Popper debateu por muito tempo sobre o status científico da teoria da evolução.

Carnap

Outro filósofo que contrapôs Popper de maneira brilhante e pouco conhecida foi Rudolf Carnap, do Círculo de Viena. Em “Testability and Meaning” (1936-7), Carnap argumenta que falsificabilidade não difere verificacionismo. Envolve a testagem das sentenças em si, um problema que outros (https://en.wikipedia.org/wiki/Ludwig_Wittgenstein) também endereçaram.

Dante de resultados inesperados em um experimento, o procedimento padrão do 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.

O cuidado com esses pontos é desejável e desnuda o inevitável calcanhar de Aquiles da falsificabilidade. É impossível obedecer às premissas necessárias para que todo o experimento produza o significado clamado por seus resultados.

Quine

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 níveis de glicemia são mensuráveis usando números e que estes podem ser comparados com valores em 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.

Por fim, estimar livremente os parâmetros de que falamos é muito mais intuitivo que adequar uma ideia ao racional procedural de testagem de hipóteses. Teste de hipóteses: “Quero comparar A e B. As probabilidades de obter minhas observações supondo igualdade entre A e B são baixas o suficiente supor que A e B são, na verdade, diferentes?”

Inferência Bayesiana: “Quero comparar A e B. Como é a distribuição da diferença entre A e B considerando os dados e minhas ideias prévias sobre ela?”

Bayes

Uma abordagem da matemática aplicada que tem se popularizado é o de inferência Bayesiana. Por princípio, partimos de um ponto diferente: não queremos testar hipóteses (ainda). Temos um modelo teórico e incerteza sobre um parâmetro.

Um parâmetro é um símbolo, uma aproximação (para, “resembling”, meter, “measure”) para uma ideia. Em geral, usamos parâmetros para representar algo que intuitivamente se comporta como número (e.g: existem elementos que podem ser ordenados por tamanho).

Em uma prova de QI, a idade do indivíduo é uma medida. O tempo total de prova é uma medida, assim como a quantidade de questões acertadas. O valor de QI é uma parâmetro, um número real estimado a partir das medidas citadas.

Inferência Bayesiana nos ajuda a estimar parâmetros. Ela usa a linguagem das probabilidades. Costumamos

tratar parâmetros como distribuições. Isso é intuitivo para qualquer pessoa. Alguém pergunta “que horas você chega no jantar?”.

A resposta “20:00” na verdade é uma estimativa pontual, porém sabemos que existe a chance de chegarmos

19:55 ou 20:05. Também sabemos que chegar 19:30 é improvável, o que também vale para 20:30.

Trabalhamos com incertezas o tempo inteiro.

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.

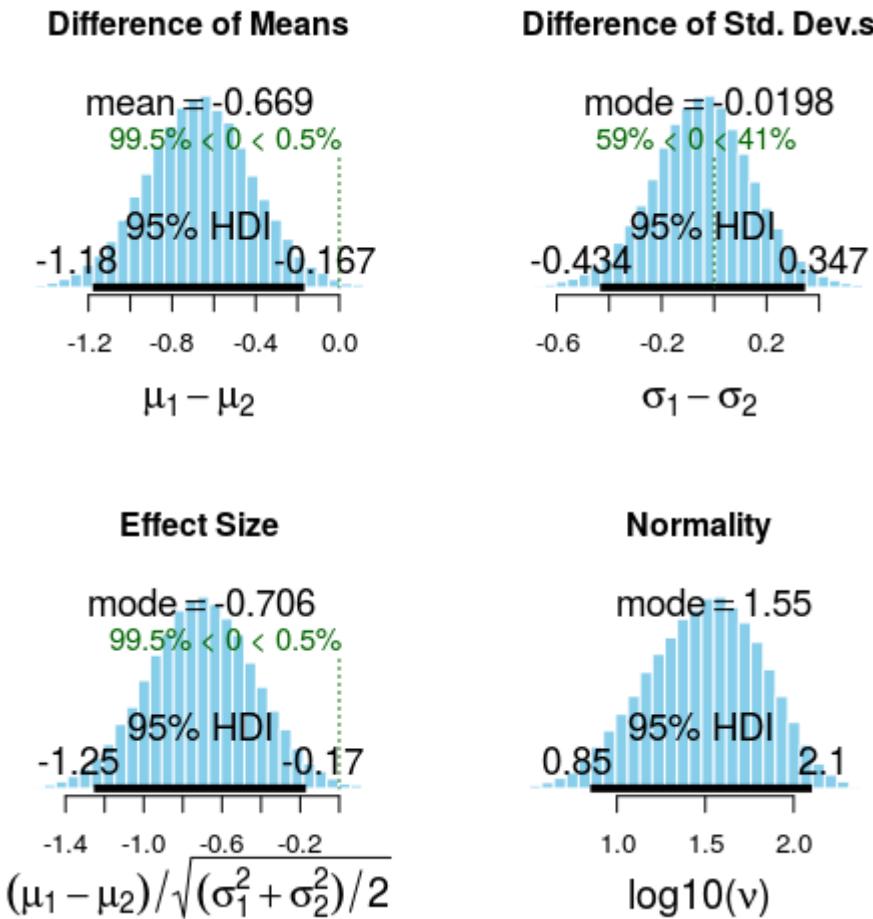
Agora, faremos algo mais simples e intuitivo. Vamos estimar um parâmetro: a diferença entre os grupos. Na verdade, valores prováveis dela. Todas as inferências subsequentes serão derivadas da distribuição produzida por nosso procedimento.

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)
```



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 como é possível estimar essas distribuições. A flexibilidade e o poder dos modelos bayesianos permite lidar com uma série de problemas dificilmente tratáveis sob outra perspectiva. 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.

Então, seguindo o exercício anterior, precisamos especificar que consideramos as duas amostras vindo de distribuições t com médias μ_1 e μ_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)
```

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

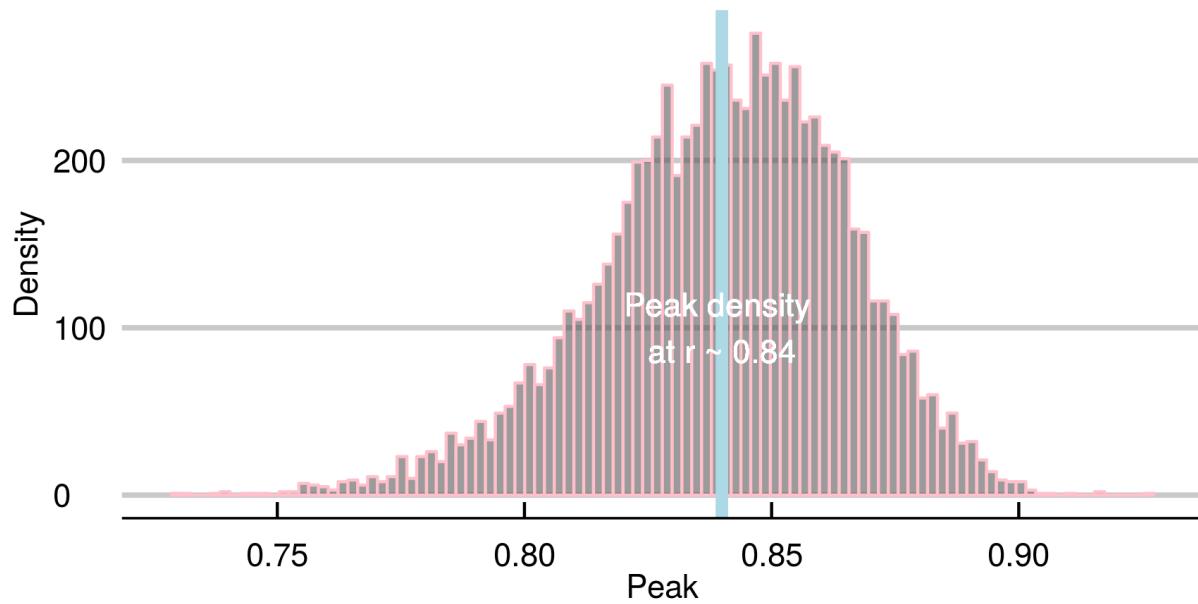
P(H): PRIOR PROBABILITY THAT H IS TRUE

P(x): PRIOR PROBABILITY OF OBSERVING X

P(C): PROBABILITY THAT YOU'RE USING
BAYESIAN STATISTICS CORRECTLY

Figure 12: <https://xkcd.com/2059/>

```
>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)
```

