INSTITUTO NACIONAL DE ESTATÍSTICA
STATISTICS PORTUGAL

» R para a Ciência dos Dados

https://r4ds.had.co.nz/

Pedro Sousa
João Lopes

DMSI / ME

Outubro de 2019

Programa

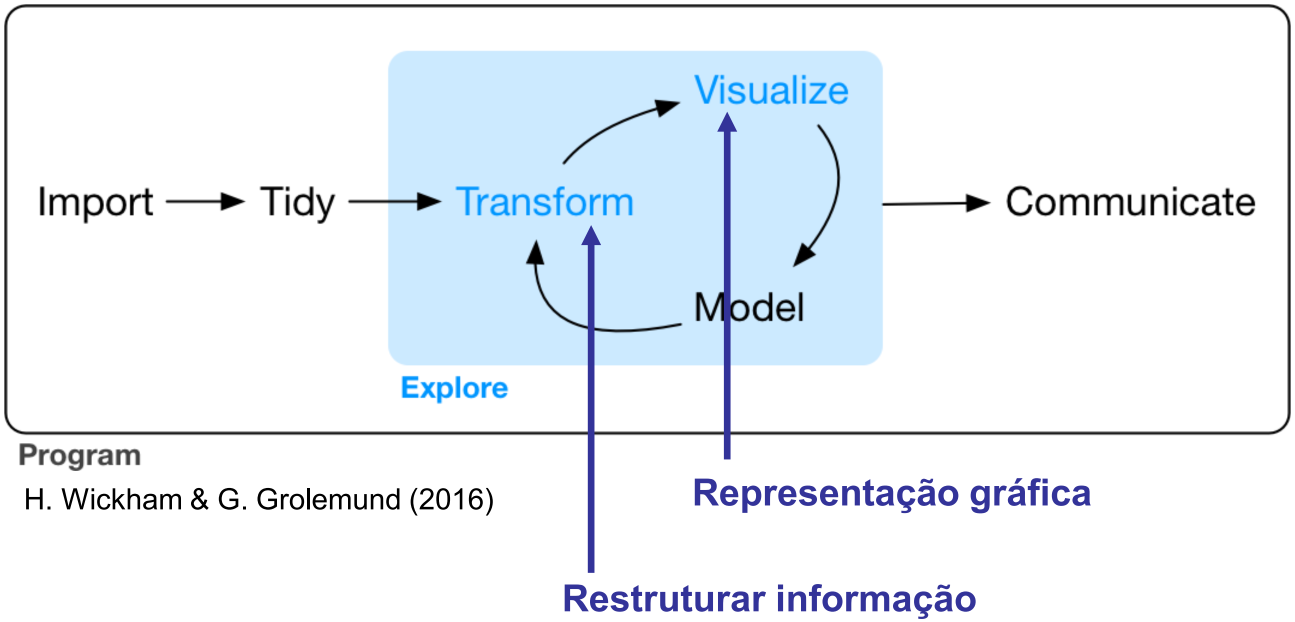
- Introdução
- Noções básicas do R
- Data *Wrangling*
- **Exploração dos dados**
- **Modelos e inferências**
- Comunicação

Programa

- **Exploração dos dados**
  - Visualização
  - Manipulação de dados
  - Exploração
- **Modelos e inferências**
  - Exemplo
  - Ajustamento
  - Diagnóstico

Exploração dos dados

» Esquema geral



H. Wickham & G. Grolemund (2016)

Exploração dos dados

» 1. Visualização (`ggplot2`):

- *Scatterplot* (e *smoothplot*);
- Gráfico de barras;
- *Boxplot*;
- Histograma (e curva de densidade).

» 1.1. Visualização: *scatterplot*

```
library("tidyverse")
?mpg
print(mpg)                                                          #table
ggplot(data=mpg) + geom_point(mapping=aes(x=displ,y=hwy))           #scatterplot
ggplot(data=mpg) + geom_point(mapping=aes(x=displ,y=hwy,color=class)) #w/ color
```

» 1.2. Visualização: *smoothplot*

```
library("tidyverse")
?mpg
print(mpg)                                                              #table
ggplot(data=mpg) + geom_point(mapping=aes(x=displ,y=hwy))               #scatterplot
ggplot(data=mpg) + geom_point(mapping=aes(x=displ,y=hwy,color=class))   #w/ color
ggplot(data=mpg) + geom_smooth(mapping=aes(x=displ,y=hwy))              #smoothplot
ggplot(data=mpg) + geom_smooth(mapping=aes(x=displ,y=hwy,color=class))  #w/ color
```

