

R for bioinformatics, data wrangler practices

HUST Bioinformatics course series

Wei-Hua Chen (CC BY-NC 4.0)

08 October, 2024

section 1: TOC

前情提要

pipe

- %>%
- %\$%
- %T>%
- %<>%

tidyverse

- pivot_longer()
- pivot_wider()
- unite()
- separate()

dplyr

- select()
- filter()

本次提要

- ① 3 个生信任务的 R 解决方案
- ② factors 的更多应用 (forcats)

section 2: contents

生信任务 1 : network analysis and visualisation (dplyr & some plot packages)

protein-protein interaction data

why PPI is important?

- ① most of the time, protein functions together with other proteins (interactions)
- ② interacting partners tend to have similar functions (guilty by association, can be used in gene annotation)

STRING database

- ① contains >2 billion interactions for 24.6 million proteins in 5090 organisms (as of May 2021; ver 11.0b)
- ② contains:
 - physical interaction
 - genetic interactions
 - gene co-occurrence (text-mining)
 - transfers through orthologous relationships

STRING is one of the most cited resources

STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets

3973

2019

D Szklarczyk, AL Gable, D Lyon, A Junge, S Wyder, J Huerta-Cepas, ...
Nucleic acids research 47 (D1), D607-D613

screenshot took in May 2021

more tools at Peer Bork's group:

https://scholar.google.com/citations?hl=en&user=M6Etr6oAAAAJ&view_op=list_works&sortby=pubdate

a typical STRING plot

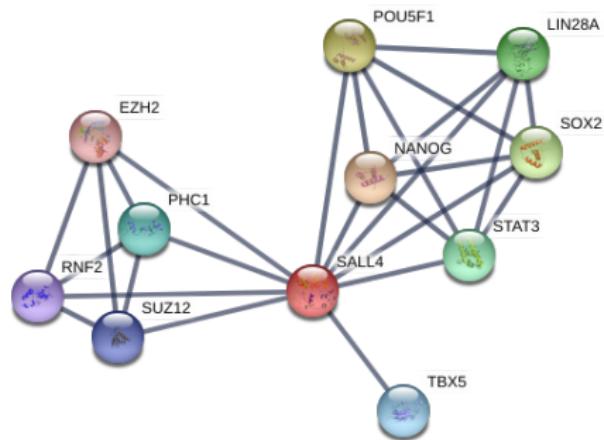


Figure 1: interacting partners of SALL4

note to previous plot

note this plot was generated using the following parameters:

- use SALL4 in human as query
- show only the top ten connectivity partners
- only connectivity score ≥ 900 (or 0.9) were shown

tasks of task 1

- ① get human PPI data
- ② limit the interactions to those with scores ≥ 900 (or 0.9)
- ③ find SALL4 and its top ten interaction partners
- ④ visualize the PPI network
- ⑤ calculate connectivities of sall4 and its top ten partners in the dataset

问题

生物学意义何在 ??

download human ppi data from STRING

go to : <https://string-db.org/cgi/download.pl>

The screenshot shows the STRING download interface. At the top, a red arrow points to a link labeled "to find PPI data by species name". Below it, a dropdown menu is set to "Homo sapiens" and has a red arrow pointing to it. A blue "UPDATE" button is to the right. The main section is titled "INTERACTION DATA" and contains a table with five rows. A red arrow points to the "File" column header. The table columns are "File", "Description", and "Access".

File	Description	Access
9606.protein.links.v11.0.txt.gz (71.2 Mb)	protein network data (scored links between proteins)	
9606.protein.links.detailed.v11.0.txt.gz (110.1 Mb)	protein network data (incl. subscores per channel)	
9606.protein.links.full.v11.0.txt.gz (127.6 Mb)	protein network data (incl. distinction: direct vs. interologs)	
9606.protein.actions.v11.0.txt.gz (14.4 Mb)	interaction types for protein links	