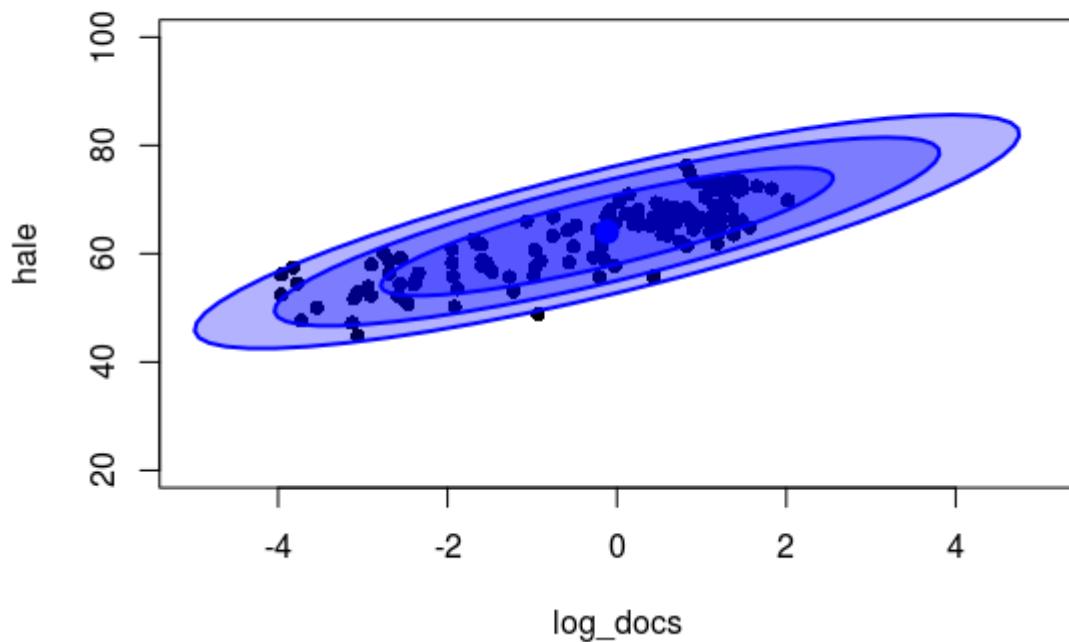


@Correlacao

```
>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



Figure 13: AlphaGo: Monte-Carlo tree search, deep neural networks e reinforcement learning jogando contra si

DRAFT de Capítulo 4 : - textos base retirados do blog

[^30]

Em 28 de Abril de 2016, o então CEO da Google (Sundar Pichai) publicou na carta anual dos fundadores (<https://blog.google/inside-google/alphabet/this-years-founders-letter/>) sua visão de futuro, compartilhando com acionistas e com o público os objetivos da companhia. O termo ‘*machine learning*’ foi usado 8 vezes, inclusive como responsável pelas principais inovações apresentadas pela companhia, como Google Maps (geonavegação) e Google Photos (visão computacional).

“(...) our long-term investment in machine learning and AI. (...) It’s what has allowed us to build products that get better over time, making them increasingly useful and helpful.” Sundar Pichai, CEO, Google, 2016

Em março do mesmo ano, o Google DeepMind AlphaGo tornou-se o primeiro programa de computador a vencer um mestre de Go. O feito é difícil por tratar-se de um jogo quase impossível de ser totalmente computado.

Existem $2,08 \times 10^{170}$ maneiras válidas de dispor as peças no tabuleiro. Vale lembrar que o número de átomos no universo observável é de módicos 10^{80} .

Inteligência artificial possui aplicações crescentes nas áreas de finanças, saúde, indústria, aviação e segurança.

Classificadores lineares e visão computacional

Neste capítulo, vamos entender como usar classificadores para visão computacional. Uma ferramenta simples é o classificador linear. Como veremos, guarda semelhança com os métodos de regressão mostrados antes.



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 [100 x 1] com valores entre 0 e 255 em cada elemento.

Podemos 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 precisam distinguir quatro tipos de animais a partir da imagem: pássaro, tartaruga, golfinho ou peixe.

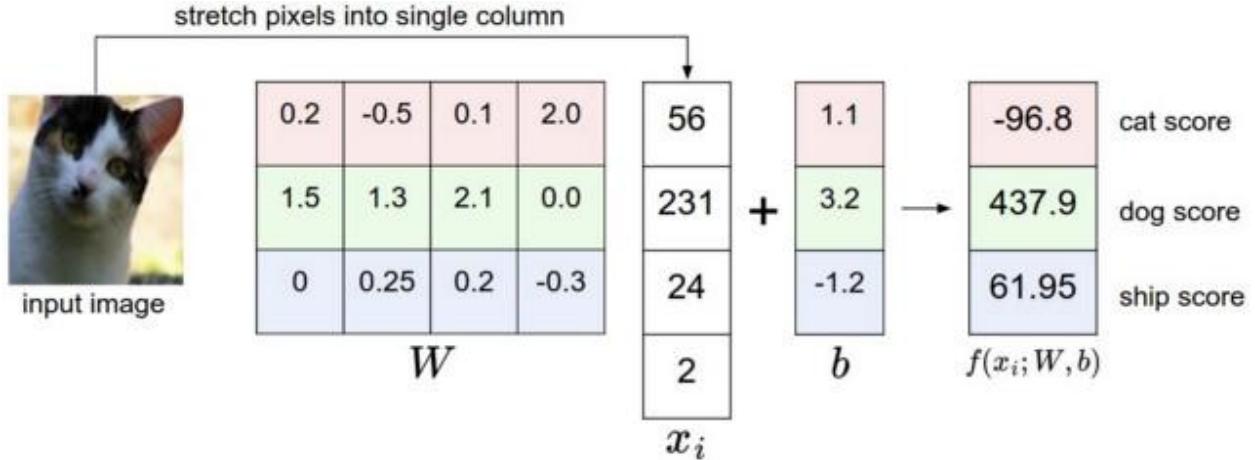


Figure 14: Retirado de <http://cs231n.github.io/linear-classify/>.

O classificador linear atribui scores para cada uma das 4 classes aplicando seus pesos em cada pixel da imagem.

Matematicamente, é uma multiplicação da matriz de valores da imagem x_i , de dimensão $[100 \times 1]$ por uma matriz $W_{[100 \times 4]}$ que traz pesos (weights) estimados para cada pixel para cada classe.

O resultado dessa multiplicação de matrizes são scores para cada classe K.

Vamos considerar que nossa ordem de rótulos é:

[pássaro, tartaruga, golfinho, peixe]

Em R:

```
#Iniciando pesos com base em distribuição normal
#Dividi os valores por 100 para reduzir a magnitude dos numeros
>my.weights <- rnorm(400)/100
#Le pesos como matriz [100x4]
>w <- matrix(my.weights, 100, 4)
#Multiplicacao usando o operador %*%
>as.vector(x)%*%w
#Resultado
[,1]      [,2]      [,3]      [,4]
[1,] 20.95787 22.10932 19.08313 -30.33214
```

O classificador retorna um valor de score para cada classe. A interpretação desses valores pode variar, mas vamos pensar, por enquanto, que nosso objetivo é que o maior score seja o da classe correta.

Em nosso exemplo, o output: [20.95787, 22.10932, 19.08313, -30.33214].

Entre os valores, o maior entre os quatro foi o segundo (22.10932), sugerindo o rótulo de tartaruga.

O processo de aprendizado, consiste em expor o classificador a diversos exemplos x até que ele ajuste os pesos em W de maneira a retornar o maior score para a classe K correta.

A imagem abaixo traz um diagrama dessa multiplicação.

Essa imagem traz um diagrama do processamento dos dados que resultam nos scores finais para classes K: cat,dog,ship. Nesse caso, a classificação seria cachorro (437.9)

$$K(x, y) = x^T y$$

y indica os pesos (antes chamados de W nesse texto) e x^T a matriz com dados do exemplo. O uso de transposição ou não depende da forma escolhida para a matriz $[nx1]$ ou $[1xn]$.

Essa função é chamada kernel function e, no em nosso caso, é linear.

Para levar em conta uma constante b , usaremos um truque: ao adicionar o valor 1 ao final da imagem, a multiplicação dos pesos associados será constante. Assim podemos incluir estimativas do valor de b como

pesos em W , que, quando multiplicados por 1, serão sempre as mesmas constantes.