» 1.3. Visualização: gráfico de barras

```
library("tidyverse")
?diamonds
print(diamonds)                                                    #table
ggplot(data=diamonds) + geom_bar(mapping=aes(x=cut))               #barplot
ggplot(data=diamonds) + geom_bar(mapping=aes(x=cut,y=..prop..,group=1)) #barplot %
ggplot(data=diamonds) + geom_bar(mapping=aes(x=cut,fill=clarity))  #stacked
ggplot(data=diamonds) + geom_bar(mapping=aes(x=cut,fill=clarity),
                                 position="fill")                  #stacked %
ggplot(data=diamonds) + geom_bar(mapping=aes(x=cut,fill=clarity),
                                 position="dodge")                 #clustered
```

» 1.4. Visualização: *boxplot*

```
library("tidyverse")
ggplot(data=diamonds) + geom_boxplot(mapping=aes(y=carat))         #boxplot
ggplot(data=diamonds) + geom_boxplot(mapping=aes(x=cut,y=carat)) #multiple boxplot
```

» 1.5. Visualização: histograma

```
library("tidyverse")
ggplot(data=diamonds) + geom_histogram(mapping=aes(x=carat)) #histogram
ggplot(data=diamonds) + geom_histogram(mapping=aes(x=carat),
                                       binwidth=0.01)        #histogram
```

» 1.6. Visualização: curvas de densidade

```
library("tidyverse")
ggplot(data=diamonds) + geom_histogram(mapping=aes(x=carat)) #histogram
ggplot(data=diamonds) + geom_histogram(mapping=aes(x=carat),
                                       binwidth=0.01)        #histogram
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat),
                                      binwidth=0.01)         #density line
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat,color=cut),
                                      binwidth=0.1)          #multiple density line
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat,y=..density..,color=cut),
                                      binwidth=0.1)          #multiple density line %
```

» 1.7. Visualização: curvas de densidade 2

```
library("tidyverse")
ggplot(data=diamonds) + geom_histogram(mapping=aes(x=carat)) #histogram
ggplot(data=diamonds) + geom_histogram(mapping=aes(x=carat),
                                        binwidth=0.01)        #histogram
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat),
                                       binwidth=0.01)         #density line
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat,color=cut),
                                       binwidth=0.1)          #multiple density line
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat,y=..density..,color=cut),
                                       binwidth=0.1)          #multiple density line %
ggplot(data=diamonds) + geom_freqpoly(mapping=aes(x=carat,y=..density..),
                                       binwidth=0.1) +
  geom_density(mapping=aes(x=carat),color="red")              #density line 2
```

Exploração dos dados

» 2. Manipulação dos dados (`dplyr`):

- `filter()` Selecionar linhas (i.e. observações);
- `arrange()` Rearranjar linhas (i.e. observações);
- `select()` Selecionar colunas (i.e. variáveis);
- `mutate()` Criar novas colunas (i.e. variáveis);
- `summarize()` Calcular estatísticas descirtivas.

- `group_by()` Criar grupos de observações para manipulação.

» 2.1. Manipulação dos dados: `filter()`

```
library("tidyverse")
library("nycflights13")
?flights
print(flights)                                              #table
jan1 <- flights %>%
  filter(month==1,day==1)
print(jan1)                                                 #filter 1
nov_dec <- flights %>%
  filter(month %in% c(11,12))
print(nov_dec)                                              #filter 2
no_late <- flights %>%
  filter(arr_delay <= 120 & dep_delay <= 120)
print(no_late)                                              #filter 3
```

» 2.2. Manipulação dos dados: arrange()

```
library("tidyverse")
library("nycflights13")
ord_date <- flights %>%
  arrange(year,month,day)
print(ord_date)                                          #arrange 1
ord_arr_delay <- flights %>%
  arrange(desc(arr_delay))
print(ord_arr_delay)                                     #arrange 2
```

» 2.3. Manipulação dos dados: `select()`

```
library("tidyverse")
library("nycflights13")
flights %>%
  select(year,month,day)                  #select 1
flights %>%
  select(year:day)                        #select 2
flights %>%
  select(-(year:day))                     #select 3
flights %>%
  select(contains("arr"))                 #select 4
flights %>%
  select(time_hour,air_time,everything()) #select 5
flights %>%
  rename(tail_num=tailnum)                #rename 1
```

