R language basics, part 3: factor
HUST Bioinformatics course series

Wei-Hua Chen (CC BY-NC 4.0)

03 August, 2021

section 1: TOC

前情提要

data frame and tibble
- declaration & usage
- manipulation （更多相关内容会在介绍 dplyr 时讲到）
- differences between data.frame and tibble
- advantages of using tibble (更多内容以后会介绍)
- with, within, attach, detach 等的用法

IO
- read from files of different formats
- write to files
- use GUI to read files (& get the corresponding code)

今次预报

1. IO, project management, working environment management
2. factors: R 中最重要的概念之一
3. exercises

section 2: IO and working enviroment management

R session 的概念

每个 R session 是一个单独的工作空间（work space），包含各自的数据、变量和操作历史。

Figure 1: two R sessions

R session in RStudio

Each RStudio session is automatically associated with a R session
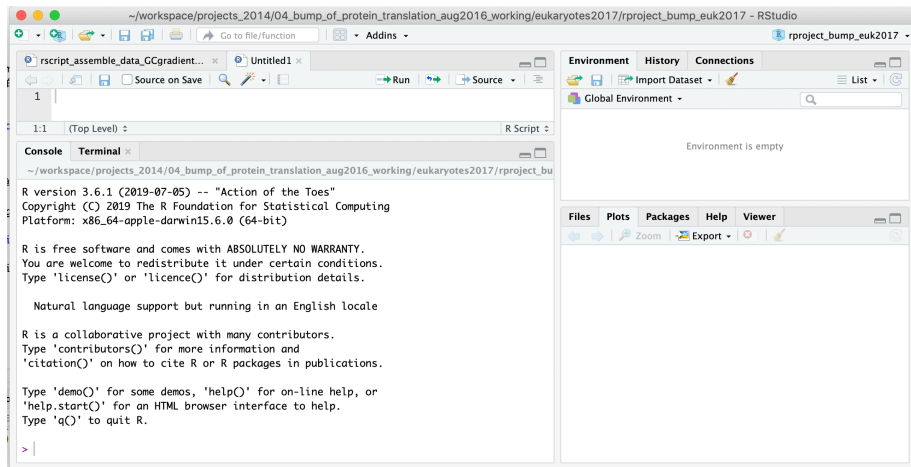
Figure 2: R session in RStudio

start a new RStudio session by creating a new project

1. 右上角的 Project 按钮，在弹出菜单里选 New Project ...
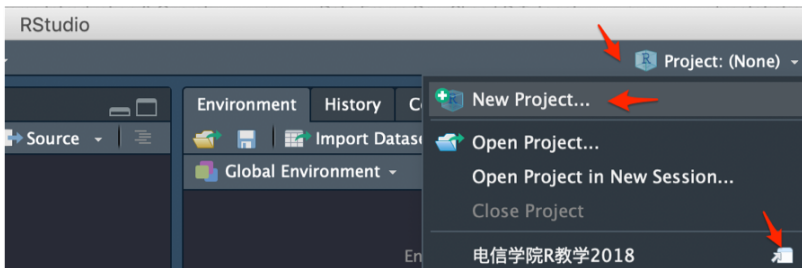
Figure 3: create new project, step 1

create a new project, cont.

2. Select: New directory -> New Project in the popup window
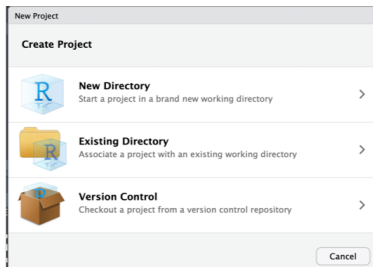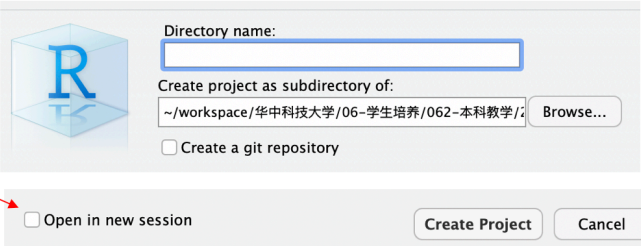
Figure 4: create new project, step 2

create a new project, cont.