```
#Adiciona valor 1 ao vetor e armazena em x.vec. Agora temos 101 elementos
>x.vec <- c(as.vector(x),1)
#Inicia pesos incluindo 4 valores extras para (uma constante para cada score)
>my.weights<- rnorm(404)/100
#Leitura como matriz w de dimensao 101x4
>w <- matrix(my.weights,101,4)
#Multiplicacao: (pixels da imagem + 1)* (Pesos do classificador)
>x.vec%*%w
#Resultado
[,1]      [,2]      [,3]      [,4]
[1,] 8.620293 10.08648 -4.423656 -7.804998
```

Agora, nossos scores aleatórios são: [8.620293, 10.08648, -4.423656, -7.804998] Agora indicando tartaruga (2a posição) com maior score. Como transformar esses pesos em valores úteis?

Inicialmente, estabelecemos pesos aleatórios a partir de uma distribuição normal.

Nosso objetivo agora é observar as respostas corretas em várias imagens e alterar os valores de W para que os scores maiores sejam os das classes corretas.

Esse aprendizado se dá através de uma função de perda L .

Support Vector Machine (SVM)

A função de perda (*loss function*) quantifica o quanto distante estamos dos pesos desejados. O score desejado deve ser maior que os outros. Sendo s_j o score atribuído à classe correta e s_i o score atribuído à i -ésima classe errada:

A função de perda retorna 0 caso a classe correta tenha score maior $s_i - s_j < 0$ ou o valor da diferença caso contrário, $s_i - s_j > 0$.

$$\max(0, s_i - s_j)$$

A perda total é a soma dos erros para cada classe. Adicionamos ainda uma margem determinada delta (Δ). O score correto deve ser maior que os outros com uma diferença mínima igual a delta. Então, a perda para a observação x é:

$$L_x = \sum_{j \neq i} \max(0, s_i - s_j + \Delta)$$

A soma L (loss) vai acumular um valor de erro se o score correto não estiver distante o suficiente (Δ) dos deltas incorretos.

Implementando em R:

```
loss <- function(x,w,cor.class){
  #Determina delta = 2, a distancia minima entre o maior score e os outros
  delta <- 1
  #Calcula scores multiplicando valores da imagem por pesos W
  scores <- x.vec%*%w
  #Score da classe correta fornecida pelo argumento da funcao
  correct.class.sc <- scores[cor.class]
  #Obtem numero de classes
  dimensions.class <- length(scores)
  #Perda inicial = 0
  cur.loss <- 0
```

```

#Loop para calcular a soma dos valores de max(0,-formula SVM)
#A função max está nas funções básicas (Base Package) do R
for (i in 1:dimensions.class){
  if (i == cor.class){next}
  cur.loss <- cur.loss + max(0,scores[i] - correct.class.sc + delta)
}
#Retorna valor total da perda
return(cur.loss)

```

E podemos testar os scores para cada classe invocando a função loss(imagem,pesos,classe_correta):

```

> loss(x,w,1)
[1] 2.466183
> loss(x,w,2)
[1] 0
> loss(x,w,3)
[1] 29.55408
> loss(x,w,4)
[1] 40.69811

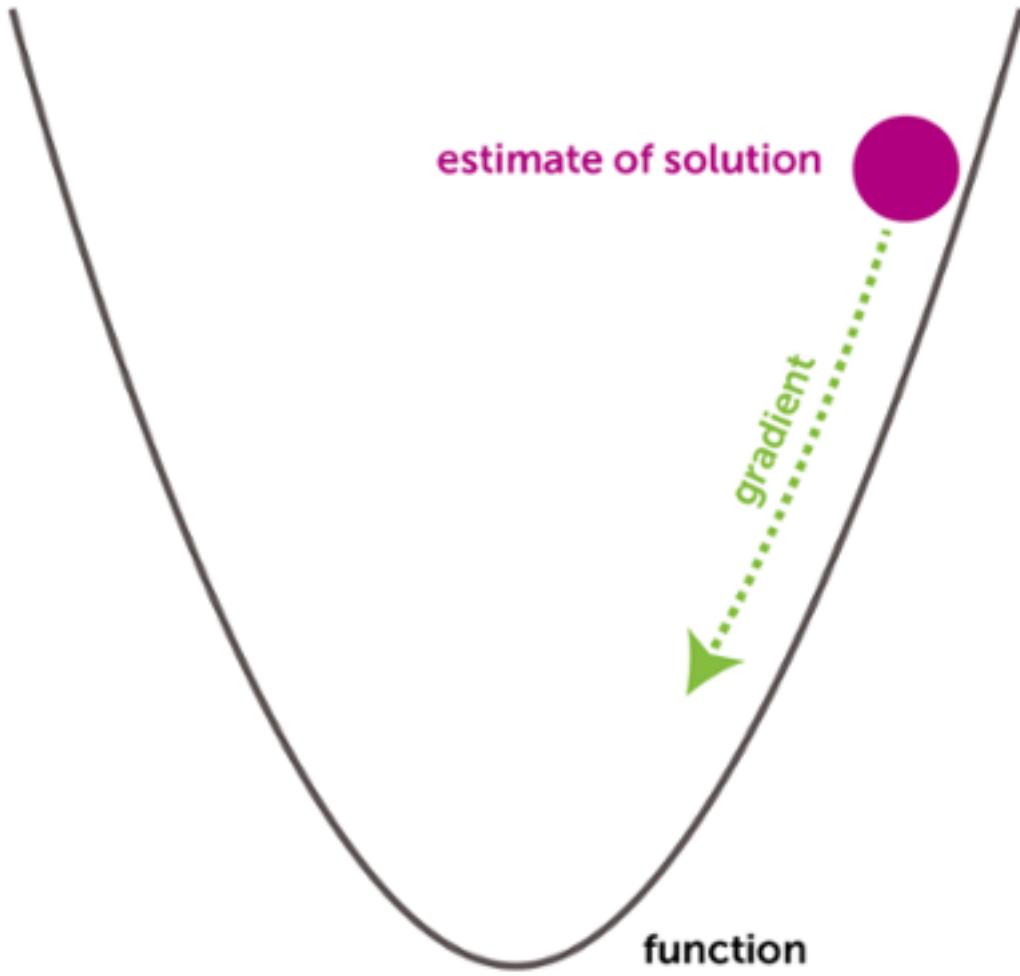
```

Notem que se informamos que a classe correta é a 2, que tinha o maior score, a função retorna 0. Isto é, o classificador apontou o maior score com a margem adequada e não há perda.

Se apontarmos que a classe correta é uma das outras (1,3 ou 4), a função retorna a perda correspondente.

Agora, o objetivo é encontrar pesos em W que minimizem L para todos os exemplos. Isso pode ser feito analisando o gradiente de L com cálculo diferencial, ou de maneira numérica, através de algoritmos recursivos (veremos o uso do estimador Markov Chain Monte Carlo).

Na prática, estamos mexendo nos parâmetros W de forma a direcionar nossa função de perda L aos menores valores, descendo a montanha.



@ Gradiente <http://cs231n.github.io/optimization-1/>

Com os novos parâmetros W , podemos aplicar o classificador na imagem inédita $x'_{[10 \times 10]}$, obter os scores para predizer a classe dela, assim como uma nova função de perda.

Como dá para notar, mesmo num exemplo simplificado, treinar o classificador implica muitas computações de matrizes n-dimensionais. Por isso, precisamos de bastante poder computacional e *machine learning* só ganhou atenção recentemente, com o avanço do hardware apropriado.

Kernels

A função do núcleo (kernel) descrita anteriormente é linear. Os valores são multiplicados pela matriz de valores W , somados a b e resultam em scores. Diferentes kernels podem ser usados, como o polinomial:

$$K(x, y) = (x'y + c)^d$$

As funções kernel retornam sempre um produto interno entre dois pontos no espaço adequado. Além do linear e do polinomial, temos outros: Gaussian Radial Basis Function (RBF) é um tipo que costuma apresentar bom desempenho.



Figure 15: Linus Torvalds (Linux) e Richard Stallman (GNU). Dois caras legais.

$$k(x, x') = e^{(-\sigma ||x - x'||^2)}$$

Agora, sabemos examinar um conjunto de imagens rotuladas, criar um classificador linear e treiná-lo (ajustar pesos W minimizando a função de perda L) para retornar um maior score nas classificações corretas.

Referências: CS231n - Stanford University: Convolutional Neural Networks for Visual Recognition Karatzoglou et al. Support Vector Machines in R. Journal of Statistical Software. April 2006, Volume 15, Issue 9.

SVM - Aplicações

Parte 2 - Aplicações

Recuperando o exemplo anterior, temos um conjunto de dados (ex: uma imagem) e um classificador com pesos para esses dados. O classificador realiza operações (definidas pela função kernel) entre seus pesos e os dados para retornar scores.

O maior dos scores deve apontar a classe correta (mecanismo de voto).

Vimos que existem diversas formas de encontrar valores para os pesos que classifiquem corretamente as imagens (minimizem as perdas).

Vimos como funcionam as operações com um kernel linear.

Apresentamos benchmarks para tempos das funções dos pacotes **kernlab**, **e1071**, **klaR** e **svmpath** em diferentes datasets para uma mesma tarefa.

Os pacotes kernlab e e1071 parecem ser os mais rápidos (e1071 ligeramente na frente). A e1071 é um interface para o libsvm, library premiada (IJCNN 2001 Challenge) e escrita em C++, o que garante a melhor performance. O problema é que não há flexibilidade para mudar muito o kernel. Já o kernlab traz maior flexibilidade, mas seleção de modelos é limitada. Recomendo brincar com as quatro libs. Já que não vamos mexer no kernel, vamos com a função `svm()` do pacote e1071 em nome do minimalismo.

Dados

Vamos usar o nosso conhecido *iris*.

	ksvm() (kernlab)	svm() (e1071)	svmlight() (klaR)	svmpath() (svmpath)
spam	18.50	17.90	34.80	34.00
musk	1.40	1.30	4.65	13.80
Vowel	1.30	* 0.30	21.46	NA
DNA	22.40	23.30	116.30	NA
BreastCancer	0.47	0.36	1.32	11.55
BostonHousing	0.72	0.41	92.30	NA

Figure 16: Benchmarks (Tempo em segundos). Support Vector Machines in R (Alexandros Karatzoglou, David Meyer, Kurt Hornik)

