Lição 3: Tipos de Variáveis

Para esta lição, vamos usar novamente o conjunto de dados DadosRT.csv. Então carregue inicialmente o pacote tidyverse, para ter acesso à função `read_csv()`.

```
library(tidyverse)
```

Defina como diretório de trabalho aquele que contém, em seu computador, o arquivo DadosRT.csv.

```
setwd("~/Dropbox/_R/swirl/Introducao_a_Estatistica_para_Linguistas/data")
```

E rode as linhas de comando no *script* com a função `read_csv()`, para carregar a planilha de dados na memória desta sessão.

```
dados <- read_csv("DadosRT.csv",
                  col_types = cols(.default = col_factor(),
                                   cont.precedente = col_character(),
                                   ocorrencia = col_character(),
                                   cont.seguinte = col_character(),
                                   IDADE = col_integer(),
                                   INDICE.SOCIO = col_double(),
                                   FREQUENCIA = col_double()
                                   )
                  )
```

O R classifica variáveis em seis tipos diferentes. Para nossos propósitos, quatro deles são mais importantes: int (=integer), numeric, character e factor. Na Lição 1, vimos vetores numéricos (como `aleatorio` e `x`) e vetores de caracteres (como `z`). Ao carregar os dados da planilha DadosRT.csv, especificamos as variáveis como factor, character, integer e double. As variáveis int e factor relacionam-se com as variáveis numéricas e de caracteres, mas têm certas especificidades.