» 2.4. Manipulação dos dados: `mutate()`

```
library("nycflights13")
flights_sml <- flights %>%
  select(year:day,ends_with("delay"),distance,air_time)
print(flights_sml)                                      #table
flights_sml %>%
  mutate(gain = arr_delay - dep_delay,
         speed = distance/air_time*60)                  #mutate 1
flights_sml %>%
  mutate(gain = arr_delay - dep_delay,
         hours = air_time/60,
         gain_per_hour = gain/hours)                    #mutate 2
flights_sml %>%
  transmute(gain = arr_delay - dep_delay,
            hours = air_time/60,
            gain_per_hour = gain/hours)                 #transmute 1
```

» 2.4. Manipulação dos dados: `mutate()`

- Operadores aritméticos `+`, `-`, `*`, `/`, `^`;

- Moduladores aritméticos `%/%`, `%%`;

- Logs `log()`, `log2()`, `log10()`;

- Deslocador `lead()`, `lag()`;

- Acumuladores `cumsum()`, `cumprod()`, `cummin()`, `cummean()`;

- Comparadores lógicos `<`, `<=`, `>`, `>=`, `!=`;

- *Ranking* `min_rank()`, `percent_rank()`.

» 2.5. Manipulação dos dados: summarize()

```
library("nycflights13")
flights %>%
  summarize(delay = mean(dep_delay,na.rm=TRUE))              #summarize 1
flights %>%
  group_by(dest) %>%
  summarize(count = n(),
            dist = mean(distance,na.rm=TRUE),
            delay = mean(dep_delay,na.rm=TRUE)) %>%
  filter(count > 20,dest != "HNL")                           #summarize 2
delay <- group_by(.data=flights,dest)
delay <- summarize(.data=delay,count = n(),
                                dist = mean(distance,na.rm=TRUE),
                                delay = mean(dep_delay,na.rm=TRUE))
delay <- filter(.data=delay,count > 20,dest != "HNL")       #summarize 3
```

» 2.5. Manipulação dos dados: `summarize()`

- Medidas de tendência central `mean()`, `median()`;
- Medidas de dispersão `sd()`, `IQR()`, `mad()`;
- Medidas de *ranking* `min()`, `quantile()`, `max()`;
- Medidas de posição `first()`, `nth()`, `last()`;
- Medidas de contagem `n()` , `sum(!is.na())`, `n_distinct()`;

» 2.6. Manipulação dos dados: `group_by()`

```
library("nycflights13")
flights %>%
  group_by(year) %>%
  summarize(nflights = n())                    #group 1
daily <- flights %>%
  group_by(year,month,day)                     #group 2
per_day <- daily %>%
  summarize(nflights = n())
per_month <- per_day %>%
  summarize(nflights = sum(nflights))
per_year <- per_month %>%
  summarize(nflights = sum(nflights))
daily %>%
  ungroup() %>%
  summarize(nflights = n())                    #ungroup 1
```

Exploração dos dados

» 3. Exploração:

- *Outliers* e *Missing values*;
- Distribuição de variáveis (Tendência, Variação, Normalidade);
- Distribuição conjunta de variáveis (Visualização, Correlação).

» 3.1. Exploração: *outliers*

```
library("tidyverse")
?diamonds
print(diamonds)
diamonds %>%
  summary()
diamonds %>% ggplot() + geom_histogram(mapping=aes(x=x),binwidth=0.5)
diamonds %>% ggplot() + geom_histogram(mapping=aes(x=y),binwidth=0.5)
diamonds %>% ggplot() + geom_histogram(mapping=aes(x=z),binwidth=0.5)
diamonds %>% ggplot() + geom_histogram(mapping=aes(x=x),binwidth=0.5) +
                        coord_cartesian(ylim=c(0,50))
diamonds %>% ggplot() + geom_histogram(mapping=aes(x=y),binwidth=0.5) +
                        coord_cartesian(ylim=c(0,50))
diamonds %>% ggplot() + geom_histogram(mapping=aes(x=z),binwidth=0.5) +
                        coord_cartesian(ylim=c(0,50))
```