Figure 2: Download human PPI data from STRING

load and load the human PPI data

```
library(tidyverse);

## read_csv 也能处理压缩文件!!!
ppi <- read_delim( file = "data/talk06/ppi900.txt.gz", col_names = T,
                   delim = "\t", quote = "" );

## 查看一下数据 --
ppi %>% filter( gene1 == "SALL4" ) %>% do( head(., n = 10) );

## # A tibble: 10 x 3
##   gene1 gene2   score
##   <chr> <chr>   <dbl>
## 1 SALL4 POLR2E     900
## 2 SALL4 POLR2C     900
## 3 SALL4 POLR2I     900
## 4 SALL4 NANOG     992
## 5 SALL4 SALL1     923
## 6 SALL4 ZSCAN10    912
## 7 SALL4 LIN28A     957
## 8 SALL4 POU5F1     986
## 9 SALL4 SMAD2     906
## 10 SALL4 EPAS1     900
```

start to process the data

```
## get top 10 interacting partners of SALL4 by interaction score ...
toppart <- ppi %>% filter( gene1 == "SALL4" ) %>%
  arrange( desc( score ) ) %>% slice( 1:10 );

## get the interaction network consisting the top genes --
genes <- unique( c( "SALL4", toppart$gene2 ) );
netdata <- ppi %>% filter( gene1 %in% genes & gene2 %in% genes );
nrow(netdata);

## [1] 80
```

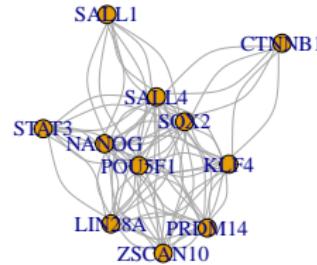
load the igraph package

```
## -- to make sure the installation will only run once ...
if (!require("igraph")){
  chooseCRANmirror();
  install.packages("igraph");
}

library( igraph );
```

visualise the network ...

```
netnet <- graph_from_data_frame( netdata, directed = FALSE );  
plot(netnet);
```



图的问题

- very ugly
- two lines (redundancy) between every two nodes

redundancy among data

```
netdata %>% filter( gene1 %in% c("SALL4", "NANOG") & gene2 %in% c("SALL4", "NANOG") );  
  
## # A tibble: 2 x 3  
##   gene1 gene2 score  
##   <chr>  <chr> <dbl>  
## 1 NANOG  SALL4    992  
## 2 SALL4  NANOG    992
```

how to remove redundancy?

```
## create a new column, sort the two gene names, and concatenate them ...
testdata <-  
  netdata %>% filter( gene1 %in% c("SALL4", "NANOG") & gene2 %in% c("SALL4", "NANOG") ) %>%  
  mutate( group =  
    if_else( gene1 > gene2,  
            str_c( gene1, gene2, sep = "-"),  
            str_c( gene2, gene1, sep = "-" ) ) );  
  
testdata;  
  
## # A tibble: 2 x 4  
##   gene1 gene2 score group  
##   <chr> <chr> <dbl> <chr>  
## 1 NANOG SALL4     992 SALL4-NANOG  
## 2 SALL4 NANOG     992 SALL4-NANOG
```

Note str_c is from the stringr package!!

remove redundancy!

```
testdata %>% group_by( group ) %>% slice( 1 );
```

```
## # A tibble: 1 x 4
## # Groups:   group [1]
##   gene1 gene2 score group
##   <chr>  <chr>  <dbl> <chr>
## 1 NANOG SALL4    992 SALL4-NANOG
```

remove redundancy, cont.

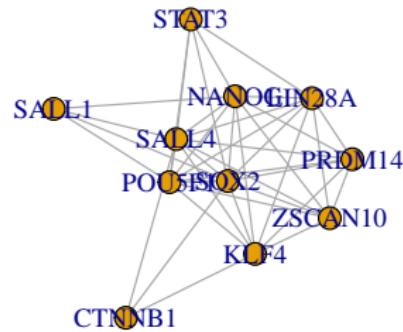
```
netdata.nr <-
  netdata %>%
  mutate( group =
    if_else( gene1 > gene2,
            str_c( gene1, gene2, sep = "-"),
            str_c( gene2, gene1, sep = "-" ) ) ) %>%
  group_by( group ) %>% slice( 1 );

nrow(netdata.nr);

## [1] 40
```