```
>library(e1071)
# O pacote caret facilita particionamento dos dados particionados
>library(caret)
>set.seed(2600)
# Usa funcao createDataPartition do caret para gerar vetor com vasos sorteados na proporcao 4/5
# A frequencia relativa de rótulos (Species) fica mantida
>iris.tr.vec <- createDataPartition(y=iris$Species,p = 4/5,list=F)
# Dataset de treino (4/5 da amostra)
>iris.tr <- iris[iris.tr.vec,]
# Dataset de teste (1/4 restantes)
>iris.ts <- iris[-iris.tr.vec,]
```

Agora, temos um banco com 80% (4/5) dos dados para treinar a SVM e outro com 20% para testar.

SVM com e1071

Vamos usar a função svm, especificando uma fórmula (“Species ~ .” significa Species como variável de classificação e as outras como input), o banco de dados e um custo (Constate C; falaremos mais sobre ela depois).

```
>svm.iris <- svm(Species ~ ., data=iris.tr, cost=100, kernel="linear")
```

Agora, fazemos as previsões:

```
>svm.pred <- predict(svm.iris,iris.ts)
# Dispoe predicoes e valores no dataset de teste em uma tabela
>agree.tab <- table(pred=svm.pred,true=iris.ts$Species)
>agree.tab
agree.tab
      true
pred      setosa versicolor virginica
setosa      10        0        0
versicolor     0        9        3
virginica      0        1        7
# Usando classAgreement do proprio pacote e1071
# Calculamos:
```

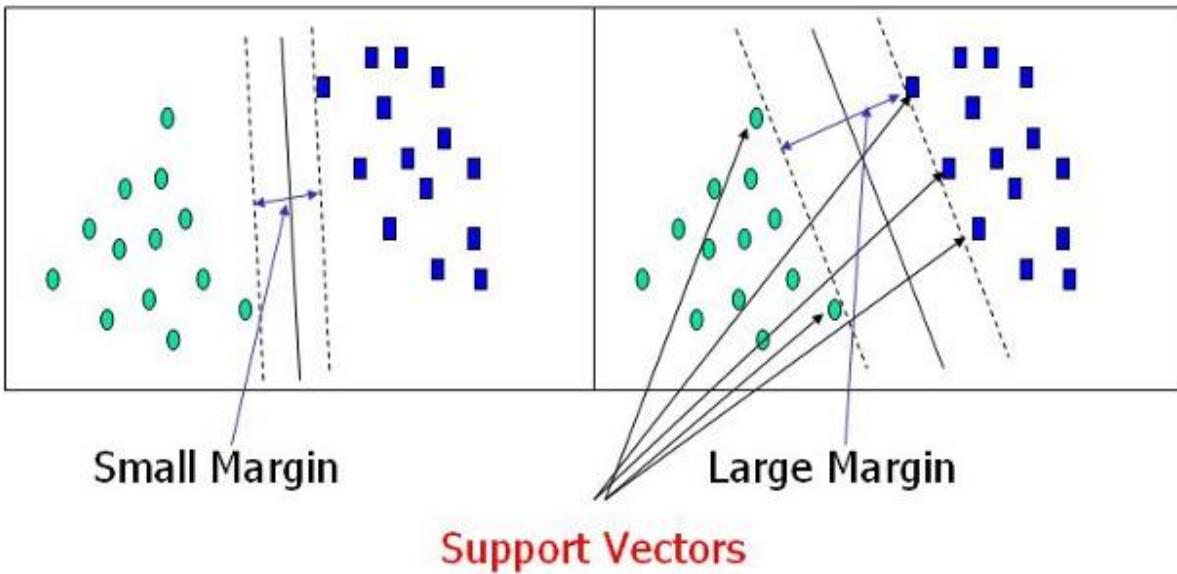


Figure 17: C maior (margens menores) vs. C Menor (margens maiores)

```
# Percentual de acertos e Kappa (leva em conta acertos aleatórios)
# Index de Rand e seu valor corrigido para acertos aleatórios.
>classAgreement(agree.tab)
$diag
[1] 0.8666667
$kappa
[1] 0.8
$rand
[1] 0.8528736
$crand
[1] 0.6590742
```

Os resultados valores foram bons. Classificamos corretamente ~86,7% das espécies com base em medidas das pétalas e sépalas em nossa amostra de teste. Fica uma dúvida. Na hora de ajustar o SVM, escolhemos o parâmetro cost. O parâmetro cost é um valor associado a Regularização dos pesos durante o treinamento. Ele reflete o quanto queremos evitar classificar exemplos de forma errada. Um valor pequeno vai priorizar margens maiores, mesmo que isso implique mais classificações erradas. Um C maior vai resultar num ajuste classificação correta para outliers, ainda que com margens menores.

Os melhores valores para o hiperparâmetro C dependem da estrutura do seus dados. Os autores do libsvm sugerem testar valores de C através de cross-validation.

Uma maneira de testar valores ótimos é através de uma função embutida no e1071(**tune.svm**), que já faz uso do dataset inteiro com 10-fold cross validation. Essa alternativa (**tune.svm**) costuma trazer melhores resultados segundo os autores do e1071.