Aplique a função `str()` a `dados` para que possamos examinar as propriedades de cada variável.

```
str(dados)
```

```
## spec_tbl_df [9,226 × 20] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ VD                : Factor w/ 2 levels "retroflexo","tepe": 1 1
1 2 1 1 1 1 2 1 ...
##  $ PARTICIPANTE      : Factor w/ 118 levels "IvanaB","HeloisaS",..:
1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ SEXO.GENERO       : Factor w/ 2 levels “feminino”,”masculino”: 1
1 1 1 1 1 1 1 1 1 ...
##  $ IDADE             : int [1:9226] 30 30 30 30 30 30 30 30 30 30 .
..
##  $ FAIXA.ETARIA      : Factor w/ 3 levels “1a”,”3a”,”2a”: 1 1 1 1 1
1 1 1 1 1 ...
##  $ ESCOLARIDADE      : Factor w/ 3 levels “fundamental”,..: 1 1 1 1
1 1 1 1 1 1 ...
##  $ REGIAO            : Factor w/ 2 levels “periferica”,”central”: 1
1 1 1 1 1 1 1 1 1 ...
##  $ INDICE.SOCIO      : num [1:9226] 2 2 2 2 2 2 2 2 2 2 ...
##  $ ORIGEM.PAIS       : Factor w/ 5 levels “mista”,”SPcapital”,..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ CONT.FON.PREC     : Factor w/ 7 levels “a”,”o”,”e”,”0”,..: 1 2 3
3 1 1 4 1 4 5 ...
##  $ CONT.FON.SEG      : Factor w/ 19 levels “ts”,”n”,”g”,”v”,..: 1 2
3 4 5 6 7 7 5 8 ...
##  $ TONICIDADE        : Factor w/ 2 levels “tonica”,”atona”: 1 2 2 2
1 1 1 1 2 1 ...
##  $ POSICAO.R         : Factor w/ 2 levels “medial”,”final”: 1 1 1 1
2 2 1 1 1 1 ...
##  $ CLASSE.MORFOLOGICA: Factor w/ 6 levels “substantivo”,..: 1 1 1 1
1 1 1 1 2 1 ...
##  $ FREQUENCIA        : num [1:9226] 1.34 0.16 0.22 0.44 1.94 1.94 0
.35 0.03 5.98 0.16 ...
##  $ ESTILO            : Factor w/ 4 levels “conversacao”,..: 1 1 1 1
1 1 1 1 1 1 ...
##  $ ITEM.LEXICAL      : Factor w/ 1151 levels “parte”,”jornal”,..: 1
2 3 4 5 5 6 7 8 9 ...
##  $ cont.precedente   : chr [1:9226] “do CEU é daquela” “eu não gost
o de” “não (es)to(u) entendendo a” “o que sei... num” ...
##  $ ocorrencia        : chr [1:9226] “parte <R>“ “jornal <R>“ “pergu
nta <R>“ “serviço <T>“ ...
##  $ cont.seguinte     : chr [1:9226] “que as perua(s) ia” “D1: mas d
igo assim” “D1: o que que/” “a não ser <A>“ ...
##  - attr(*, “spec”)=
##   .. cols(
##   ..   .default = col_factor(),
##   ..   VD = col_factor(levels = NULL, ordered = FALSE, include_na =
FALSE),
##   ..   PARTICIPANTE = col_factor(levels = NULL, ordered = FALSE, in
clude_na = FALSE),
##   ..   SEXO.GENERO = col_factor(levels = NULL, ordered = FALSE, inc
lude_na = FALSE),
##   ..   IDADE = col_integer(),
##   ..   FAIXA.ETARIA = col_factor(levels = NULL, ordered = FALSE, in
clude_na = FALSE),
##   ..   ESCOLARIDADE = col_factor(levels = NULL, ordered = FALSE, in
clude_na = FALSE),
##   ..   REGIAO = col_factor(levels = NULL, ordered = FALSE, include_
na = FALSE),
##   ..   INDICE.SOCIO = col_double(),
##   ..   ORIGEM.PAIS = col_factor(levels = NULL, ordered = FALSE, inc
lude_na = FALSE),
```

```
##   ..   CONT.FON.PREC = col_factor(levels = NULL, ordered = FALSE, i
nclude_na = FALSE),
##   ..   CONT.FON.SEG = col_factor(levels = NULL, ordered = FALSE, in
clude_na = FALSE),
##   ..   TONICIDADE = col_factor(levels = NULL, ordered = FALSE, incl
ude_na = FALSE),
##   ..   POSICAO.R = col_factor(levels = NULL, ordered = FALSE, inclu
de_na = FALSE),
##   ..   CLASSE.MORFOLOGICA = col_factor(levels = NULL, ordered = FAL
SE, include_na = FALSE),
##   ..   FREQUENCIA = col_double(),
##   ..   ESTILO = col_factor(levels = NULL, ordered = FALSE, include_
na = FALSE),
##   ..   ITEM.LEXICAL = col_factor(levels = NULL, ordered = FALSE, in
clude_na = FALSE),
##   ..   cont.precedente = col_character(),
##   ..   ocorrencia = col_character(),
##   ..   cont.seguinte = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Como já vimos, por meio da função `str()`, o R mostra a estrutura de um objeto. Para um dataframe, depois de mostrar o número de linhas (=observações) e colunas (=variáveis), o R lista as variáveis identificadas por `$` e, para cada uma, reporta como a variável está classificada.

Como a variável IDADE (que indica a idade de cada falante em anos) está classificada?

- int
- num
- factor

Como a variável INDICE.SOCIO (que indica a categorização dos falantes dentro de um índice que vai de 0 a 5) está classificada?

- int
- num
- factor

Como a variável FREQUENCIA (que indica a proporção de determinado item lexical no *corpus*) está classificada?

- int

- num
- factor

IDADE está classificada como 'int' (= números inteiros), enquanto INDICE.SOCIO e FREQUENCIA estão classificadas como 'num'. A diferença do segundo para o primeiro é o fato de que 'num' admite valores numéricos com casas decimais. Fomos nós que informamos ao R que essas variáveis são numéricas no momento de carregar a planilha no dataframe `dados`. No segundo argumento de `read_csv()`, col_types, indicamos a variável IDADE como `col_integer()`, e INDICE.SOCIO e FREQUENCIA como `col_double()`. `double` especifica "dupla precisão", daí a leitura das casas decimais.

Como está classificada a variável FAIXA.ETARIA (que separa os falantes em três categorias: "1a", de 20 a 34 anos; "2a", de 35 a 59 anos; e "3a", de 60 ou mais anos)?

- int
- num
- factor

A variável FAIXA.ETARIA é um reagrupamento dos falantes de acordo com sua IDADE, em três categorias. No entanto, apesar de falarmos em "1a", "2a" ou "3a" faixas etárias, não se trata de uma variável verdadeiramente numérica. Por exemplo, um falante de 36 anos – da 2ª faixa etária – não tem o dobro de idade de um falante de 25 – da 1ª faixa etária, de modo que a relação entre 1 e 2 não é quantitativa, mas *qualitativa*. Contudo, o R não sabe disso. A variável FAIXA.ETARIA poderia ter sido codificada como "A", "B" e "C", ou como "1", "2" e "3". O R classificou FAIXA.ETARIA como um fator porque assim especificamos na função `read_csv()` – usando `col_factor()` com opção *default* para as colunas. Factor é um tipo específico de vetores de caracteres que têm uma propriedade adicional: fatores têm níveis que normalmente ocorrem ao menos uma vez.

Acesse os elementos da coluna FAIXA.ETARIA, digitando `dados$FAIXA.ETARIA`.