» 3.1. Exploração: *outliers*

```
unusual <- diamonds %>%
  filter((x < 1) | (y < 1 | y > 20) | (z < 1 | z > 10)) %>%
  arrange(x)
print(unusual,n=23)
diamonds %>% ggplot() + geom_point(mapping=aes(x=price,y=x)) +
                        geom_point(data=unusual,mapping=aes(x=price,y=x),color="red")
diamonds %>% ggplot() + geom_point(mapping=aes(x=price,y=y)) +
                        geom_point(data=unusual,mapping=aes(x=price,y=y),color="red")
diamonds %>% ggplot() + geom_point(mapping=aes(x=price,y=z)) +
                        geom_point(data=unusual,mapping=aes(x=price,y=z),color="red")
diamonds %>%
  filter((x > 0) & (y > 0 & y < 20) & (z > 0 & z < 10)) %>%
  summary()
```

» 3.2. Exploração: *missing values*

```
library("tidyverse")
diamonds_NA <- diamonds %>%
  mutate(x = ifelse(x < 1,NA,x)) %>%
  mutate(y = ifelse(y < 1 | y > 20,NA,y)) %>%
  mutate(z = ifelse(z < 1 | z > 10,NA,z))
print(diamonds_NA)
```

» 3.2. Exploração: *missing values*

```
diamonds_NA %>%
  group_by(cut) %>%
  summarize(mean_x = mean(x), mean_y = mean(y), mean_z = mean(z), n = n(),
            x_NA = sum(is.na(x)), y_NA = sum(is.na(y)), z_NA = sum(is.na(z)))
diamonds_NA %>%
  group_by(cut) %>%
  mutate(xyz = x + y + z) %>%
  summarize(mean_x = mean(x,na.rm=TRUE), mean_y = mean(y,na.rm=TRUE),
            mean_z = mean(z,na.rm=TRUE), n=n(), x_NA = sum(is.na(x)),
            y_NA = sum(is.na(y)), z_NA = sum(is.na(z)), nNA = sum(is.na(xyz)))
diamonds_NA %>%
  ggplot() + geom_point(mapping=aes(x=price,y=x))            #warning
diamonds_NA %>%
  ggplot() + geom_point(mapping=aes(x=price,y=x),na.rm=TRUE) #no warn
```

» 3.3. Exploração: distribuição 1D

```
library("tidyverse")
library("moments")
diamonds2 <- diamonds %>%
 filter((x > 0) & (y > 0 & y < 20) & (z > 0 & z < 10))
print(diamonds2)
diamonds2 %>% ggplot() + geom_histogram(mapping=aes(x=price),binwidth=50)
ldiamonds2 <- diamonds2 %>%
  mutate(price = log(price))
ldiamonds2 %>% ggplot() + geom_histogram(mapping=aes(x=price),binwidth=0.01)
param <- diamonds2 %>% summarize(n = n(), mean = mean(price), sd = sd(price),
                                 median = median(price), IQR = IQR(price))
lparam <- ldiamonds2 %>% summarize(n = n(), mean = mean(price), sd = sd(price),
                                   median = median(price), IQR = IQR(price))
print(round(cbind(t(param),t(lparam)),digits=2))
```

» 3.3. Exploração: distribuição 1D

```
ldiamonds2_scale <- ldiamonds2 %>%
  mutate(price = (price - mean(price))/sd(price))
ldiamonds2_scale %>% ggplot() +
  geom_histogram(mapping=aes(x=price,y=..density..),binwidth=0.2) +
  stat_function(fun=dnorm,color="red",args=list(mean=0,sd=1))
price_mean <- mean(ldiamonds2$price)
price_sd <- sd(ldiamonds2$price)
ldiamonds2 %>% ggplot() +
  geom_histogram(mapping=aes(x=price,y=..density..),binwidth=0.2) +
  stat_function(fun=dnorm,color="red",args=list(mean=price_mean,sd=price_sd))
param <- ldiamonds2_scale %>% summarize(n = n(), mean = mean(price), sd = sd(price),
  skewness = skewness(price), kurtosis = kurtosis(price))
tab1 <- round(cbind(t(param),c(1/0,0,1,0,3)),digits=3)
colnames(tab1) <- c("original","Gaussian")
print(tab1)
```

Exploração dos dados

» 3.4. Exploração: distribuição 2D (cat vs. cont)

```
library("tidyverse")
?diamonds
diamonds2 <- diamonds %>%
 filter((x > 0) & (y > 0 & y < 20) & (z > 0 & z < 10))
print(diamonds2)
diamonds2 %>%
  filter(cut=="Ideal") %>%
  ggplot() + geom_boxplot(mapping=aes(x=clarity,y=price))
diamonds2 %>%
  filter(cut=="Ideal") %>%
  group_by(clarity) %>%
  summarize(mean_price = mean(price))
```