③ Enter a new directory name, choose its mother directory …

Figure 5: create new project, step 3

现场演示

演示 ~~

working space

当前工作空间，包括所有已装入的数据、包和自制函数

可通过以下代码管理变量

```
ls();   ## 显示当前环境下所有变量
```

```
## [1] "color_block"
```

```
rm( x ); ## 删除一个变量
```

```
## Warning in rm(x): object 'x' not found
```

```
ls();
```

```
## [1] "color_block"
```

```
##rm(list=ls()); ## 删除当前环境下所有变量!!!
```

variables in working space in RStudio

在 RStudio 右上角的 "Environment" 窗口显示了所有当前工作间的变量
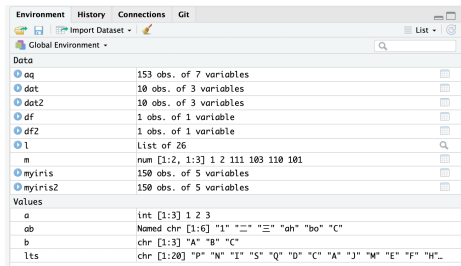
Figure 6: RStudio enviroment window

save and restore work space

```
## -- save all loaded variables into an external .RData file
save.image( file = "prj_r_for_bioinformatics_aug3_2019.RData" );

## -- restore ( load ) saved work space
load( file = "prj_r_for_bioinformatics_aug3_2019.RData" );
```

Notes

- existing variables will be kept, however, those will the same names will be replaced by loaded variables
- please consider using rm(list=ls()) to remove all existing variables to have a clean start
- you may need to reload all the packages

save selected variables

Sometimes you need to transfer processed data to a collaborator ...

```
## save selected variables to external
save(city, country, file="1.RData"); ## you can specify directory name

## --
load( "1.RData" );
```

close and (re)open a project

close a project is easy:
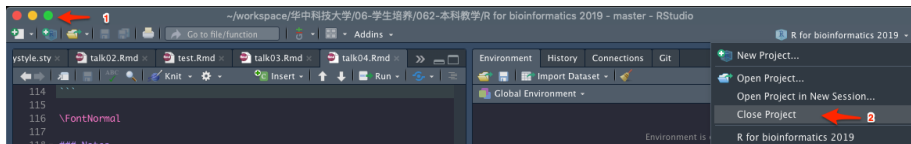
Figure 7: Two ways of closing a project

however ...

退出 projects 时的一些选项（RStudio）
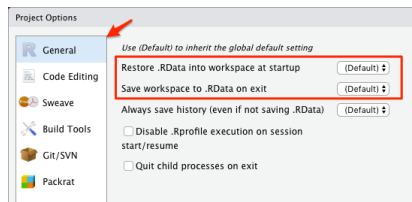
Figure 8: Project options

notes

- 退出时保存
- 打开时装入
- 但数据较大时，装入时间可能过长 …

open a project
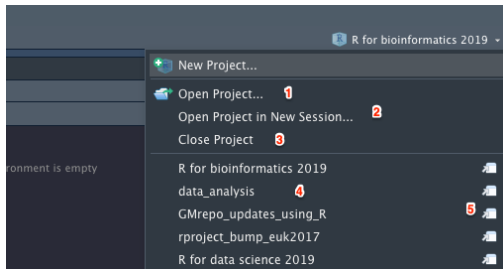
Figure 9: Open a project

演示项目的不同打开姿式（1-5）。

练习

- 创建一个项目
- 定义一些变量
- 从外部文件装入一些数据
- 保存 workspace 到.RData
- 退出 project
- 重新打开 project 并恢复 workspace

section 3: factors

什么是 **factors**？

Factor is a data structure used for fields that takes only predefined, finite number of values (categorical data).

Facor 用于限制某个字段（列），只允许其接受某些值

```r
x <- c("single", "married", "married", "single");
str(x);
```

```
##  chr [1:4] "single" "married" "married" "single"
```

```r
## create factor as it is ...
x <- as.factor(x);

## please note the change in the displayed values ...
str(x);
```

```
##  Factor w/ 2 levels "married","single": 2 1 1 2
```

```r
## create factor from scratch ...
x <- factor( c( "single", "married", "married", "single" ) );
str(x);
```

```
##  Factor w/ 2 levels "married","single": 2 1 1 2
```

factors, cont.