```
dados$FAIXA.ETARIA
```

N.B.: Resultado aqui omitido.

Veja que o R não só mostra os elementos dessa coluna, mas também indica, ao final, quais são os níveis (=levels) dessa variável: 1a, 3a e 2a. Acesse agora apenas o primeiro elemento da coluna FAIXA.ETARIA.

```
dados$FAIXA.ETARIA[1]
```

```
## [1] 1a
## Levels: 1a 3a 2a
```

Veja que, mesmo mostrando um único elemento, o R mostra quais são todos os níveis possíveis da variável, e isso porque a variável FAIXA.ETARIA está classificada como factor, e não como character.

Com a função `read_csv()` do pacote tidyverse, é o próprio usuário que define o tipo de cada variável de seu conjunto de dados. A definição adequada para cada variável é importante porque sua natureza determina que tipos de operações e análises podem ser feitas sobre cada uma delas. Por exemplo, para variáveis numéricas (int e num), é possível calcular médias e medianas; para variáveis fatoriais, é possível calcular frequências (como vimos com a função `summary()` na última lição). O R não sabe quais operações são adequadas para cada tipo de dado – cabe a você determinar isso.

Você deve ter notado que o R está mostrando os níveis da variável FAIXA.ETARIA numa ordem "não lógica": 1a, 3a, 2a. Quando carregamos variáveis como factor com a função `read_csv()`, o *default* é que elas sejam ordenadas na mesma sequência em que aparecem na planilha. Contudo, a variável FAIXA.ETARIA, além de ser uma variável nominal/qualitativa (não numérica), é também uma variável ordinal: seus níveis se organizam hierarquicamente.

Novamente, esta é uma informação que o R não tem, pois somente o pesquisador pode determinar a natureza de uma variável.

Das variáveis de `dados`, quais outras também são ordinais?

- ESCOLARIDADE e FREQUENCIA
- REGIAO e CONT.FON.PREC
- ORIGEM.PAIS e CLASSE.MORFOLOGICA

Você pode ter achado estranho FREQUENCIA ser uma variável ordinal, uma vez que ela é numérica. Entretanto, toda variável numérica é também ordinal – embora nem toda variável ordinal seja numérica! E toda variável ordinal também pode ser nominal – embora nem toda variável nominal seja ordinal!

A ordem dos níveis de uma variável fatorial vai ser importante adiante na hora de plotar gráficos e ver o resultado de modelos estatísticos. Então, logo após importar os dados de uma planilha, é importante checar como o R está organizando os níveis dessas variáveis e, se necessário, reordená-los.

Um modo de checar os níveis de uma variável (além do que fizemos mais acima) é aplicar a função `levels()` a um vetor. Aplique essa função à coluna FAIXA.ETARIA de `dados`.

```
levels(dados$FAIXA.ETARIA)
```

```
## [1] "1a" "3a" "2a"
```

Entretanto, quando se tem um dataframe complexo, como é `dados`, pode ser muito trabalhoso aplicar essa função coluna a coluna. Um modo mais fácil de fazer isso para múltiplas colunas é aplicando a função `lapply()`, que toma como primeiro argumento o objeto a ser manipulado e como segundo argumento o nome da função que se quer aplicar. Aplique então a função `lapply()` a `dados` para executar a função levels em todas as suas colunas.