plot the non-redundant data

```
netnet.nr <- graph_from_data_frame( netdata.nr, directed = FALSE );  
plot(netnet.nr);
```



redundant 的数据可以用来计算 degree

```
net.stats <-  
  netdata %>% group_by( gene1 ) %>% summarise( degree = n() ) %>%  
  arrange( desc (degree) );  
net.stats;
```

```
## # A tibble: 11 x 2  
##   gene1     degree  
##   <chr>     <int>  
## 1 SALL4      10  
## 2 SOX2       10  
## 3 NANOG       9  
## 4 POU5F1      9  
## 5 KLF4        8  
## 6 LIN28A      8  
## 7 PRDM14      7  
## 8 ZSCAN10      7  
## 9 STAT3        5  
## 10 SALL1       4  
## 11 CTNNB1       3
```

继续美化 network

```

## node 大小由 degree 决定
## 查看网络中基因名存储的顺序
V(netnet.nr)$name;

## [1] "KLF4"      "LIN28A"     "POU5F1"     "PRDM14"    "SALL1"     "CTNNB1"     "NANOG"
## [8] "SOX2"       "STAT3"      "ZSCAN10"    "SALL4"

## 获取它们相对应的 degree ...
net.stats[match( V(netnet.nr)$name , net.stats$gene1 ), ];

```

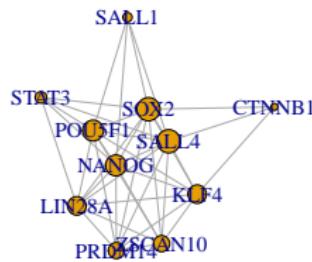
```

## # A tibble: 11 x 2
##   gene1   degree
##   <chr>     <int>
## 1 KLF4        8
## 2 LIN28A      8
## 3 POU5F1      9
## 4 PRDM14      7
## 5 SALL1       4
## 6 CTNNB1      3
## 7 NANOG       9
## 8 SOX2        10
## 9 STAT3       5
## 10 ZSCAN10     7
## 11 SALL4      10

```

继续美化 network, cont.

```
vertex_attr(netnet.nr, "size") <-  
  net.stats$degree[match( V(netnet.nr)$name , net.stats$gene1 ) ] * 2;  
plot( netnet.nr );
```



更多美化

详见：

- ① igraph introduction
- ② a comprehensive network visualization tutorial in R

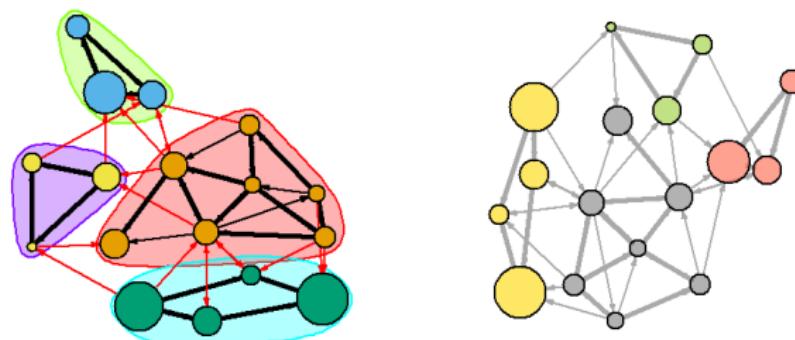


Figure 3: Example final outcomes

other network visualisation packages

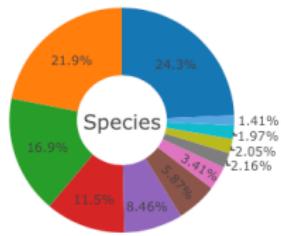
- ① ggnet
- ② interactive networkD3 R package
- ③ plotly.js
- ④ D3

生信任务 2：宏基因基因数据展示的小应用 (forcats)