```
#Ajustando valores testaveis de entre 1 e 1024
>tune.info <- tune.svm(Species~, data = iris, cost = 2^(0:10), kernel="linear")

#Sumario do tuning atraves de 10-fold-cross-validation
>summary(tune.info)
```

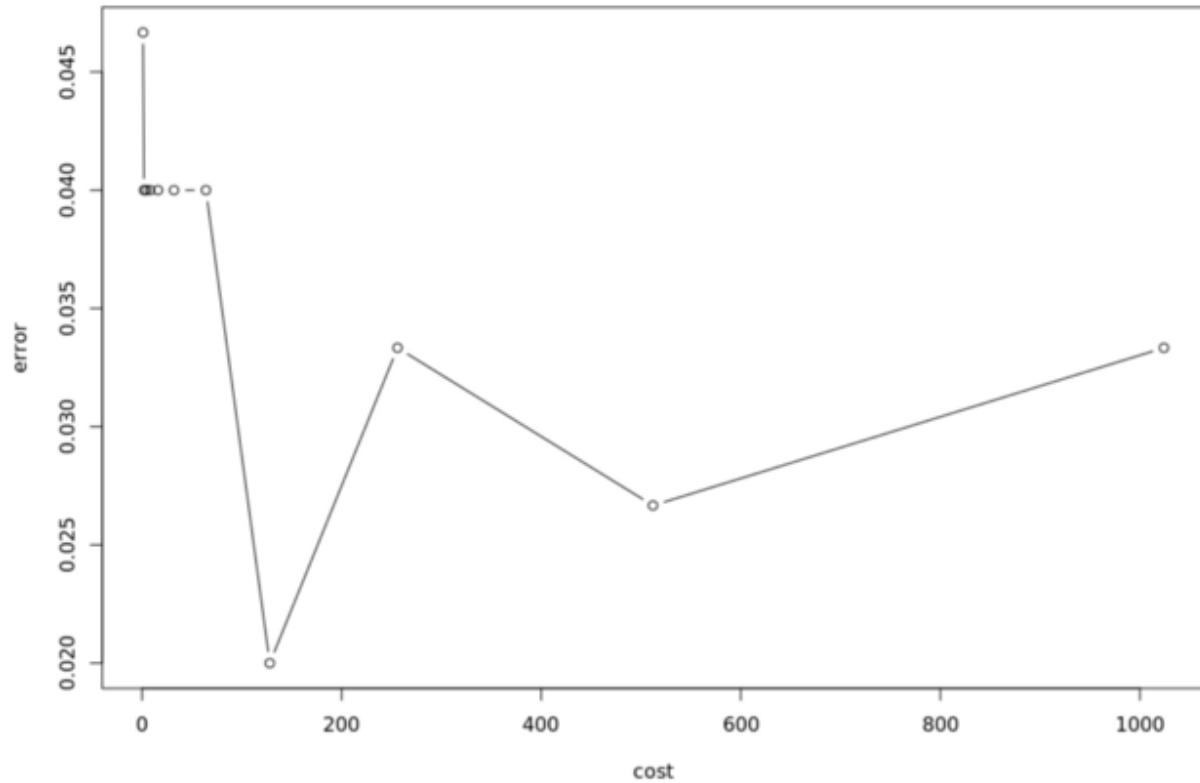
```
Parameter tuning of 'svm':
```

- sampling method: 10-fold cross validation
- best parameters:
cost
128
- best performance: 0.01333333
- Detailed performance results:

cost	error	dispersion
1	0.04000000	0.04661373
2	0.03333333	0.03513642
3	0.03333333	0.03513642
4	0.04000000	0.04661373
5	0.03333333	0.04714045
6	0.04000000	0.04661373
7	0.04000000	0.04661373
8	0.01333333	0.02810913
9	0.02000000	0.04499657
10	0.01333333	0.02810913
11	0.01333333	0.02810913

```
>plot(tune.info)
```

Performance of 'svm'



O tune.svm retornou, entre os valores, que os melhores parâmetros são cost = 128. Pelo gráfico, ainda é possível detectar zonas mais propícias e testar valores no intervalo. Podemos estabelecer esses parâmetros manualmente.

```
>svm.opt <- svm(Species ~ ., data=iris.tr, cost=128, kernel="linear")
```

Ou invocar o objeto com \$best.model (names(tune.info) para outros valores e objetos):

```
#Invocando melhor modelo.
>tune.info$best.model
Call:
best.svm(x = Species ~ ., data = iris, cost = 2^(0:10), kernel = "linear")