Factors 会限制输入数据的选择范围

```
str(x);
```

```
##  Factor w/ 2 levels "married","single": 2 1 1 2
```

```
x[ length(x) + 1 ] <- "widowed";
```

```
## Warning in `[<-.factor`(`*tmp*`, length(x) + 1, value = "widowed"): invalid
## factor level, NA generated
```

```
x;
```

```
## [1] single  married married single  <NA>
## Levels: married single
```

Use levels() function to add new factors

```
levels(x) <- c(levels(x), "widowed");
x[ length(x) + 1 ] <- "widowed";
str(x);
```

```
##  Factor w/ 3 levels "married","single",..: 2 1 1 2 NA 3
```

factors, cont.

Play around with levels():

```
## other ways of assigning factors ...
y <-  as.factor( c( "single", "married", "married", "single" ) );
levels( y );
```

```
## [1] "married" "single"
```

```
levels(y) <- c("single", "married", "widowed");
str(y);
```

```
##  Factor w/ 3 levels "single","married",..: 2 1 1 2
```

```
## 这个代码现在就没有问题了
y[ length(y) + 1 ] <- "widowed";
```

** 注意 ** 用 as.factor 创建 factor 时，得到的 levels 按字母表排列;

但是，用 levels(y) 方式指定 levels 时，则按照指定的顺序;

levels 的顺序决定了排序的顺序

```
##
y <-  as.factor( c( "single", "married", "married", "single" ) );
levels(y);
```

```
## [1] "married" "single"
```

```
sort(y);
```

```
## [1] married married single  single
## Levels: married single
```

```
##
y2 <- y;
levels(y2) <- c("single", "married", "widowed");
sort(y2);
```

```
## [1] single  single  married married
## Levels: single married widowed
```

sort data in a meaningful way ...

```
## Month
x1 <- c("Dec", "Apr", "Jan", "Mar");
sort(x1);
```

```
## [1] "Apr" "Dec" "Jan" "Mar"
```

```
month_levels <- c(
  "Jan", "Feb", "Mar", "Apr", "May", "Jun",
  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
)

y1 <- factor(x1, levels = month_levels)
sort(y1);
```

```
## [1] Jan Mar Apr Dec
## Levels: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
```

以出现的顺序为 factor

```
## Sometimes you'd prefer that the order of the levels match the order of the first appearance
f1 <- factor(x1, levels = unique(x1));
f1;
```

```
## [1] Dec Apr Jan Mar
## Levels: Dec Apr Jan Mar
```

```
library(forcats); ## just to make sure the codes will run smoothly ...
## you can also use fct_inorder in the forcats package ...
f2 <- x1 %>% factor() %>% fct_inorder()
f2
```

```
## [1] Dec Apr Jan Mar
## Levels: Dec Apr Jan Mar
```

use factor to clean data

```
## 假设我有一组性别数据，其写法非常不规整;
gender <- c("f", "m ", "male ","male", "female", "FEMALE", "Male", "f", "m");

## 要求: 都改为 Female, Male
gender <- as_factor( gender );
fct_count( gender );


## # A tibble: 8 x 2
##    f            n
##    <fct>     <int>
## 1 "f"           2
## 2 "m "          1
## 3 "male "       1
## 4 "male"        1
## 5 "female"      1
## 6 "FEMALE"      1
## 7 "Male"        1
## 8 "m"           1

gender <- fct_collapse(
  gender,
  Female = c("f", "female", "FEMALE"),
  Male   = c("m ", "m", "male ", "male", "Male")
)
fct_count(gender)
```

or use `fct_relabel`

```
gender <- c("f", "m ", "male ","male", "female", "FEMALE", "Male", "f", "m")
gender <- as_factor(gender)
gender <- fct_relabel(gender, ~ ifelse(tolower(substring(., 1, 1)) == "f", "Female", "Male"))

fct_count(gender)
```

```
## # A tibble: 2 x 2
##   f          n
##   <fct>  <int>
## 1 Female     4
## 2 Male       5
```

factor 在做图中的应用（真正精髓）

```
## 一项 mock 调查结果数据
( responses <- factor( c("Agree", "Agree", "Strongly Agree", "Disagree",
                          "Disagree", "Agree") ) );
```

```
## [1] Agree          Agree          Strongly Agree Disagree       Disagree
## [6] Agree
## Levels: Agree Disagree Strongly Agree
```

```
## -- plot the results --
library(ggplot2);
barplot <-
  ggplot( data = data.frame( res = responses ), aes( x = res )) +
  geom_bar();
```

factor 在做图中的应用，cont.