Relative abundances

Species

Relative abundance



Relative abundance

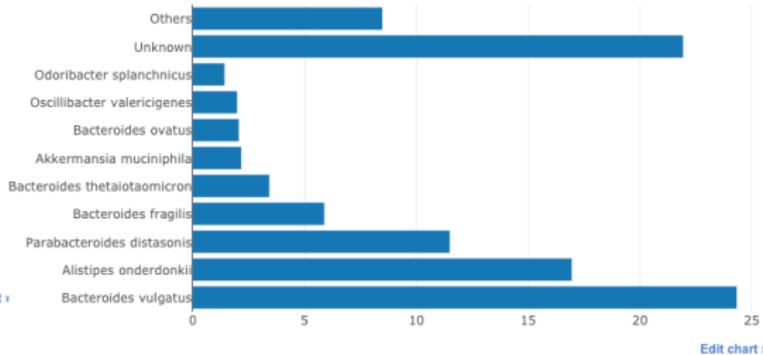


Figure 4: 肠道物种丰度示意图

Find more at the GMrepo database.

什么是宏基因组？

Metagenomics is the study of genetic material recovered directly from environmental samples.

research contents

- mostly prokaryotes
- unicellular eukaryotes
- viruses

techniques

- 16S (universially conserved gene in prokaryotes)
- whole genome sequencing (WGS or metagenomics)

what can metagenomics do?

- identify new species
- reveal species kinds and abundances
- relate species changes to human health and disease

biomes



Figure 5: Biomes

Screenshot taken from the EBI MGnify database on Aug 4, 2021;

environmental microbiome

- soil
- ocean



Figure 6: The tara oceans expedition

host associated microbiomes

- human body sites
- animals & plants



Figure 7: NIH human microbiome project

why human gut microbiota is important?

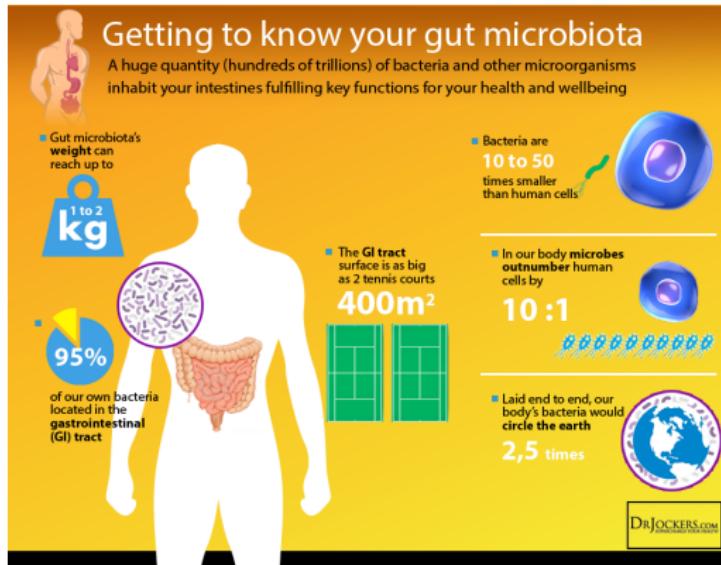


Figure 8: human gut microbiota

tasks of human gut microbiota analysis

- identify new species
- find good, bad and commensal microbes
- link microbial variations to human health
- mechanisms
- modulation, intervention and regulation

typical human gut microbiome data

Species abundances

```
abu <-  
  read_delim(  
    file = "data/talk06/relative_abundance_for_RUN_ERR1072629_taxonlevel_species.txt",  
    delim = "\t", quote = "", comment = "#");  
  
## Rows: 122 Columns: 3  
## -- Column specification -----  
## Delimiter: "\t"  
## chr (1): scientific_name  
## dbl (2): ncbi_taxon_id, relative_abundance  
##  
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.  
  
nrow(abu);  
  
## [1] 122
```

Species abundances, cont.

```
abu %>% arrange( desc( relative_abundance ) ) %>% do( head(., n = 10) );
```

```
## # A tibble: 10 x 3
##   ncbi_taxon_id relative_abundance scientific_name
##       <dbl>            <dbl> <chr>
## 1 821             24.3  Bacteroides vulgatus
## 2 -1              21.9  Unknown
## 3 328813          16.9  Alistipes onderdonkii
## 4 823              11.5  Parabacteroides distasonis
## 5 817              5.87   Bacteroides fragilis
## 6 818              3.41   Bacteroides thetaiotaomicron
## 7 239935           2.16   Akkermansia muciniphila
## 8 28116            2.05   Bacteroides ovatus
## 9 351091           1.97   Oscillibacter valericigenes
## 10 28118           1.41   Odoribacter splanchnicus
```