```
lapply(dados, levels)
```

```
## $VD
## [1] "retroflexo" "tepe"
##
## $PARTICIPANTE
##   [1] "IvanaB"      "HeloisaS"    "DercyF"      "SergioP"
##   [5] "AmandaA"     "PamelaR"     "PatriciaT"   "GiovanaA"
##   [9] "EdnaS"       "InesC"       "GilsonS"     "AdolfoF"
## [...]
##
## $SEXO.GENERO
## [1] "feminino"  "masculino"
##
## $IDADE
## NULL
##
## $FAIXA.ETARIA
```

```
## [1] “1a” “3a” “2a”
##
## $ESCOLARIDADE
## [1] “fundamental” “superior”    “medio”
##
## $REGIAO
## [1] “periferica” “central”
##
## $INDICE.SOCIO
## NULL
##
## $ORIGEM.PAIS
## [1] “mista”       “SPcapital”   “interior”    “nordeste”
## [5] “estrangeira”
##
## $CONT.FON.PREC
## [1] “a” “o” “e” “0” “3” “u” “i”
##
## $CONT.FON.SEG
##  [1] “ts” “n”  “g”  “v”  “k”  “dz” “t”  “s”  “d”  “#”  “m”  “l”  “p
”
## [14] “f”  “j”  “b”  “z”  “h”  “x”
##
## $TONICIDADE
## [1] “tonica” “atona”
##
## $POSICAO.R
## [1] “medial” “final”
##
## $CLASSE.MORFOLOGICA
## [1] “substantivo” “conj.prep”   “adverbio”    “verbo”
## [5] “adjetivo”    “morf.inf”
##
## $FREQUENCIA
## NULL
##
## $ESTILO
## [1] “conversacao” “palavras”    “jornal”      “depoimento”
##
## $ITEM.LEXICAL
##    [1] “parte”            “jornal”           “pergunta”
##    [4] “serviço”          “lugar”            “porta”
##    [7] “Marta”            “porque”           “comércio”
##   [10] “absurdo”          “melhor”           “suportei”
##   [13] “perdia”           “quarta”           “concordo”
## [...]
##
## $cont.precedente
## NULL
##
## $ocorrencia
## NULL
##
```

```
## $cont.seguinte
## NULL
```

N.B.: Resultado aqui abreviado.

O resultado é extenso, pois trata-se de um dataframe com 20 colunas! Como já sabíamos, a variável FAIXA.ETARIA não está ordenada de modo "lógico". Para reordenar os níveis de uma variável fatorial, podemos usar a função `fct_relevel()`, do pacote forcats, que faz parte do conjunto de pacotes do tidyverse (já carregado no início da lição). Aplique essa função à coluna FAIXA.ETARIA, usando sort como segundo argumento, para colocarmos os níveis em ordem alfabética: 1a, 2a e 3a. Além disso, guarde o resultado da operação novamente em `dados$FAIXA.ETARIA`, para que o dataframe seja atualizado.

```
dados$FAIXA.ETARIA <- fct_relevel(dados$FAIXA.ETARIA, sort)
```

Quer saber se o R fez a devida modificação nos níveis de FAIXA.ETARIA? Cheque novamente com a função `levels()`, voltando a essa linha de comando com a flecha para cima.

```
levels(dados$FAIXA.ETARIA)
## [1] "1a" "2a" "3a"
```

Veja agora, no resultado de `lapply()`, a ordem dos níveis de ESCOLARIDADE. Eles estão incorretos, não é? Por coincidência, a ordem "lógica" – fundamental, medio, superior – também segue a ordem alfabética, de modo que também podemos aplicar `fct_relevel()` com sort para reordená-los. Faça isso agora, guardando o resultado da linha de comando em `dados$ESCOLARIDADE`.

```
dados$ESCOLARIDADE <- fct_relevel(dados$ESCOLARIDADE, sort)
```

Reveja agora os níveis da variável VD no resultado de `lapply()` mais acima. Os níveis estão organizados como "retroflexo" e "tepe", a ordem em que aparecem na planilha. Bem mais adiante, na Lição 14, veremos que os resultados de modelos de regressão logística são fornecidos de acordo com o segundo nível da variável resposta – que, no momento, é o tepe! Digamos que um pesquisador tenha interesse em visualizar

os resultados de acordo com a variante retroflexa. Para obter esse resultado, é necessário mudar a ordem dos níveis dessa variável.

Nesse caso, os níveis já estão (coincidentemente) em ordem alfabética, de modo que não faz sentido usar o argumento sort. Outro modo de aplicar a função `fct_relevel()` é especificando, como segundo argumento, qual é o nível que se quer como primeiro. O nível especificado deve vir entre aspas. Faça isso agora, guardando o resultado em `dados$VD`.