默认情况下，factor 按字母表排序：Agree -> Disagree -> Strong Agree 。
ggplot2 也会按 factor 的排序作图

调整 factor 以调整画图顺序

```
res <- data.frame( res=responses );
## -- 按照同意程度从强->弱 排序
res$res <- factor( res$res, levels = c( "Strongly Agree", "Agree",  "Disagree" ) );
str(res);
```

```
## 'data.frame':    6 obs. of  1 variable:
##  $ res: Factor w/ 3 levels "Strongly Agree",..: 2 2 1 3 3 2
```

```
plot2 <-
  ggplot( data = res, aes( x = res )) +
  geom_bar() +
  xlab( "Response" );
```

调整 factor 以调整画图顺序，cont.
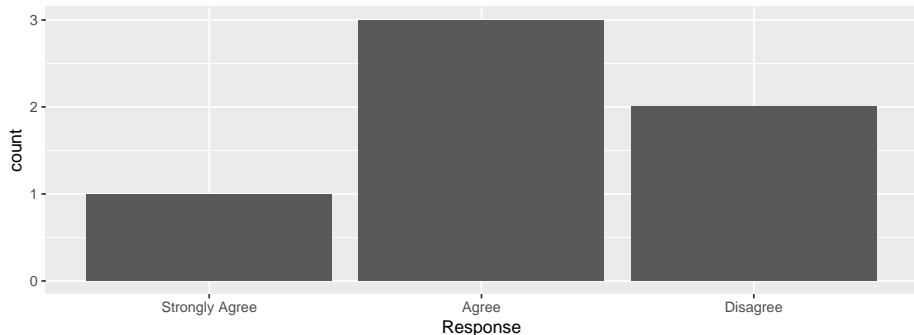
** 练习 ** 按意程度从弱-> 强排序并作图!!

ordered factor

通过 ordered 参数，让用户知道 factors 是经过精心排序的
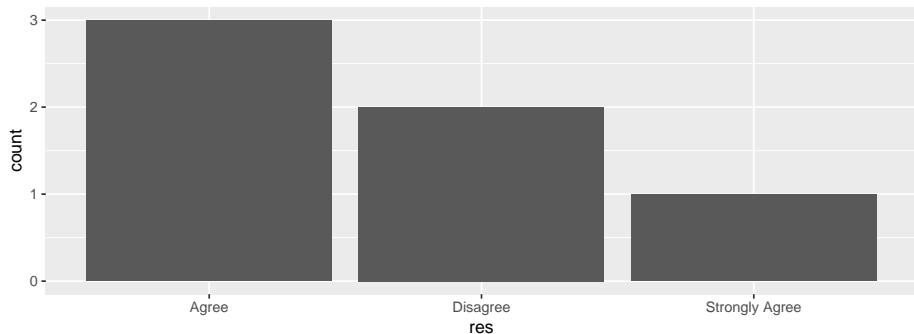
```
( responses <- factor( c("Agree", "Agree", "Strongly Agree", "Disagree", "Disagree", "Agree"),
                       ordered = T ) );
```

```
## [1] Agree          Agree          Strongly Agree Disagree       Disagree
## [6] Agree
## Levels: Agree < Disagree < Strongly Agree
```

```
is.ordered( responses );
```

```
## [1] TRUE
```

通过 factor 改变值

使用 dplyr 包的 recode() 函数改变 value

```r
( x <- factor( c( "alpha", "beta", "gamma", "theta", "beta", "alpha" ) ) );
```

```
## [1] alpha beta  gamma theta beta  alpha
## Levels: alpha beta gamma theta
```

```
## --
library( dplyr );
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
x <- recode( x, "alpha" = "one", "beta" = "two" );
str(x);
```

```
##  Factor w/ 4 levels "one","two","gamma",..: 1 2 3 4 2 1
```

去除不用的 levels

？什么时候会用到：

```
mouse.genes <- read.delim( file = "data/talk04/mouse_genes_biomart_sep2018.txt",
                           sep = "\t", header = T, stringsAsFactors = T );

str(mouse.genes);
```

```
## 'data.frame':    138532 obs. of  6 variables:
##  $ Gene.stable.ID                       : Factor w/ 55029 levels "ENSMUSG00000000001",.
##  $ Transcript.stable.ID                 : Factor w/ 138532 levels "ENSMUST00000000001",
##  $ Protein.stable.ID                    : Factor w/ 65897 levels "","ENSMUSP00000000001
##  $ Transcript.length..including.UTRs.and.CDS.: int  67 67 1144 69 519 1824 71 59 67 1378 ...
##  $ Transcript.type                      : Factor w/ 48 levels "3prime_overlapping_ncRNA
##  $ Chromosome.scaffold.name             : Factor w/ 117 levels "1","10","11",..: 115 11
```