Parameters:
SVM-Type: C-classification
SVM-Kernel: linear
cost: 128
gamma: 0.25

Number of Support Vectors: 15
#Fazendo predicoes
>tune.pred <- predict(tune.info$best.model,iris)
#Tabela de classificacoes predicoes vs. observacoes
>agree.tune <- table(pred = tune.pred,true=iris$Species)
>agree.tune
      true
pred      setosa versicolor virginica
setosa     50       0       0
versicolor   0      48       1
virginica    0       2      49
#Observando concordancia das predicoes e observacoes
>classAgreement(agree.tune)
$diag
[1] 0.98
$kappa
[1] 0.97
$rand
[1] 0.9739597
$crand
[1] 0.9410123
```

Está implementado nosso classificador de espécies com base no comprimento e largura de sépalas e pétalas. Agora, um biólogo em dúvida sobre a espécie de uma nova amostra pode usar nosso programa para classificar a planta usando suas medidas. A mesma lógica serve para uso de SVM em outras áreas, como classificação de imagens e classificação de risco de créditos em instituições financeiras.

Considerações Algumas observações: Alguns devem ter notado que o modelo ajustado pelo tune tem um parâmetro gamma além do parâmetro C. Normalmente, o parâmetro gamma pertence a outros kernels (e1071, pag.50). Mudar o parâmetro gamma não altera o desempenho de uma SVM com kernel linear. Imagino que seja um artefato da função tune por lidar com diversos kernels. O paper a seguir sugere um caminho para implementação de SVMs para iniciantes (fazer Scaling dos dados sempre e dar preferência ao Kernel RBF). A Practical Guide to Support Vector Classification. Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. XXX XXX

Referências Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A Practical Guide to Support Vector Classification. Karatzoglou et al. Support Vector Machines in R. Journal of Statistical Software. April 2006,

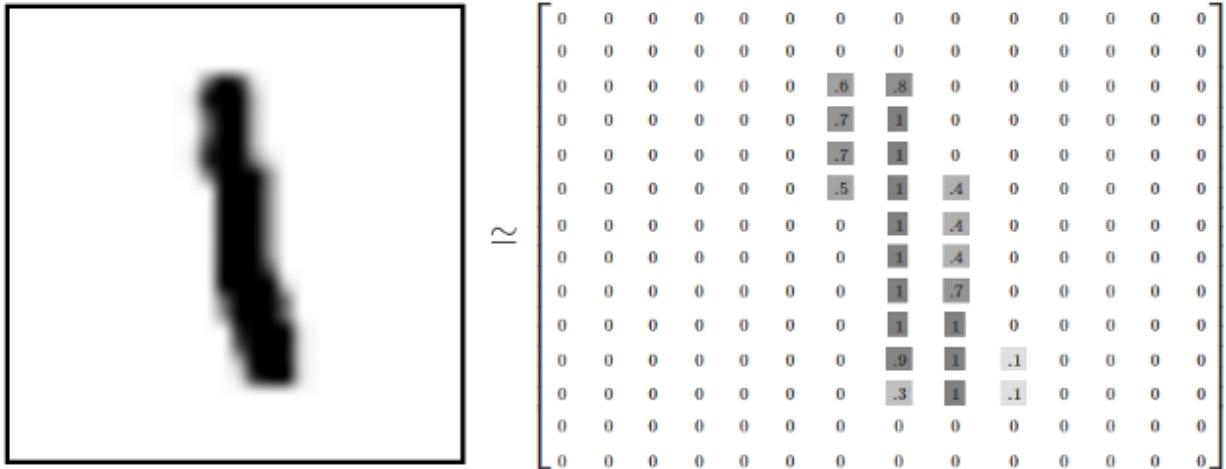


Figure 18: Exemplo de “1” em letra cursiva e sua representação numa matriz 2x2. <http://colah.github.io/posts/2014-10-Visualizing-MNIST/>

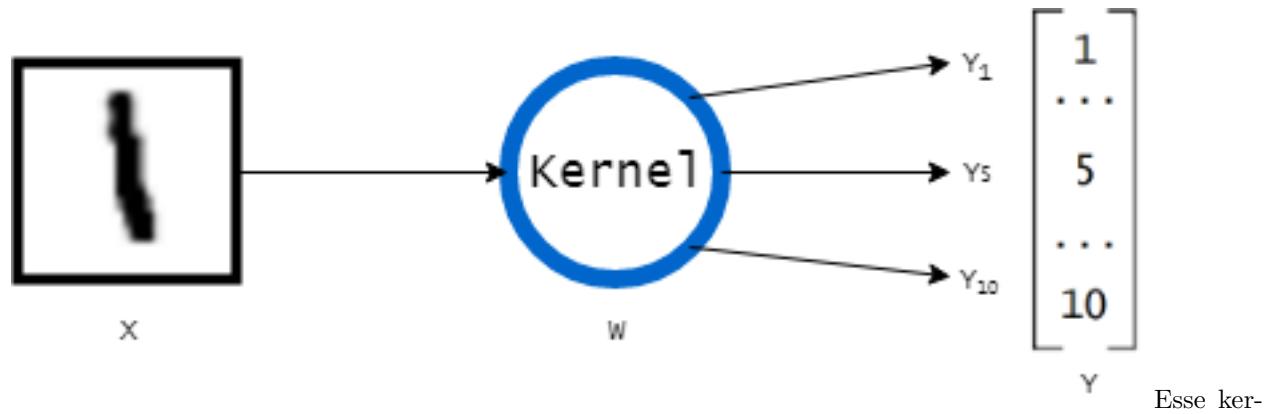
Volume 15, Issue 9. CS231n—Stanford University: Convolutional Neural Networks for Visual Recognition Documentação dos pacotes e1071 e caret do R.

Parte 3 - Deep learning

Nos textos anteriores (1^a parte — link), mostramos o funcionamento de um classificador simples e usamos (link) um pacote popular para ilustrar a configuração, treinamento e avaliação do modelo (Support Vector Machine). Um mal entendido envolvendo deep learning é de que produzimos uma caixa preta, útil porém inacessível. Vamos entender como funcionam redes profundas e de onde surge essa confusão.

Revisão