```
dados$VD <- fct_relevel(dados$VD, "tepe")
```

Faça a mesma operação que fizemos acima com a variável ORIGEM.PAIS, colocando "SPcapital" como primeiro nível dessa variável. Não se esqueça de guardar o resultado no dataframe!

```
dados$ORIGEM.PAIS <- fct_relevel(dados$ORIGEM.PAIS, "SPcapital")
```

No caso da variável VD, só há duas variantes/dois níveis. Em ORIGEM.PAIS, há vários outros níveis – mista, interior, nordeste, estrangeira – mas, ao colocar SPcapital como primeiro nível, estamos indicando que a ordem dos demais não importa, e que – por motivos que ficarão mais claros na Lição 14 – interessa apenas ter "SPcapital" como referência.

Em outros casos, no entanto, podemos querer redefinir a ordem de todos os níveis. Tomemos como exemplo a variável CONT.FON.PREC, que codifica qual vogal ocorreu antes do /r/ em coda. Seus níveis são as sete vogais orais do português brasileiro: a e ɛ i o ɔ u. As vogais /ɛ/ e /ɔ/ foram codificadas como "3" e como "0", respectivamente, para evitar o uso de caracteres especiais. Embora CONT.FON.PREC não seja uma variável ordinal, pode fazer mais sentido ordenar as vogais de acordo com algum critério fonológico, como o traço de anterioridade – de [+anterior] a [-anterior], de modo que seus níveis sejam i e 3 a 0 o u.

Aqui, não podemos usar o argumento sort, tampouco basta especificar apenas o primeiro nível da variável. Para casos como esse, podemos especificar todos os níveis como diferentes argumentos da função `fct_relevel()`, como a linha de comando já pronta no *script*. Rode-a agora.

```
dados$CONT.FON.PREC <- fct_relevel(dados$CONT.FON.PREC,
                                   "i", "e", "3", "a", "0", "o", "u")
```

Os níveis da variável ESTILO são "conversacao", "palavras", "jornal" e "depoimento". Eles codificam de que ponto da entrevista sociolinguística o dado foi extraído, de acordo com a hipótese laboviana de "grau de atenção à fala". Do modo como o roteiro da entrevista foi elaborado, espera-se que o falante esteja prestando menos atenção durante a conversação com o documentador, e que preste progressivamente mais atenção em três estilos de leitura: depoimento – que é a leitura de um texto informal –, jornal – a leitura de uma notícia – e palavras – a leitura de palavras isoladas.

Aqui, coincidentemente, a ordem de "grau de atenção" é a mesma que a ordem alfabética, o que significa que poderíamos usar o argumento sort. No entanto, como nem sempre temos essa sorte, vale aplicar a função `fct_relevel()` com especificação explícita dos níveis, apenas para treino. Digite a linha de comando com os níveis na ordem desejada, sem esquecer de guardar o resultado.

```
dados$ESTILO <- fct_relevel(dados$ESTILO,
                            "conversacao", "depoimento", "jornal",
                            "palavras")
```

Com a função `fct_relevel()`, estamos reorganizando os níveis das variáveis em diferentes linhas de comando. No entanto, também é possível fazer isso no momento da importação de dados, com a função `read_csv()`. Examine a linha de comando abaixo. Ela é semelhante ao comando que usamos mais acima, mas, desta vez, também estamos especificando a ordem dos níveis de FAIXA.ETARIA com o argumento levels da função `col_factor()`. Rode essa linha de comando.

```
dados <- read_csv("DadosRT.csv",
                  col_types = cols(.default = col_factor(),
                                   cont.precedente = col_character(),
                                   ocorrencia = col_character(),
                                   cont.seguinte = col_character(),
                                   IDADE = col_integer(),
                                   INDICE.SOCIO = col_double(),
                                   FREQUENCIA = col_double(),
                                   FAIXA.ETARIA = col_factor(levels =
c("1a",  "2a", "3a"))
                                   )
                  )
```

E cheque a ordem dos níveis da variável FAIXA.ETARIA com a função `levels()`.