相对丰度作图要求

- ① 按丰度从高到低排序
- ② 只取前 10 个 species (保留 10 行)
- ③ 将后面的丰度累加在一起，汇总为 "Others" 分类

数据处理

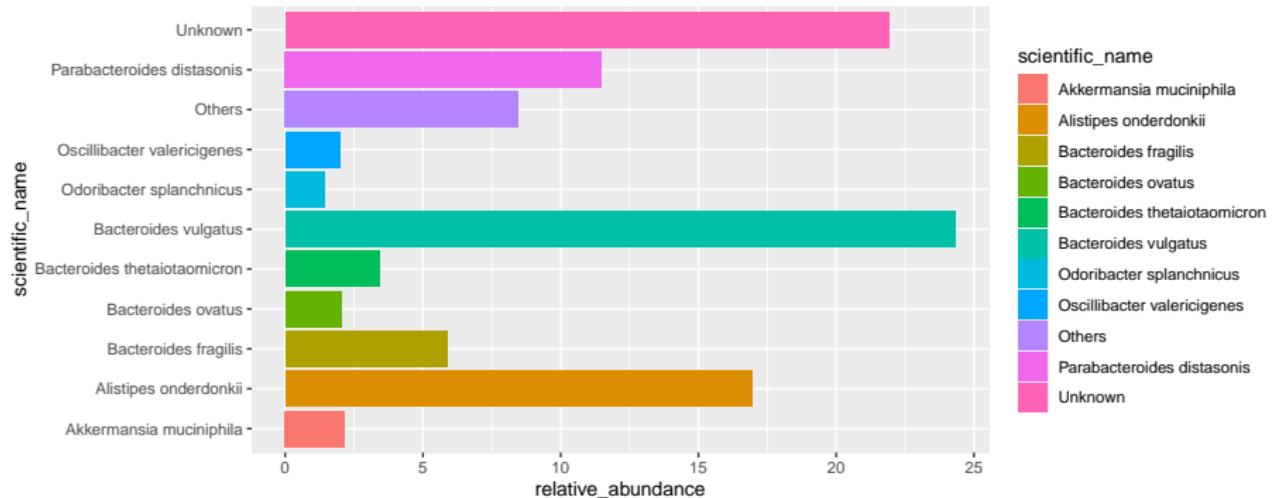
```
library( tidytidbits );
abu.dat <-
  abu %>% arrange( desc( relative_abundance ) ) %>%
  lump_rows( scientific_name, relative_abundance, n = 10, other_level = "Others" );

head(abu.dat, n = 11);

## # A tibble: 11 x 3
##   ncbi_taxon_id relative_abundance scientific_name
##   <dbl>           <dbl> <chr>
## 1 821            24.3  Bacteroides vulgatus
## 2 -1              21.9  Unknown
## 3 328813          16.9  Alistipes onderdonkii
## 4 823             11.5  Parabacteroides distasonis
## 5 817             5.87   Bacteroides fragilis
## 6 818             3.41   Bacteroides thetaiotaomicron
## 7 239935          2.16   Akkermansia muciniphila
## 8 28116            2.05   Bacteroides ovatus
## 9 351091          1.97   Oscillibacter valericigenes
## 10 28118           1.41   Odoribacter splanchnicus
## 11 31199992        8.46   Others
```