Podemos representar imagens usando matrizes, como na figura acima. O monitor lê cada número e ativa o ponto brilhante correspondente na tela, criando a ilusão de imagens e filmes. Implementamos um classificador simples (Support Vector Machine, SVM), que lê uma imagem, como o 1 acima, na forma de matriz. Usando uma função (kernel function), que aceita esse input e considera pesos internos (w), gerando scores empregados nas predições.

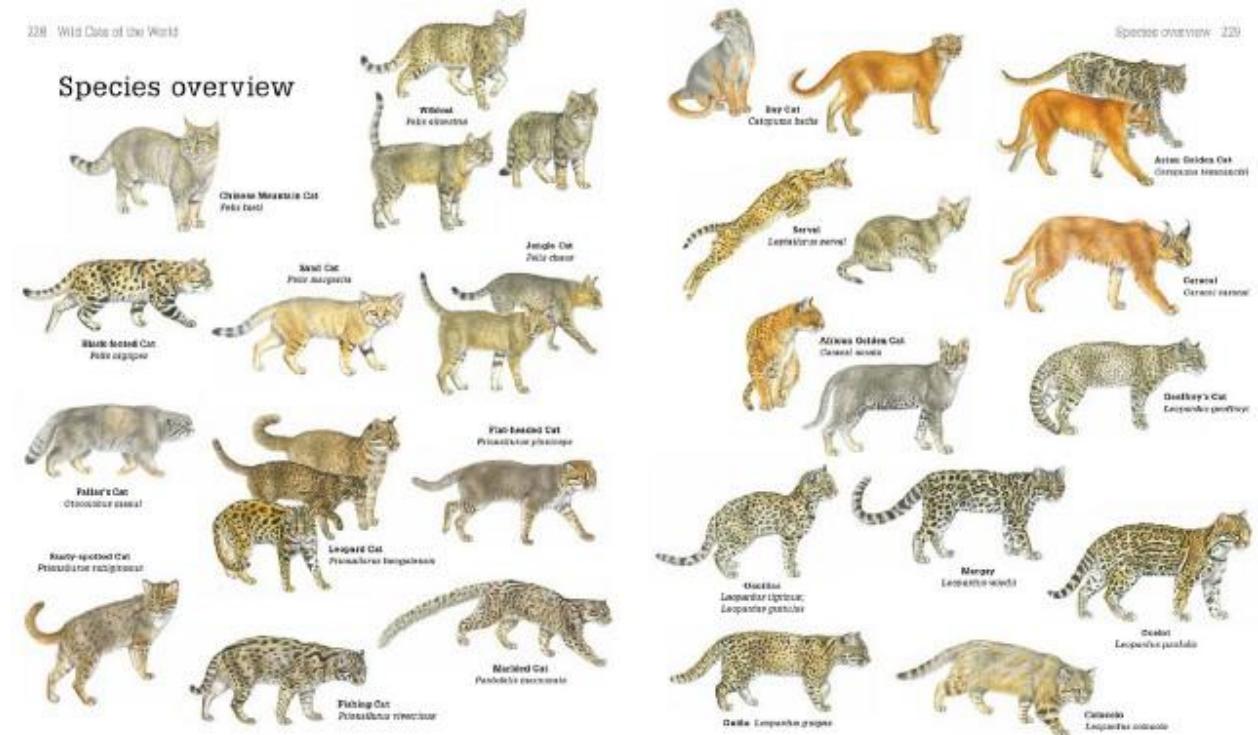


Intuições Com o aprendizado através de exemplos, otimizamos nosso classificador (mudando pesos W) para minimizar a perda, erro, usando aproximações(e.g: Adagrad). A função de perda é menor quando temos pontuações (votos) maiores para as classes certas. SVMs têm bom desempenho em diversas estruturas de dados, especialmente quando a arquitetura é otimizada por um usuário experiente. Onde entram as redes neurais?



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. Alguns filósofos acreditam que abstrações humanas são instâncias de um conceito mais genérico: mapas biológicos contidos em redes neurais (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.



Figure 19: Resposta a estímulos visuais em V1 de *Macaca fascicularis* <http://www.jneurosci.org/content/32/40/13971>

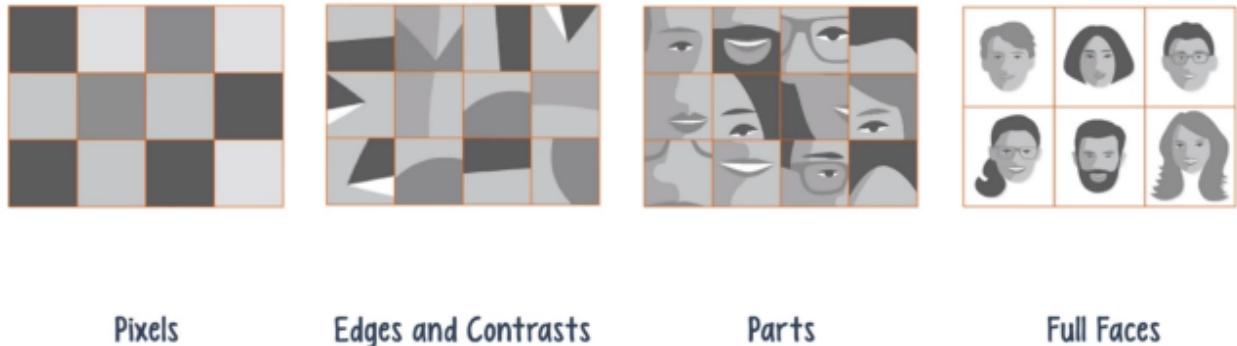


Figure 20: Retirado de: <https://www.youtube.com/watch?v=SeyIg6ArS4Y>

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).

Numa regressão logística simples, fazemos essa classificação com base nas probabilidades de uma função sigmoide. Temos uma probabilidade (valor entre 0 e 1). $P(h, \beta) = 1/(1 + \exp(-i + t * h + \beta * y + \epsilon))$. ϵ representa o erro e i é uma constante.

Em uma linha de R:

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

A vantagem de usar essa modelagem é que temos uma relação direta entre o inverso dessa função ($P^{\wedge}(-1)$, “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.

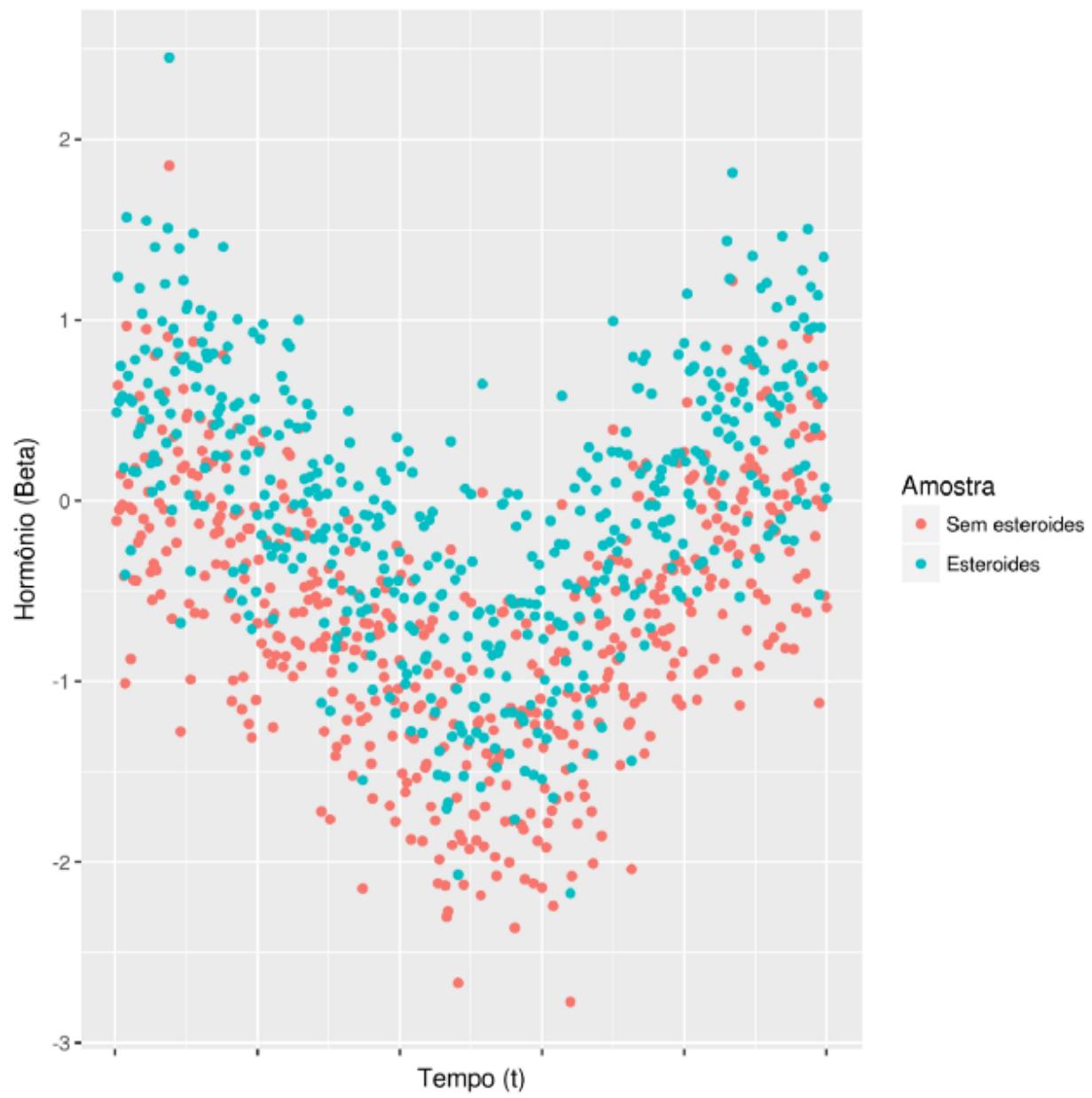
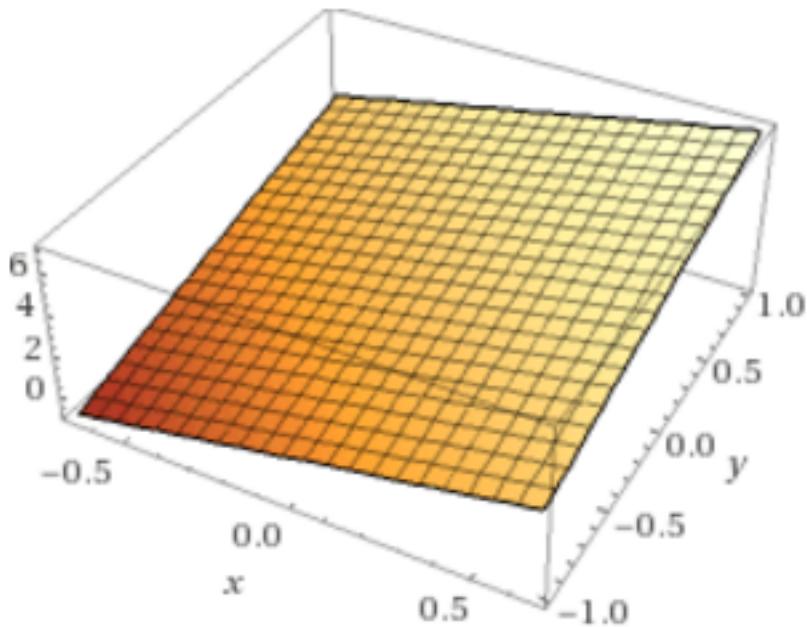
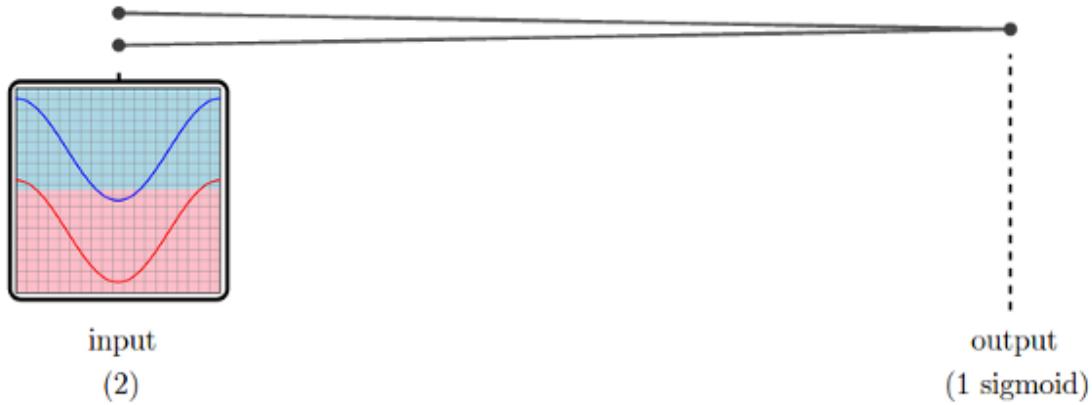


Figure 21: Exemplo inspirado no texto de Chris Olah (<http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/>)



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 nosso modelo é um plano. $z = 3 + 3x + 2y$. Plotado no Wolfram Alpha 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 apenas 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. <http://colah.github.io/posts/2015-01-Visualizing-Representations/> 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 dopping, 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
-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

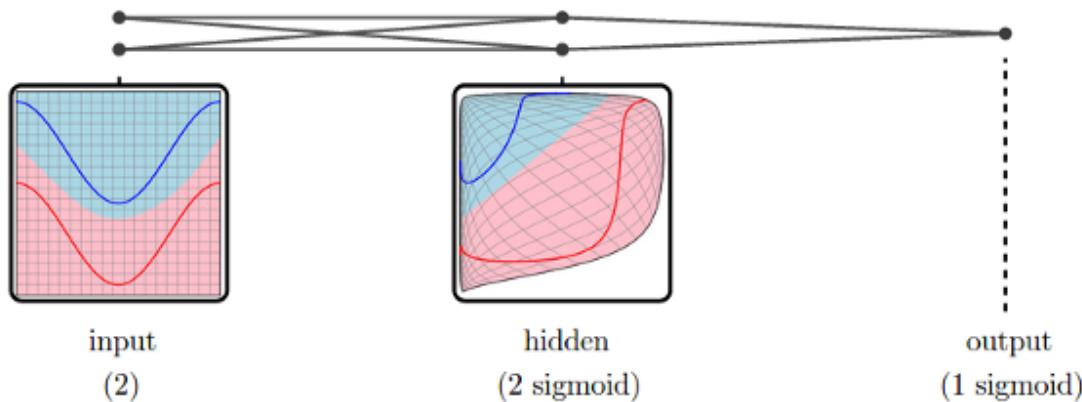
```