```
levels(dados$FAIXA.ETARIA)

## [1] “1a” “2a” “3a”
```

Nesse último exemplo, redefinimos apenas os níveis de FAIXA.ETARIA, mas é possível fazer isso com todas as variáveis fatoriais simplesmente acrescentando novos argumentos na função `cols()`. Volte, então, à mesma linha de comando com `read_csv()`. Ao final da linha com redefinição dos níveis de FAIXA.ETARIA, inclua uma vírgula (para poder acrescentar um novo argumento) e pressione ENTER para mudar de linha. Depois, redefina os níveis de ESCOLARIDADE para “fundamental”, “medio”, “superior”.

```
dados <- read_csv(“DadosRT.csv”,
                  col_types = cols(.default = col_factor(),
                                   cont.precedente = col_character(),
                                   ocorrencia = col_character(),
                                   cont.seguinte = col_character(),
                                   IDADE = col_integer(),
                                   INDICE.SOCIO = col_double(),
                                   FREQUENCIA = col_double(),
                                   FAIXA.ETARIA = col_factor(levels =
c(“1a”,  “2a”, “3a”)),
                                   ESCOLARIDADE = col_factor(levels =
c(“fundamental”, “medio”, “superior”))
                                   )
                  )
```

Pode parecer trabalhoso redefinir os níveis de todas as variáveis fatoriais desse modo, em uma única linha de comando, em vez de fazê-lo passo a passo, como fizemos com `fct_relevel()`. Um dos motivos disso é que, para produzi-la, você precisa saber quais são todas as variáveis fatoriais de seus dados, quais são os níveis de cada variável e em qual ordem você os quer. Entretanto, a principal dificuldade que você está sentindo aqui se deve ao fato de que este não é o *seu* conjunto de dados. Quando estiver trabalhando com sua própria planilha, você certamente saberá quais são as variáveis e suas respectivas variantes!

Além disso, essa é a grande vantagem de se trabalhar com *scripts*: não é necessário digitar uma linha de comando completa de primeira. Você pode, por exemplo, importar primeiramente os dados e checar os níveis com `lapply()`, como fizemos mais acima. Ao

verificar que os níveis de algumas variáveis precisam ser redefinidos, você pode voltar à linha de comando já digitada, acrescentar um novo argumento e rodá-la novamente. No fundo, o trabalho de digitar mais um argumento em `read_csv()` é o mesmo de digitar várias linhas de comando separadas com `fct_relevel()`!

Para aproveitar que estamos no mundo do pacote forcats, vale a pena apresentar uma função para amalgamar os níveis de uma variável fatorial, algo que o pesquisador pode querer fazer por motivos vários: porque não há dados suficientes para um dos níveis, porque não há diferença relevante entre certos níveis, porque faz sentido dentro da teoria linguística.

Para exemplificar, vamos usar a variável CONT.FON.SEG. De modo semelhante a CONT.FON.PREC, a variável codifica o contexto fonológico de ocorrência de /r/ em coda – neste caso, qual é a consoante que vem em seguida, ou ainda se é uma pausa. Essa variável foi originalmente codificada de modo detalhado, pois é fácil juntar fatores, mas dá mais trabalho separá-los (seria necessário recodificá-los um a um!). Seguindo a hipótese que se quer testar, o contexto fônico seguinte pode ser reorganizado de acordo com o ponto de articulação, ou modo de articulação, ou a sonoridade do segmento, ou o Ponto de C etc.

Para juntar fatores em um mesmo nível, vamos usar a função `fct_collapse()`, que pode ser vista no *script*. Assim como a função `fct_relevel()`, o primeiro argumento é o vetor/coluna que se quer modificar. Os demais argumentos devem especificar, um a um, o nome do novo nível à esquerda do sinal de igual, e os valores a amalgamar à direita do sinal de igual. Note o uso da função `c()` para juntar os fatores; quando há apenas um fator, como no caso da categoria `pausa`, não é necessário aplicar a função de concatenação. Isso também indica que a função pode ser usada para renomear os níveis de uma variável. Observe também que, em vez de guardar o resultado num vetor de mesmo nome, estamos criando uma nova coluna, chamada `CONT.FON.SEG_PC`, e inserindo-a no dataframe. Isso pode ser interessante para preservar a variável original. Rode então essa linha de comando, que já está pronta.