尝试作图

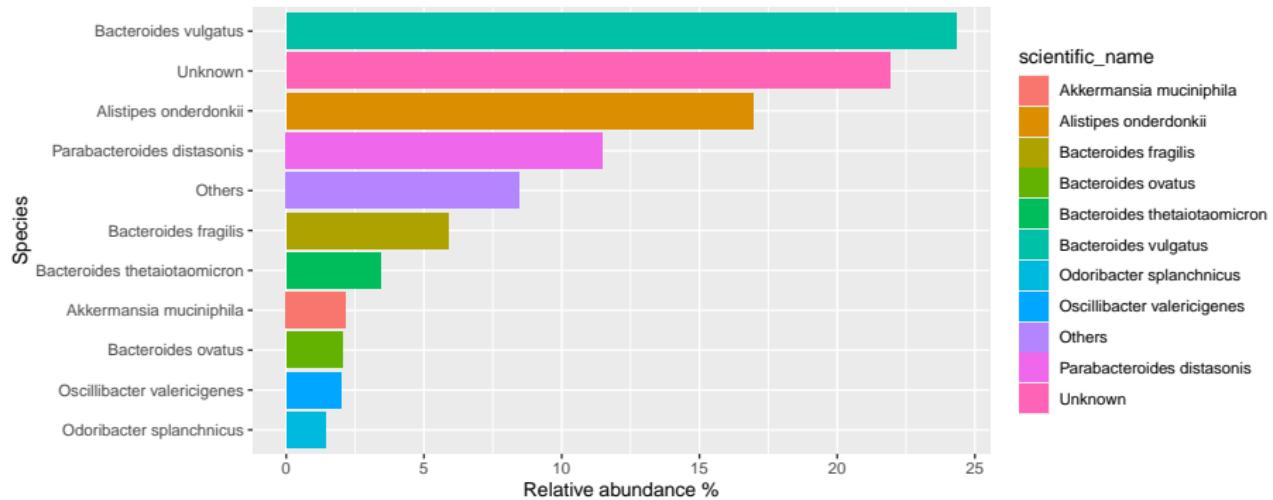
```
ggplot(abu.dat, aes(x = scientific_name, y = relative_abundance, fill = scientific_name ) ) +
  geom_bar( stat = "identity" ) +
  coord_flip()
```



调整排列顺序

用 `forcat` 包的 `fct_reorder` 函数；注意它 3 个参数的意义!!!

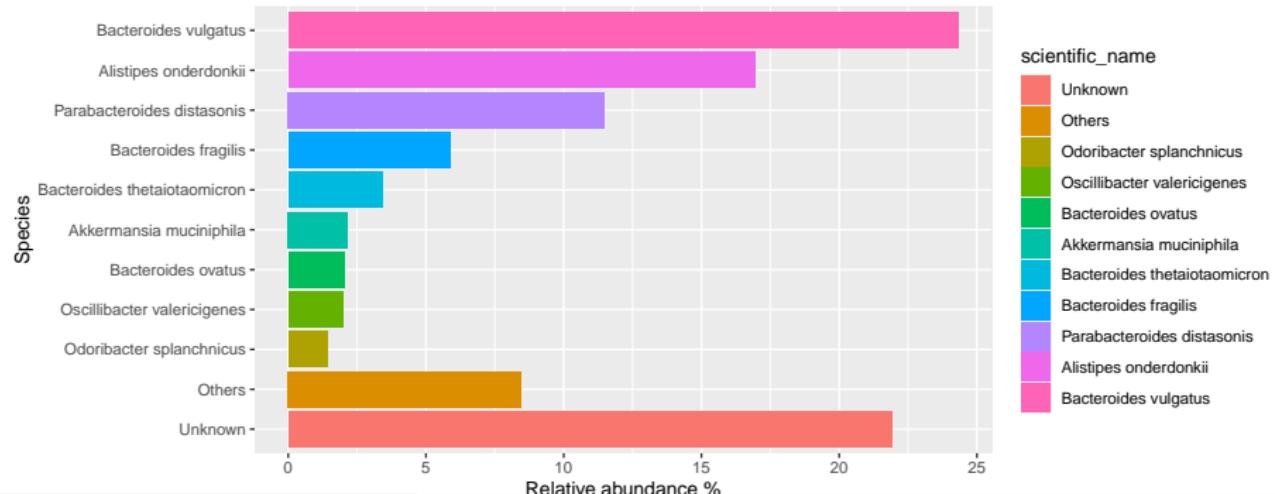
```
ggplot(abu.dat, aes(x = fct_reorder( scientific_name, relative_abundance, .desc = F),
                     y = relative_abundance, fill = scientific_name ) ) +
  geom_bar( stat = "identity" ) +
  coord_flip() + xlab("Species") + ylab( "Relative abundance %" )
```



