胡伟老师的绘图尝试

夏骁凯

目录

```
knitr::opts_chunk$set(tidy = FALSE)
```

对胡伟老师的实验进行可视化尝试，使用 ggplot2 包进行绘图，colorspace 包进行颜色选择。

0.0.1 1. 数据结构的实现：

考虑 ggplot2 需要数据框作为数据数据结构，同时选择 geom_tile 作为绘图函数，涉及到的 aes 包括 x，y，fill，因而设计的数据框结构如下：

x	y	fill
1	1	#000000
1	2	#000001

其中，x，y 分别表示横纵坐标，fill 为 HCL 色值。同时，为了便于程序随后的修改与扩展性，因而在本数据框加入数据分析的数据源，包括对不同单词出现频率，以及使用条件判断语句决定是否需要混色的 factor。使用 wide-table 的形式表现，因此数据框结构如下：

x	y	heed	hid	hood	head	mix	fill
1	2	5	3	2	0	1	#000000

0.0.2 2. 混色的实现：

由于为了使不同的数据通过颜色混合的形式表达出来，因此需要工具对色彩进行分析与混合，预期的方法是将特定的词频的颜色进行指定，随后根据映射关系将相应的色彩进行混合。查询 CRAN 后选择 colorspace 完成相应任务。

colorspace 是一个进行调色的 R package。CRAN 上的地址为：https://cran.r-project.org/web/packages/colorspace/ 。该工具包提供 vignettes ，地址为：https://cran.r-project.org/web/packages/colorspace/vignettes/hcl-colors.pdf 。本文的色彩实现主要通过该 vignettes 。

在 colorspace 中，可以通过不同的 color palettes 实现配色，由于可视化的目标是将不同频率的击中用连续色彩显示，且相对立的词汇需要对立的颜色进行显示，因此选择 Diverging palettes 色系。

以下代码尝试实现：构造 41 个对立渐变的色彩 scale 并建立映射，随后进行呈现。

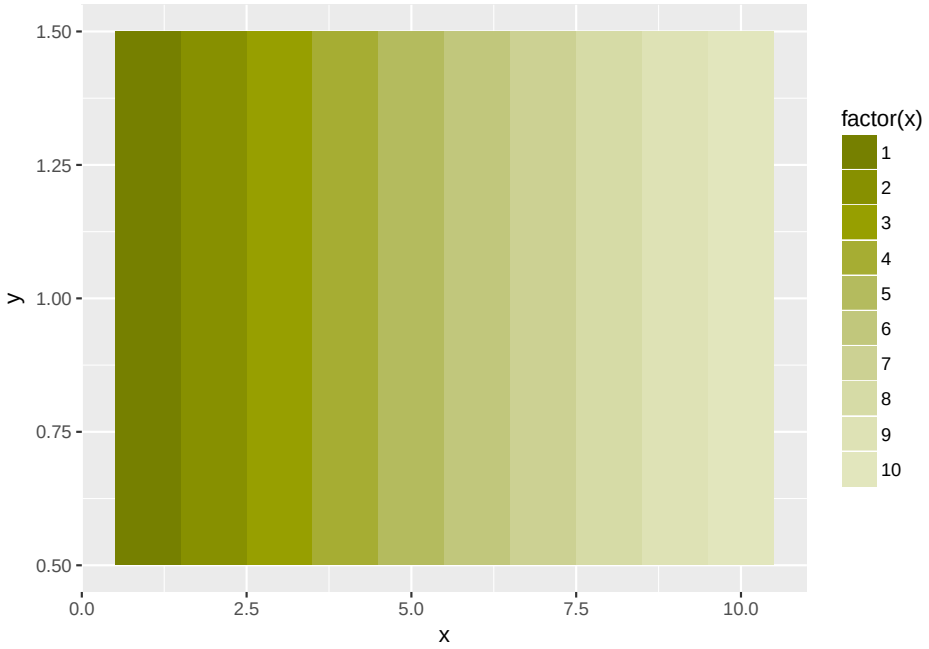
```
library(colorspace)
library(tidyverse)
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

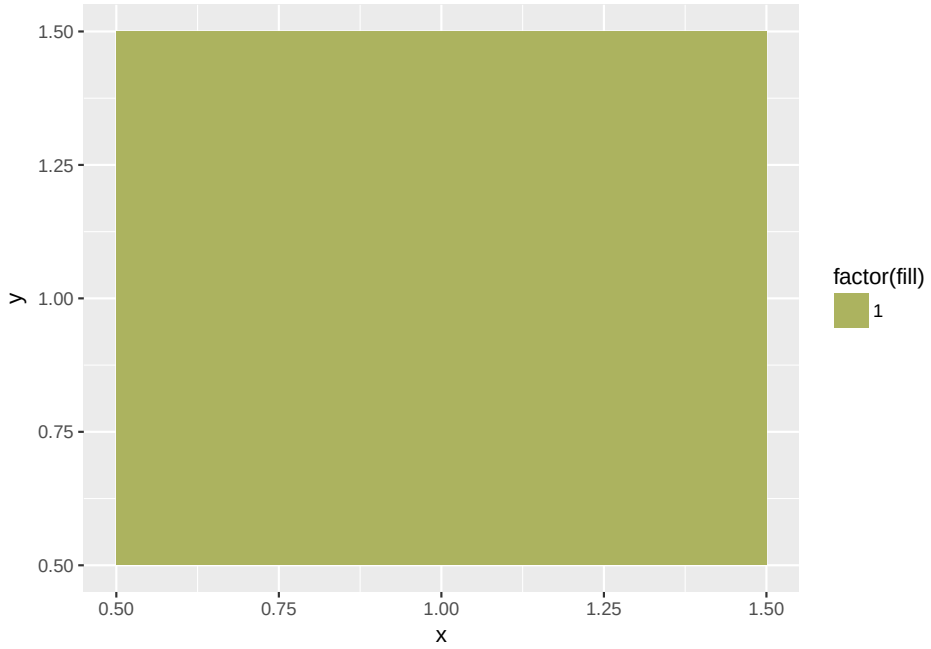
```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```
color <- sequential_hcl(10, h = 90, c = c(100, 30), l = c(50, 90))
colortable <- data.frame(x = 1:10, y = 1, color = color) # 构造绘图函数
names(color) <- factor(1:10) # 构造色彩映射
ggplot(colortable, aes(x = x, y = y, fill = factor(x))) + geom_tile() + scale_fill_manu
```