» 3.4. Exploração: distribuição 2D (cat vs. cont)

```
diamonds2 %>%
  filter(cut=="Ideal") %>%
  group_by(clarity) %>%
  mutate(mean_price = mean(price)) %>%
  ggplot() +
    geom_point(mapping=aes(x=mean_price,y=price),size=2,alpha=0.01) +
    geom_point(mapping=aes(x=mean_price,y=mean_price),size=2,color="red")
diamonds2 %>%
  filter(cut=="Ideal") %>%
  group_by(clarity) %>%
  mutate(mean_price = mean(price)) %>%
  ungroup() %>%
  summarize(res = cor(mean_price,price))   #correlation observed vs. predicted values
```

Exploração dos dados

» 3.4. Exploração: distribuição 2D (2 cat)

```
library("tidyverse")
diamonds2 <- diamonds %>%
 filter((x > 0) & (y > 0 & y < 20) & (z > 0 & z < 10))
print(diamonds2)
diamonds2 %>% ggplot() +
  geom_count(mapping=aes(x=clarity,y=cut))
diamonds2 %>%
  group_by(clarity,cut) %>%
  summarize(n = n()) %>%
  ggplot() + geom_count(mapping=aes(x=clarity,y=cut,color=n,size=n))
diamonds2 %>%
  group_by(clarity,cut) %>%
  summarize(n = n()) %>%
  ggplot() + geom_tile(mapping=aes(x=clarity,y=cut,fill=n))
```

» 3.4. Exploração: distribuição 2D (2 cat)

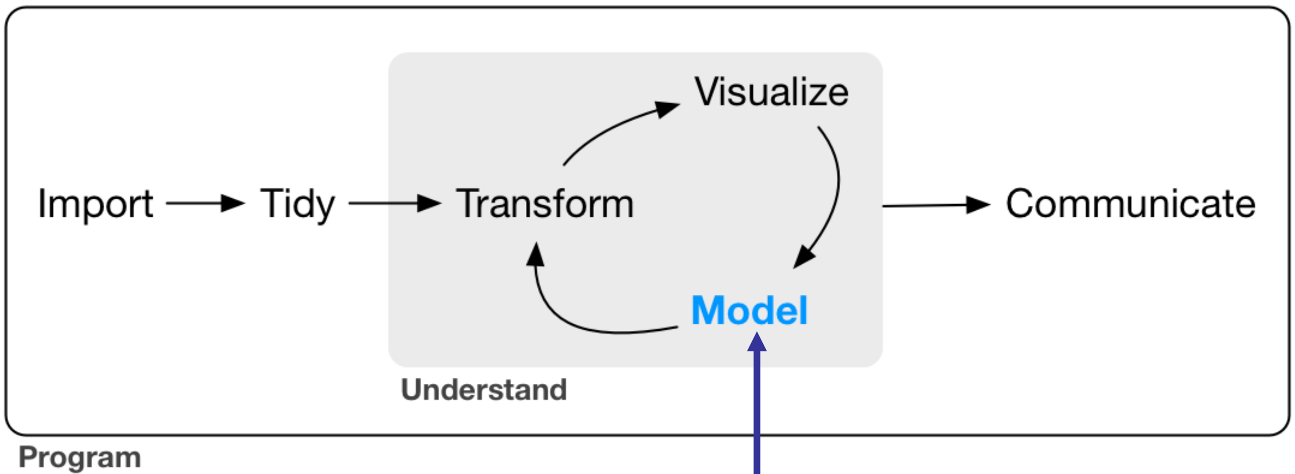
```
#Calculate Cramer's V from contingency table
calc_Cramers_V <- function(x){
    chi_stat <- chisq.test(x)$statistic #chi-sqrt
    n <- sum(x)                         #sample size
    min_dim <- min(dim(x)) - 1          #minimum number of dimensions - 1
    res <- sqrt(chi_stat/n/min_dim)     #Cramer's V
    names(res) <- "Cramers.V"
    return(res)
}
cont_table <- table(diamonds2$clarity,diamonds2$cut) #contingency table
print(cont_table)
round(calc_Cramers_V(cont_table),digits=3)
```