去除不用的 **levels, cont.**

```
mouse.chr_10_12 <- subset( mouse.genes,  Chromosome.scaffold.name %in% c( "10", "11", "12" ) );
## plot length distribution --

boxplot( Transcript.length..including.UTRs.and.CDS. ~ Chromosome.scaffold.name,
         data = mouse.chr_10_12, las = 2 );
```
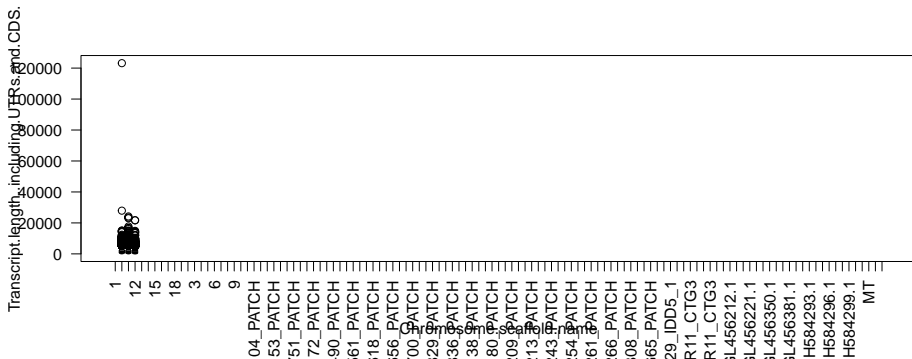

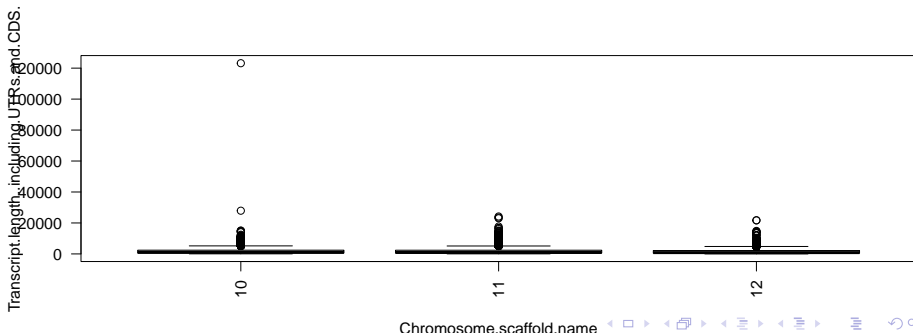subset() 无法去除不用的 factors ...

去除不用的 **levels, cont.**

```
mouse.chr_10_12$Chromosome.scaffold.name <-
  droplevels( mouse.chr_10_12$Chromosome.scaffold.name );

levels( mouse.chr_10_12$Chromosome.scaffold.name );
```

```
## [1] "10" "11" "12"
```

```
## 再次 plot ...
boxplot( Transcript.length..including.UTRs.and.CDS. ~ Chromosome.scaffold.name,
         data = mouse.chr_10_12, las = 2 );
```

也可以使用 tibble，完全不用担心 factor 的问题 ...

```
library( readr );
mouse.tibble <- read_delim( file = "data/talk04/mouse_genes_biomart_sep2018.txt",
                            delim = "\t", quote = "" )


##
## -- Column specification --------------------------------------------------
## cols(
##   `Gene stable ID` = col_character(),
##   `Transcript stable ID` = col_character(),
##   `Protein stable ID` = col_character(),
##   `Transcript length (including UTRs and CDS)` = col_double(),
##   `Transcript type` = col_character(),
##   `Chromosome/scaffold name` = col_character()
## )

mouse.tibble.chr10_12 <-
  mouse.tibble %>% filter( `Chromosome/scaffold name` %in%  c( "10", "11", "12" ) );

plot3 <-
  ggplot( data = mouse.tibble.chr10_12,
          aes( x = `Chromosome/scaffold name`,
               y = `Transcript length (including UTRs and CDS)` ) ) +
  geom_boxplot() +
  coord_flip() +
  ylim( 0, 2500 ) ;
```

用 tibble 解决 factor 的问题，cont.