```
mixcolorhcl <- function(first, second) {
  hex(mixcolor(alpha = 0.5,
               color1 = hex2RGB(first), color2 = hex2RGB(second)))
}

mixcolor <- mixcolorhcl(color[1], color[10])
names(mixcolor) <- "1"
mixtable <- data.frame(x = 1, y = 1, fill = 1)
ggplot(mixtable, aes(x = x, y = y, fill = factor(fill))) + geom_tile() + scale_fill_man
```



0.0.3 3. 表格的制作

```
# 配置分析环境
library(stringr)

# 导入实验数据
en_nv <- read.csv("sub-data/English/nv.csv", header = T)
en_ihiy <- read.csv("sub-data/English/ihiy.csv", header = T)

# 对数据进行初步筛选与整理
en_nv <- select(en_nv, answer1.Block., sound111, sound112)
en_ihiy <- select(en_ihiy, answer1.SubTrial.,answer2.SubTrial.,starts_with("sound"))

names(en_nv)[1] <- "answer"
names(en_ihiy)[1] <- "answer"
names(en_ihiy)[2] <- "rate"

# 制作实验 1 的表格
```

```r
en_nv <- unite(en_nv, sound, sound111, sound112)
en_ihiy <- unite(en_ihiy, col = sound, sound111, sound112, sound113, sound114, sound115

# 清除缺失值
en_nv <- na.omit(en_nv)
en_ihiy <- na.omit(en_ihiy)

# 对数据中的文件路径含义进行修改

# 将实验 1 的文件路径改为对应单词，并确认是否匹配
en_nv$sound <- str_extract(en_nv$sound, pattern = "(w|h).*d")
en_nv <- mutate(en_nv, match = answer == sound)
table(en_nv$match, en_nv$answer)
```

```
##
##       had hawed hayed head heard heed hid hod hoed hood hud whod
##   FALSE  15    10     3   20    10   12   6   7   14   26  33    4
##   TRUE    4     4     2    8    20   14   5   4    3    4   4    0
```

```r
# 将实验 2 的文件路径改为定位坐标
en_ihiy$sound <- str_extract(en_ihiy$sound, pattern = "F1_..-F2_..")
en_ihiy <- separate(en_ihiy, col = sound, into = c("X","Y"), sep = "-")
en_ihiy$X <- as.numeric(str_replace(en_ihiy$X, pattern = "F1_", replacement = ""))
en_ihiy$Y <- as.numeric(str_replace(en_ihiy$Y, pattern = "F2_", replacement = ""))

# 对实验 2 中出现的单词数进行计数
summary(en_ihiy$answer)
```

```
##  had hawed hayed  head heard  heed   hid   hod  hoed  hood   hud  whod
##  185   145   249   504   381   370   398   187   213   162    80    55
```

0.0.4 4. 绘制 heatmap

```
select(en_ihiy, answer, X, Y) %>%
  reshape2::dcast(., X + Y ~ answer) -> data_en_ihiy
```

```
## Using Y as value column: use value.var to override.
```

```
## Aggregation function missing: defaulting to length
```

```
# 选择出现频率最高的几个单词作为绘图依据
data_en_ihiy <- select(data_en_ihiy, X, Y, heed, hid, hood, head)
data_en_ihiy[data_en_ihiy == 0] <- NA

# 构造色彩映射
heed_color <- sequential_hcl(10, h = 10, c = c(100, 30), l = c(50, 90))
names(heed_color) <- factor(str_c("heed",1:10))
hid_color <- sequential_hcl(10, h = 100, c = c(100, 30), l = c(50, 90))
names(hid_color) <- factor(str_c("hid",1:10))
hood_color <- sequential_hcl(10, h = 190, c = c(100, 30), l = c(50, 90))
names(hood_color) <- factor(str_c("hood",1:10))
head_color <- sequential_hcl(10, h = 280, c = c(100, 30), l = c(50, 90))
names(head_color) <- factor(str_c("head",1:10))
color_total <- c(heed_color, hid_color, hood_color, head_color)