» 3.4. Exploração: distribuição 2D (2 cont)

```
library("tidyverse")
library("hexbin")
?diamonds
diamonds2 <- diamonds %>%
 filter((x > 0) & (y > 0 & y < 20) & (z > 0 & z < 10))
print(diamonds2)
diamonds2 %>% ggplot() + geom_point(mapping=aes(x=carat,y=price))
diamonds2 %>% ggplot() + geom_point(mapping=aes(x=carat,y=price),alpha=0.01)
diamonds2 %>% ggplot() + geom_bin2d(mapping=aes(x=carat,y=price),bins=100)
diamonds2 %>% ggplot() + geom_hex(mapping=aes(x=carat,y=price),bins=100)
diamonds2 %>% ggplot() +
  geom_boxplot(mapping=aes(x=carat,y=price,group=cut_width(carat,0.1)))
diamonds2 %>% ggplot() +
  geom_boxplot(mapping=aes(x=carat,y=price,group=cut_number(carat,20)))
diamonds2 %>% summarize(res = cor(carat,price))
```

Modelos e inferências

» Esquema geral



H. Wickham & G. Grolemund (2016)

Criação de modelos e inferências

» 1. Modelação: exemplo

“All models are wrong, but some are useful”

“Now it would be very remarkable if any system existing in the real world could be exactly represented by any simple model. However, cunningly chosen parsimonious models often do provide remarkably useful approximations.”

George Box

Modelos e inferências

» 1. Modelação: exemplo

```
library("tidyverse")
library("hexbin")
library("modelr")
?diamonds
print(diamonds)
diamonds %>% ggplot() + geom_boxplot(aes(cut,price))          #price ~ cut
diamonds %>% ggplot() + geom_boxplot(aes(clarity,price))      #price ~ clarity
diamonds %>% ggplot() + geom_hex(aes(carat,price),bins=50)    #price ~ carat
diamonds2 <- diamonds %>%
  filter((x > 0) & (y > 0 & y < 20) & (z > 0 & z < 10)) %>%
  filter(carat <= 2.5) %>%
  select(carat,cut,color,clarity,price) %>%
  mutate(lprice = log(price), lcarat = log(carat))
print(diamonds2)
diamonds2 %>% ggplot() + geom_hex(aes(lcarat,lprice),bins=50) #lprice ~ lcarat
```

» 1. Modelação: exemplo

```
diamonds_mod <- lm(lprice ~ lcarat,data=diamonds2)              #fit model
summary(diamonds_mod)
diamonds2 <- diamonds2 %>%
  add_predictions(diamonds_mod,"lpred") %>%
  add_residuals(diamonds_mod,"lresid") %>%
  mutate(pred = exp(lpred))
print(diamonds2)
diamonds2 %>% ggplot() +
  geom_hex(aes(carat,price),bins=50) +
  geom_line(aes(carat,pred),color="red")                        #price ~ carat
diamonds2 %>% ggplot() + geom_boxplot(aes(cut,lresid))          #lresid ~ cut
diamonds2 %>% ggplot() + geom_boxplot(aes(clarity,lresid))      #lresid ~ clarity
diamonds2 %>% ggplot() + geom_hex(aes(lcarat,lresid),bins=50)   #lresid ~ lcarat
diamonds2 %>% ggplot() + geom_point(aes(pred,price)) +
  geom_abline(aes(intercept=0,slope=1),size=1,color="white")    #price ~ pred
```

» 1. Modelação: exemplo

```
diamonds_mod2  <- lm(lprice ~ lcarat + clarity + cut,data=diamonds2) #fit model
summary(diamonds_mod2)
diamonds2 <- diamonds2 %>%
  add_predictions(diamonds_mod2,"lpred2") %>%
  add_residuals(diamonds_mod2,"lresid2") %>%
  mutate(pred2 = exp(lpred2))
print(diamonds2)
diamonds2 %>%
  ggplot() +
    geom_hex(aes(carat,price),bins=50) +
    geom_line(aes(carat,pred2),color="red") +
    facet_grid(cut ~ clarity)                                        #price ~ carat
diamonds2 %>% ggplot() + geom_hex(aes(lcarat,lresid2),bins=50)       #lresid ~ lcarat
diamonds2 %>% ggplot() + geom_point(aes(pred2,price)) +
  geom_abline(aes(intercept=0,slope=1),size=1,color="white")        #price ~ pred2
```

» 2. Ajustamento (**modelr**)

```
library("tidyverse")
library("modelr")
set.seed(12345)
real_a1 <- 4.22
real_a2 <- 2.05
x <- round(runif(n=30,min=0,max=10),digits=2)
y <- round(real_a1*x + real_a2 + rnorm(n=30,mean=0,sd=1),digits=2)
sim1 <- tibble(x,y)
print(sim1)
sim1 %>% ggplot() + geom_point(aes(x,y))
```