Warning: Removed 4770 rows containing non-finite values (st
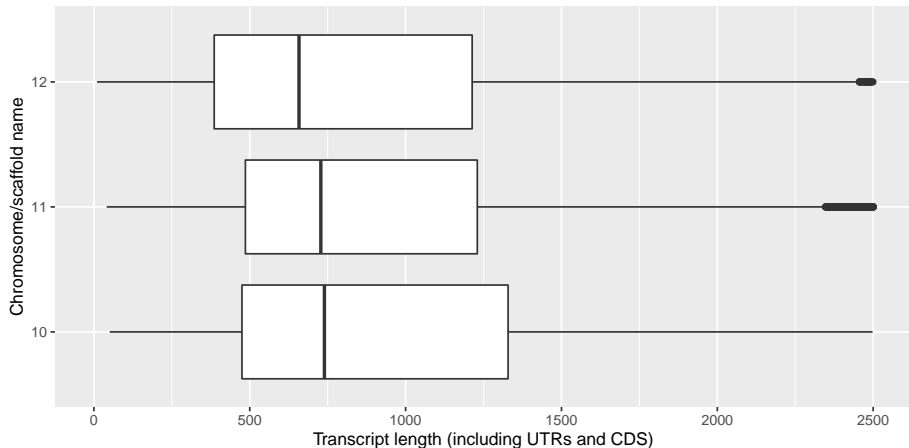
按基因长度中值从大 -> 小排序

```
plot4 <-
  ggplot( data = mouse.tibble.chr10_12,
          aes( x = reorder( `Chromosome/scaffold name`,
                            `Transcript length (including UTRs and CDS)`,
                            median ),
               y = `Transcript length (including UTRs and CDS)` ) ) +
  geom_boxplot() +
  coord_flip() +
  ylim( 0, 2500 ) ;
```

reorder(vector_with_factor, numeric_value , FUN = mean) 的用法

按基因长度中值从大 -> 小排序，cont.

Warning: Removed 4770 rows containing non-finite values (st
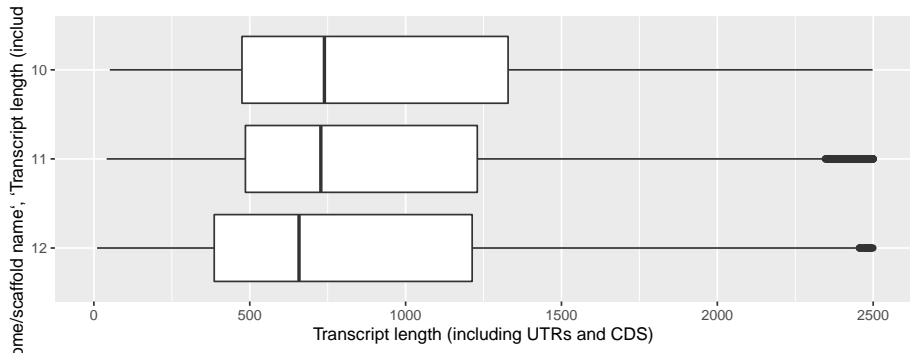
** 注意 ** reorder(`Chromosome/scaffold name`, - `Transcript
length (including UTRs and CDS)`, median) 的作用

按基因长度中值从大 -> 小排序，cont.

** 问题 **

1. 如果要按小 -> 大的顺序排序呢？("reorder(Chromosome/scaffold name，-Transcript length (including UTRs and CDS)', median))

2. reorder 的作用是什么 ?? 只在 ggplot2 里有用吗 ??

use forcats::fct_reorder to reorder factors

```
ggplot( data = mouse.tibble.chr10_12,
        aes( x = fct_reorder( `Chromosome/scaffold name`,
                              `Transcript length (including UTRs and CDS)`,
                        median ),
             y = `Transcript length (including UTRs and CDS)` ) ) +
geom_boxplot() +
coord_flip() +
ylim( 0, 2500 ) ;
```
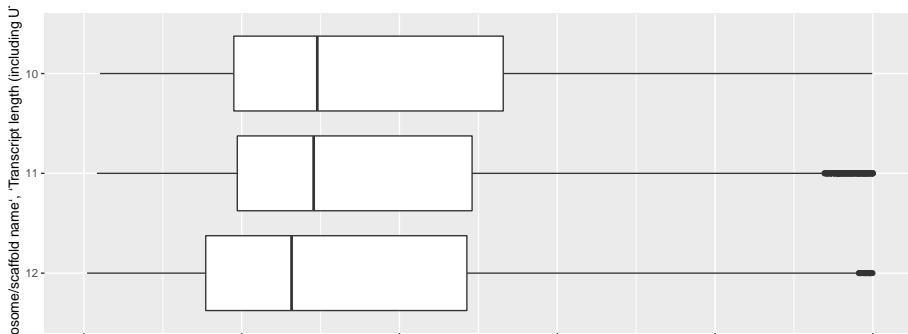
Warning: Removed 4770 rows containing non-finite values (stat_boxplot).