```
dados$CONT.FON.SEG_PC <- fct_collapse(dados$CONT.FON.SEG,
                                      coronal = c("ts", "dz", "t", "d"
, "n", "s", "z", "x", "j", "l"),
                                      dorsal = c("k", "g", "h"),
                                      labial = c("f", "v", "p", "b", "
m"),
                                      pausa = "#")
```

Você pode ver, em Environment, que o dataframe `dados`, antes com 20 colunas, agora tem 21. A variável CONT.FON.SEG_PC foi adicionada como última coluna do dataframe.

Agora é a sua vez! Vamos criar uma nova variável a partir de CONT.FON.PREC, amalgamando as vogais precedentes em dois níveis, definidos pela altura da vogal. A estrutura desse comando está no *script*, mas você deve completá-lo para que o R possa interpretá-lo corretamente. A nova variável já está definida: `dados$CONT.FON.PREC_altura`. Insira nos pontos devidos as seguintes informações: (i) o vetor original cujos níveis serão amalgamados; (ii) o nível alta, com os fatores i, e, u e o (nessa ordem); e (iii) o nível baixa, com os fatores 3, a e 0 (nessa ordem).

```
dados$CONT.FON.PREC_altura <- fct_collapse(dados$CONT.FON.PREC,
                                           alta = c("i", "e", "u", "o"
),
                                           baixa = c("3", "a", "0")
                                           )
```

Nesta lição, vimos que o R classifica as variáveis de um dataframe em int, num, chr ou factor. A verdadeira natureza de cada variável é determinada pelo pesquisador e deve ser levada em conta no momento de codificação e importação dos dados ao R. Também é possível redefinir os níveis de uma variável fatorial usando a função `fct_relevel()`, e juntar fatores com a função `fct_collapse()`, ambas do pacote forcats/tidyverse.

Para saber mais

Recomendo a leitura de Wickham (2014) e as páginas 28–31 de Gries (2019) sobre a coleta e a organização de dados.

Exercícios

A planilha de dados é normalmente preparada fora do ambiente do R, em programas como o Excel ou o Calc. O modo como cada variável é codificada na planilha e importada ao R depende de escolhas do pesquisador e da natureza das variáveis sob análise. Com isso em mente, responda as questões a seguir.

1. Qual é o melhor modo de codificar a variável IDADE, a ser lida pelo R como uma variável como numérica/int?
    a. 20 anos, 21 anos, 22 anos...
    b. 20, 21, 22...
    c. vinte, vinte e um, vinte e dois...
2. Caso um pesquisador codifique a variável CLASSE.SOCIAL como baixa, media.baixa, media etc., como a função `read_csv()` vai ler essa coluna, sem qualquer especificação adicional?
    a. como character
    b. como int
    c. como num
3. Como a variável CLASSE.SOCIAL codificada como baixa, media.baixa, media etc. deve ser especificada em `col_types()`, dentro de `read_csv()`?
    a. `col_character()`
    b. `col_double()`
    c. `col_factor()`
    d. `col_integer()`
4. Qual é a ordem em que `read_csv()`, do pacote tidyverse, colocará os níveis de CLASSE.SOCIAL, se codificada como baixa, media.baixa, media, media.alta, alta?
    a. na ordem em que aparecem na planilha

   b. na ordem lógica – baixa, media.baixa, media, media.alta, alta
   c. na ordem alfabética – alta, baixa, media, media.alta, media.baixa

5. Rode a linha de comando a seguir e visualize o vetor CLASSE.SOCIAL.