Modelos e inferências

» 2.1. Ajustamento: *random search*

```
models <- tibble(
  a1 = runif(250,-20,40),
  a2 = runif(250,-5,5)
)
sim1 %>% ggplot() +
  geom_abline(data=models,aes(intercept=a1,slope=a2),alpha=0.25) +
  geom_point(aes(x,y))
```

» 2.1. Ajustamento: *random search*

```
#Linear regression model
model1 <- function(a,dat){
  a[1] + dat$x*a[2]
}
#Calculate Root-mean-squared-deviation
measure_distance <- function(params,dat){
  diff <- dat$y - model1(params,dat)
  sqrt(mean(diff^2))
}
#Helper function to calculate RMSD for synthetic data "sim1"
sim1_dist <- function(a1,a2){
  measure_distance(c(a1,a2),dat=sim1)
}
models <- models %>% mutate(RMSD = map2_dbl(a1,a2,sim1_dist))
print(models)
```

» 2.1. Ajustamento: *random search*

```
best_models <- models %>%
  filter(rank(RMSD) <= 10)
sim1 %>% ggplot() +
  geom_point(aes(x,y),size=2,color="grey30") +
  geom_abline(data=best_models,aes(intercept=a1,slope=a2,color=-RMSD))
models %>% ggplot() +
  geom_point(data=best_models,aes(a1,a2),size=4,color="red") +
  geom_point(aes(a1,a2,color=-RMSD))
```

» 2.2. Ajustamento: *grid search*

```
models_grid <- expand.grid(
  a1 = seq(0,15,length=25),
  a2 = seq(2,5,length=25)) %>%
    mutate(RMSD=map2_dbl(a1,a2,sim1_dist))
head(models_grid,n=15)
best_grid <- models_grid %>%
  filter(rank(RMSD) <= 10)
sim1 %>% ggplot() +
  geom_point(aes(x,y),size=2,color="grey30") +
  geom_abline(data=best_grid,aes(intercept=a1,slope=a2,color=-RMSD))
models_grid %>% ggplot() +
  geom_point(data=best_grid,aes(a1,a2),size=4,color="red") +
  geom_point(aes(a1,a2,color=-RMSD))
```

» 2.3. Ajustamento: *optimization*

```
best_optim <- optim(c(0,0),measure_distance,dat=sim1)
best_a1 <- best_optim$par[1]
best_a2 <- best_optim$par[2]
sim1 %>% ggplot() +
  geom_point(aes(x,y),size=2,color="grey30") +
  geom_abline(aes(intercept=best_a1,slope=best_a2))
```

» 2.4. Ajustamento: *least-squares*

```
best_lm <- lm(y ~ x,data=sim1)
best_a1 <- best_lm$coef[1]
best_a2 <- best_lm$coef[2]
sim1 %>% ggplot() +
  geom_point(aes(x,y),size=2,color="grey30") +
  geom_abline(aes(intercept=best_a1,slope=best_a2))
```

» 2.5. Ajustamento: comparação

```
res1 <- models %>%
  filter(rank(RMSD) == 1)
res2 <- models_grid %>%
  filter(rank(RMSD) == 1)
res3 <- tibble(
  a1 = best_optim$par[1],
  a2 = best_optim$par[2],
  RMSD = best_optim$value)
res4 <- tibble(
  a1 = best_lm$coef[1],
  a2 = best_lm$coef[2],
  RMSD = sqrt(mean(best_lm$residuals^2)))
tab1 = round(data.frame(rbind(res1,res2,res3,res4)),digits=2)
rownames(tab1) = c("random","grid","NR","LS")
print(tab1)
```

Modelos e inferências

» 3. Diagnóstico (`modelr`)

1. Linearidade entre variável de resposta e variáveis explicativas;
2. Não há multicolinearidade entre variáveis explicativas;
3. Resíduos têm média igual a zero;
4. Resíduos têm variância constante (homocedasticidade);
5. Resíduos não estão autocorrelacionados;
6. Resíduos são independentes das variáveis explicativas;
7. Resíduos têm distribuição Normal.