play around with `gss_cat`: General Social Survey

先看一下数据：

```
## # A tibble: 15 x 9
##     year marital       age race  rincome     partyid      relig     denom     tvhours
##    <int> <fct>       <int> <fct> <fct>       <fct>        <fct>      <fct>       <int>
##  1  2000 Never ma~      26 White $8000 to ~  Ind,near r~  Protesta~  Souther~       12
##  2  2000 Divorced       48 White $8000 to ~  Not str re~  Protesta~  Baptist~       NA
##  3  2000 Widowed        67 White Not appli~  Independent  Protesta~  No deno~        2
##  4  2000 Never ma~      39 White Not appli~  Ind,near r~  Orthodox~  Not app~        4
##  5  2000 Divorced       25 White Not appli~  Not str de~  None       Not app~        1
##  6  2000 Married        25 White $20000 - ~  Strong dem~  Protesta~  Souther~       NA
##  7  2000 Never ma~      36 White $25000 or~  Not str re~  Christian  Not app~        3
##  8  2000 Divorced       44 White $7000 to ~  Ind,near d~  Protesta~  Luthera~       NA
##  9  2000 Married        44 White $25000 or~  Not str de~  Protesta~  Other           0
## 10  2000 Married        47 White $25000 or~  Strong rep~  Protesta~  Souther~        3
## 11  2000 Married        53 White $25000 or~  Not str de~  Protesta~  Other           2
## 12  2000 Married        52 White $25000 or~  None         Not app~   NA
## 13  2000 Married        52 White $25000 or~  Strong dem~  Protesta~  Souther~        1
## 14  2000 Married        51 White $25000 or~  Strong rep~  Protesta~  United ~       NA
## 15  2000 Divorced       52 White $25000 or~  Ind,near d~  None       Not app~        1
```
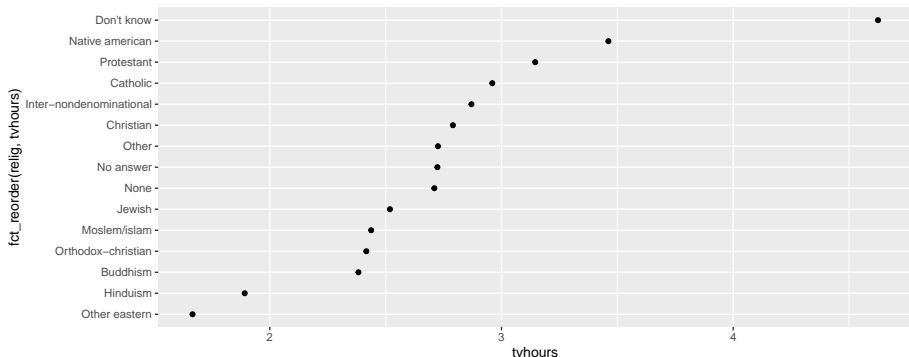
tv hours vs. religion

```r
relig_summary <- gss_cat %>% group_by(relig) %>%
  summarise(
    age = mean(age, na.rm = TRUE),
    tvhours = mean(tvhours, na.rm = TRUE),
    n = n()
  )

ggplot(relig_summary, aes(tvhours, fct_reorder(relig, tvhours))) + geom_point()
```

section 4: 练习 & 作业

练习 & 作业

- Exercises and homework 目录下 talk04-homework.Rmd 文件；
- 完成时间：见钉群的要求

小结

今次提要

- IO, project management, working environment management
- factor : R 另一个超级重要且难以上手的概念
    - 定义
    - 操作
    - 使用
- 基础和进阶绘图（配合 factor 讲解）

下次预告

- data-wrangler: `dplyr`

important

- all codes are available at Github:
  https://github.com/evolgeniusteam/R-for-bioinformatics