```
CLASSE.SOCIAL <- as.factor(c("alta", "baixa",
                             "media", "media_baixa", "media_alta"))
```

6. Reorganize os níveis da variável CLASSE.SOCIAL numa ordem "lógica", de classe mais baixa para classe mais alta. Use a função `fct_relevel()`. Não é necessário guardar o resultado.
7. Reorganize os níveis da variável CLASSE.SOCIAL, juntando os fatores relativos à classe média (media_baixa, media e media_alta) em um único fator chamado "media". Guarde o resultado em um vetor chamado CLASSE.SOCIAL2.
8. Reorganize os níveis da variável CLASSE.SOCIAL2 na ordem lógica ascendente: baixa, media e alta. Não é necessário guardar o resultado.
9. Costuma-se classificar variáveis em 3 tipos: nominais (ou categóricas); ordinais; e numéricas (Gries, 2019, p.25–26). O R as classifica de modo diferente (int, num, double, factor), mas há uma relação entre elas. Escolha a opção que *não* faz uma relação correta.
   a. categórica – factor
   b. categórica – num
   c. numérica – double
   d. numérica – int
   e. numérica – num
   f. ordinal – factor, int ou num
10. Que tipo de variável é CLASSE.MORFOLOGICA (codificada como substantivo, adjetivo, verbo etc.)?
   a. nominal
   b. numérica
   c. ordinal

11. Como a variável CLASSE.MORFOLOGICA (codificada como "substantivo", "adjetivo", "verbo" etc.) deve ser especificada em `col_types()`, dentro de `read_csv()`?
    a. `col_character()`
    b. `col_double()`
    c. `col_factor()`
    d. `col_integer()`
12. Qual das categorizações a seguir não se aplica a FREQUENCIA.LEXICAL, codificada como muito.frequente, frequente, pouco.frequente, infrequente?
    a. categórica
    b. nominal
    c. numérica
    d. ordinal
13. Como a variável FREQUENCIA.LEXICAL (codificada como muito.frequente, frequente, pouco.frequente, infrequente) deve ser especificada em `col_types()`, dentro de `read_csv()`?
    a. `col_character()`
    b. `col_double()`
    c. `col_factor()`
    d. `col_integer()`
14. Qual das categorizações a seguir não se aplica a FREQUENCIA.LEXICAL, codificada como o número de ocorrências por mil palavras (0,01, 0,02, 0,03, 0,5, 1,0 etc.)?
    a. nominal
    b. numérica
    c. ordinal
15. Como a variável FREQUENCIA.LEXICAL, codificada como o número de ocorrências por mil palavras (0,01, 0,02, 0,03, 0,5, 1,0 etc.) deve ser especificada em `col_types()`, dentro de `read_csv()`?
    a. `col_character()`
    b. `col_double()`
    c. `col_factor()`

   d. `col_integer()`

16. Qual das categorizações a seguir não se aplica a F1, codificada como as medições de F1 em Hertz de diferentes vogais (300, 325, 450 etc.)?
    a. nominal
    b. numérica
    c. ordinal
17. Como a variável F1, codificada como as medições de F1 em Hertz de diferentes vogais (300, 325, 450 etc.), deve ser especificada em `col_types()`, dentro de `read_csv()`?
    a. `col_character()`
    b. `col_factor()`
    c. `col_integer()`
18. Defina como diretório de trabalho aquele que contém a planilha DadosRT.csv em seu computador.
19. Carregue o pacote tidyverse.
20. Use a função `read_csv()` para carregar a planilha DadosRT.csv em um dataframe chamado `dadosRT`. Ao importar os dados: (i) especifique o *default* das variáveis como `col_factor()`; (ii) especifique a variável IDADE como `col_integer()`; (iii) especifique as variáveis INDICE.SOCIO e FREQUENCIA como `col_double()`; e defina a ordem dos níveis da variável CONT.FON.PREC como "i", "e", "3", "a", "0", "o" e "u".
21. Aplique a função `levels()` para visualizar os níveis da variável POSICAO.R do dataframe `dadosRT`.
22. Reorganize os níveis da variável POSICAO.R na ordem "final" e "medial". Não é necessário guardar o resultado.
23. Aplique a função `levels()` para visualizar os níveis da variável TONICIDADE do dataframe `dadosRT`.
24. Reorganize os níveis da variável TONICIDADE na ordem "atona" e "tonica". Não é necessário guardar o resultado.

25. Aplique a função `levels()` para visualizar os níveis da variável ORIGEM.PAIS do dataframe `dadosRT`.
26. Reorganize os níveis da variável ORIGEM.PAIS na ordem de maior para menor proximidade da cidade de São Paulo, com o nível "mista" (= pai e mãe nascidos em locais distintos) por último. Não é necessário guardar o resultado.