Greene, 2012, Econometric Analysis

» 3. Diagnóstico (modelr)

```
library("tidyverse")
library("modelr")
set.seed(12345)
real_a1 <- 4.22
real_a2 <- 2.05
x <- round(runif(n=30,min=0,max=10),digits=2)
y <- real_a1*x + real_a2 + rnorm(n=30,mean=0,sd=1)
sim1 <- tibble(x,y)
sim1_mod <- lm(y ~ x,data=sim1)
sim1 <- sim1 %>%
  add_predictions(sim1_mod) %>%
  add_residuals(sim1_mod)
print(sim1)
```

» 3. Diagnóstico (modelr)

```
sim1 %>% ggplot() + geom_point(aes(x,y),size=2,color="grey30") +
  geom_line(aes(x,pred)) +
  geom_segment(aes(x=x,xend=x,y=y,yend=pred),alpha=0.2)          #scatterplot y ~ x
sim1 %>% ggplot() + geom_abline(aes(intercept=0,slope=1),size=2,color="white") +
  geom_point(aes(pred,y),size=2,color="grey30")                  #scatterplot y ~ pred
sim1 %>% ggplot() + geom_density(aes(resid)) +
  stat_function(fun=dnorm,color="red",args=list(mean=0,sd=1))    #density(resid)
sim1 %>% ggplot() + geom_hline(aes(yintercept=0),size=2,color="white") +
  geom_point(aes(x,resid))                                       #scatterplot res ~ x
sim1 %>% ggplot() + geom_qq_line(aes(sample=resid),size=2,color="white") +
  geom_qq(aes(sample=resid)) +
  labs(x="Theoretical Quantiles",y="Sample Quantiles")           #qqplot(resid)
sim1 %>% ggplot() + geom_point(aes(resid,c(resid[-1],NA))) +
  labs(x=expression(resid[i-1]),y=expression(resid[i]))          #lagplot(resid)
```

Gestão de projectos

» Boas práticas

- Estrutura de pastas;

- Ficheiro README (e nomeação de ficheiros).

Gestão de projectos

» 1. Estrutura de pastas

- Projects
  - 2017
    - 2017.01.01_ProjectName
    - 2017.04.01_ProjectName
    - 2017.10.31_ProjectName
  - 2018
    - 2018.01.01_ProjectName
    - 2018.12.25_ProjectName
  - 2019
    - 2019.10.18_RCD2019
      - bin
      - docs
      - results
        - exploration
        - modelling
        - visualization

Gestão de projectos

» 2. Ficheiro README

```
#autor:      Joao Sollari Lopes
#local:      INE, Lisboa
#criado:     18.10.2019
#modificado: 18.10.2019
+bin
  |exploration.r                       #exploracao de dados
  |install_packages.r                  #instalar pacotes necessarios
  |manipulation.r                      #manipulacao de dados
  |modelling.r                         #modelação de dados
  |visualization.r                     #visualizacao de dados
+docs
  |RCD2019_programa.pdf                #programa
  |RCD2019_slides.pdf                  #slides [versao final]
  |RCD2019_slides_short.pdf            #slides principais [versao final]
  |RCD2019_slides_20191014.pptx        #slides [v2019-10-14]
  |RCD2019_slides_short_20191014.pptx  #slides principais [v2019-10-14]
+results
  +exploration                         #resultados de "exploration.r"
  +modelling                           #resultados de "modelling.r"
  +visualization                       #resultados de "visualization.r"
README.txt                             #Este ficheiro
```

Comunidade R

» help!

- https://www.r-project.org/
- https://www.r-project.org/foundation/

- https://www.r-project.org/mail.html
- https://stackoverflow.com/
- https://www.r-bloggers.com/
- https://community.rstudio.com/
- comunidade R no INE

Bibliografia

» Azzalini A & Scarpa B (2012) Data Analysis and Data Mining - An Introduction. Oxford University Press, New York.

» Due KL & Swamy MNS (2016) Search and Optimization by Metaheuristics - Techniques and Algorithms Inspired by Nature. Springer, Switzerland.

» Gama J, Carvalho APL, Faceli K, Lorena AC e Oliveira M (2012) Extração de Conhecimento de Dados. *Data Mining*. Edições Sílabo. Lisboa.

» Larose DT & Larose CD (2014) Discovering Knowledge in Data – An Introduction to Data Mining. John Wiley & Sons, New Jersey.

» **Torgo L, (2017) Data Mining with R – Learning with Case Studies. Taylor & Francis Group, New York.**

» **Wickham H & Grolemund G (2017) R for Data Science. O'Reilly Media Inc., Sevastopol.**