# 制作绘图参考值的表格
data_en_ihiy <- mutate(data_en_ihiy, heedcolor = heed, hidcolor = hid, hoodcolor = hood
data_en_ihiy$heedcolor <- str_c("heed", data_en_ihiy$heedcolor)
data_en_ihiy$hidcolor <- str_c("hid", data_en_ihiy$hidcolor)
data_en_ihiy$hoodcolor <- str_c("hood", data_en_ihiy$hoodcolor)
data_en_ihiy$headcolor <- str_c("head", data_en_ihiy$headcolor)

# 统计是否需要颜色混合
count <- vector()
for(i in 1:nrow(data_en_ihiy)){
```

```
  count[i] <- sum(is.na(data_en_ihiy[i,3:6]) == FALSE)
}

data_en_ihiy <- cbind(data_en_ihiy, count)


# 建立整合色彩函数
select_one_color <- function(x){
  a <- select(x, heedcolor, hidcolor, hoodcolor, headcolor)
  b <- a[which(is.na(a) == FALSE)]
  return(color_total[[b]])
}

mix_two_color <- function(x){
  a <- select(x, heedcolor, hidcolor, hoodcolor, headcolor)
  b <- a[which(is.na(a) == FALSE)]
  c <- mixcolorhcl(color_total[[b[[1]]]], color_total[[b[[2]]]])
  return(c)
}

mix_three_color <- function(x){
  a <- select(x, heedcolor, hidcolor, hoodcolor, headcolor)
  b <- a[which(is.na(a) == FALSE)]
  c <- mixcolorhcl(color_total[[b[[1]]]], color_total[[b[[2]]]])
  d <- mixcolorhcl(c, color_total[[b[[3]]]])
  return(d)
}

mix_four_color <- function(x){
  a <- select(x, heedcolor, hidcolor, hoodcolor, headcolor)
  b <- a[which(is.na(a) == FALSE)]
  c <- mixcolorhcl(color_total[[b[[1]]]], color_total[[b[[2]]]])
  d <- mixcolorhcl(color_total[[b[[4]]]], color_total[[b[[3]]]])
```

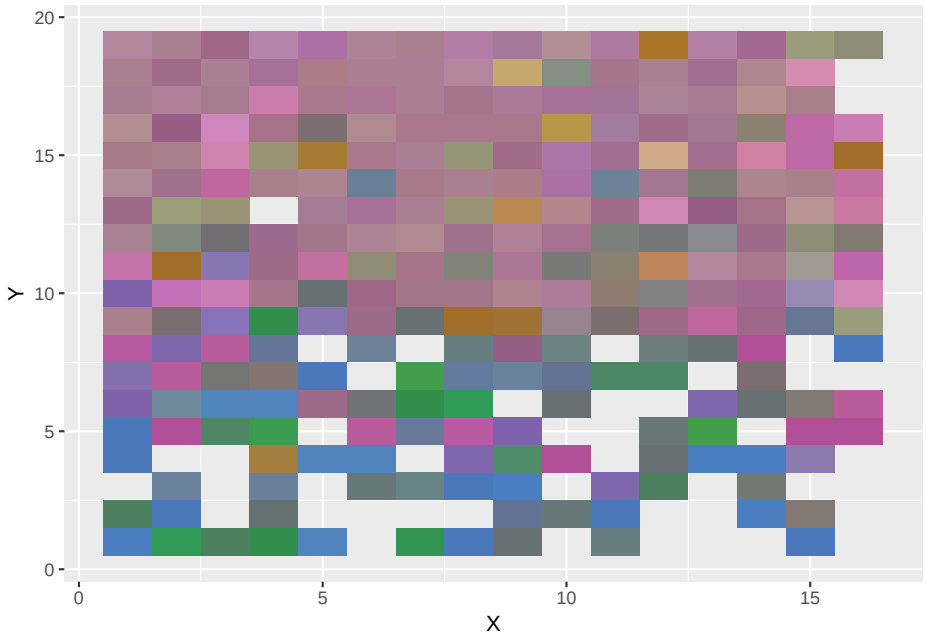
```
  e <- mixcolorhcl(c,d)
  return(e)
}

# 构建色彩映射向量
color_map <- vector()
data_en_ihiy <- filter(data_en_ihiy, count > 1)
for(i in 1 : nrow(data_en_ihiy)){
  z <- data_en_ihiy[i,]
  color_map[i] <- switch(z$count,
    select_one_color(z),
    mix_two_color(z),
    mix_three_color(z),
    mix_four_color(z)
  )
}

names(color_map) <- as.character(1:nrow(data_en_ihiy))

position <- c(1:nrow(data_en_ihiy))
data_en_ihiy <- cbind(data_en_ihiy, position)

ggplot(data_en_ihiy, aes(x = X, y = Y, fill = as.factor(position))) + geom_tile(show.le
```


20
15
10
5
0
Y
0
5
10
15
X