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.

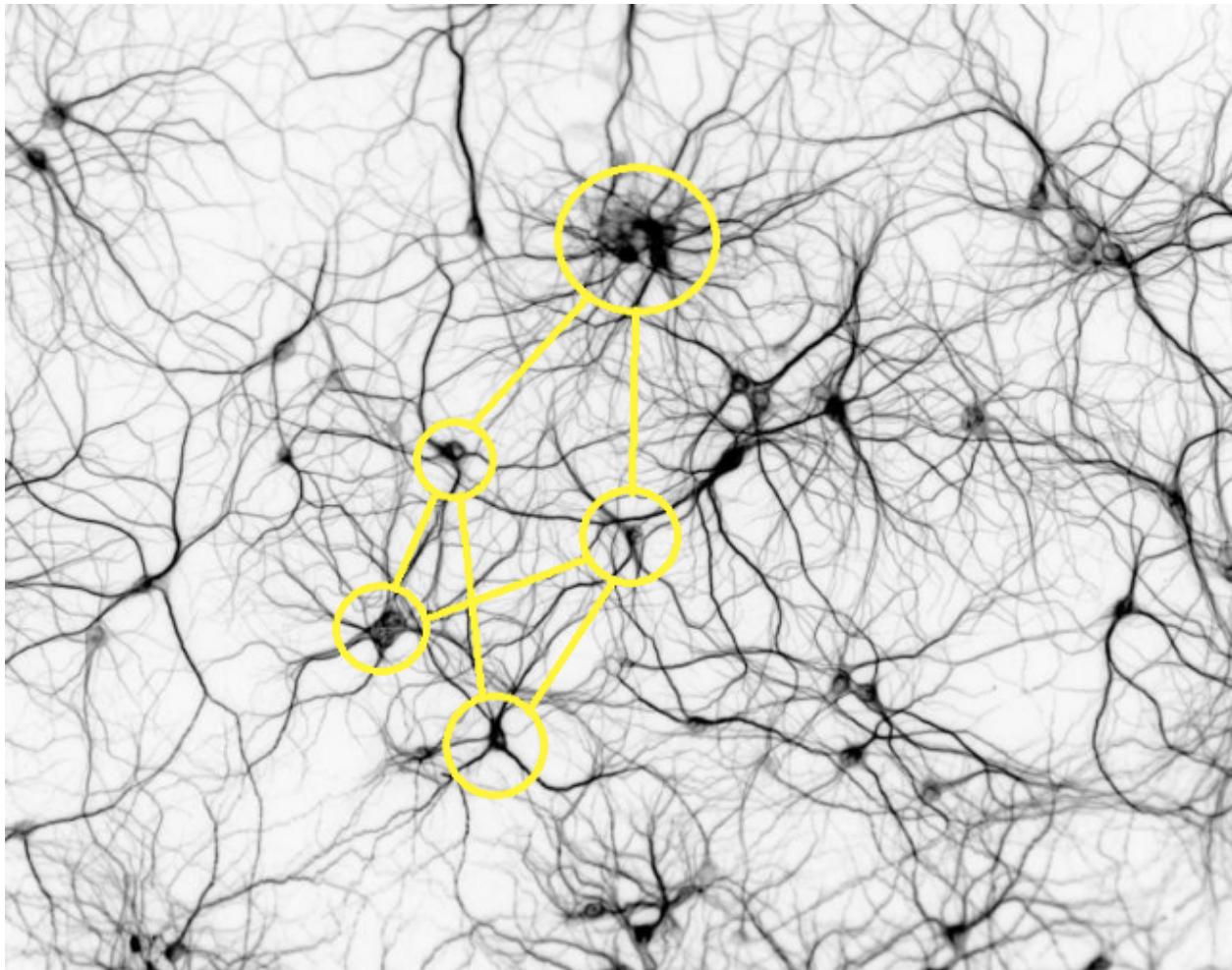
As camadas intermediárias permitem a transformação gradual do sinal, e o sistema acerta usando apenas dois classificadores simples (sigmoids). No exemplo acima, temos uma camada de 2 neurônios entre o input e o output.



put.

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).

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 mamíferos (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1557912/>). 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. Modificado de <http://www.rzagabe.com/2014/11/03/an-introduction-to-artificial-neural-networks.html>

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 redes neurais 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.

```
# Neural Net para o exemplo
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)

```

As redes neurais passaram algum tempo esquecidas, até que algumas reviravoltas (<http://people.idsia.ch/~juergen/who-invented-backpropagation.html>) permitiram o treinamento eficaz delas 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.

Backpropagation

Uma vez que o texto é sobre deep leaning, precisamos falar de backpropagation . Como vimos nas partes 1 e 2, o treinamento consiste em ajustar os pesos W do classificador (SVM) para minimizar a função que calcula nosso erro E. Como alguém de olhos vendados em uma ladeira, podemos dar um passo e saber medir qual o efeito sobre a nossa altura (subimos ou descemos, + ou -), assim como a intensidade (magnitude numérica: 50cm ou 70 cm). A partir daí, definimos uma regra para movimentação.



Quando treinamos um único nodo (SVM), o nosso trabalho é como o de um cego tateando até descer ao lugar mais baixo. É possível seguir o caminho aos poucos. Com redes profundas, a entrada de um nodo depende da saída dos que se conectam a ele. O sistema é um pouco mais complexo. Vamos usar derivadas. Ou seu equivalente para funções de múltiplas variáveis, gradiente. O gradiente é um vetor/lista com as derivadas parciais daquela função. Matematicamente, queremos a derivada parcial da função de custo (f) com respeito às entradas. Como vimos, podemos encarar a rede neural como uma sequência de funções plugadas. Se o primeiro nó tem $q(x,y)$, o segundo, f , tem valor $f(q(x,y))$ ou $f \circ q$.

$$q(x, y) = 3x + 2y \text{ #camada inferior}$$

$$f(z) = z^2 \text{ #camada superior}$$

$$f(q(x, y)) = q^2 = (3x + 2y)^2 \text{ # input inferior para superior}$$

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 ; df/dy] que é precisamente nosso gradiente.

```
# 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(dfdfx,dfdy)
> grad
[1] 24 16
```

Usando essa lógica, calculamos os gradientes para a função de erro e treinamos o modelo.

Referência Para uma história completa: 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).

Recomendo esse paper aqui para uma abordagem mais profunda e definições formais com hiperplanos — Support Vector Machines in R (Alexandros Karatzoglou, David Meyer, Kurt Hornik).