把 Others 和 Unknown 放在最后

注意 `fct_reorder` 的用法：

```
abu.dat$scientific_name <-
  fct_relevel( fct_reorder( abu.dat$scientific_name, abu.dat$relative_abundance, .desc = F ),
    "Unknown", "Others" );
ggplot(abu.dat, aes(x = scientific_name,
                     y = relative_abundance, fill = scientific_name ) ) +
  geom_bar( stat = "identity" ) +
  coord_flip() + xlab("Species") + ylab( "Relative abundance %" )
```



更多forcats的应用 ...

see here: <https://cran.r-project.org/web/packages/forcats/vignettes/forcats.html>

更多应用将会在作图（ggplot2）时讲到。

生信任务 3：整合基因表达、甲基化、突变数据 (dplyr::join)

在生信分析中，常需要将多个来源的数据整合在一个表格中，以方便后续分析。

```

meth <- read_delim( file = "data/talk06/methylation_data.txt.gz",
                     delim = "\t", quote = "", col_names = T);

## # Rows: 52232 Columns: 3
## -- Column specification -----
## Delimiter: "\t"
## chr (2): gene, site
## dbl (1): methylation_score
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

head(meth, n = 3);

## # A tibble: 3 x 3
##   gene    site  methylation_score
##   <chr>   <chr>        <dbl>
## 1 A1BG   1stExon      0.799
## 2 A1BG   5UTR         0.799
## 3 A1BG   Body          0.811

```

表达数据

```
expr <- read_delim( file = "data/talk06/expression_data.txt.gz",
                     delim = "\t", quote = "", col_names = T );

## Rows: 46150 Columns: 2
## -- Column specification -----
## Delimiter: "\t"
## chr (1): gene
## dbl (1): rkpm
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

head( expr, n = 5);

## # A tibble: 5 x 2
##   gene      rkpm
##   <chr>    <dbl>
## 1 5S_RRNA     0
## 2 7SK        0
## 3 A1BG       1
## 4 A1BGAS1    4
## 5 A1CF       0
```

整合后的结果应该是什么 ??

gene	TSS200	TSS1500	UTR	body	expression
gene1	0.1	0.2	NA	0.8	100
gene2	0.12	0.32	NA	0.9	18

Figure 9: 整合后的结果示意

第一种方法，使用 pivot_wider

思路：先合并，再 pivot_wider

① 准备数据

```
## 用 `bind_rows` 合并两个 tibble 时，列名需要一致
meth2 <- meth %>% select( gene, group=site, value=methylation_score );
head(meth2, n=2);
```

```
## # A tibble: 2 x 3
##   gene   group   value
##   <chr>  <chr>   <dbl>
## 1 A1BG   1stExon 0.799
## 2 A1BG   5UTR     0.799
```

```
expr2 <- expr %>% mutate( group = "rkpm" ) %>%
  select( gene, group, value=rkpm ) %>% group_by( gene ) %>% slice( 1 );
head(expr2, n=2);
```

```
## # A tibble: 2 x 3
## # Groups:   gene [2]
##   gene   group value
##   <chr>  <chr> <dbl>
## 1 5S_RRNA rkpm    0
## 2 7SK      rkpm    0
```

合并 & pivot_wider

② 合并

③ pivot_wider

```
comb <- bind_rows( meth2, expr2 );

comb.wide <- comb %>% pivot_wider( names_from = "group", values_from = "value" );

head(comb.wide);

## # A tibble: 6 x 8
##   gene   `1stExon` `5UTR`     Body TSS1500 TSS200 `3UTR`   rkpm
##   <chr>    <dbl>  <dbl>    <dbl>  <dbl>  <dbl>  <dbl>    <dbl>
## 1 A1BG      0.799  0.799  0.811   0.852  0.850 NA       1
## 2 A1CF      0.0946  0.118  0.00798  0.110  NA     NA       0
## 3 A2M       0.899  NA     0.716   0.654  NA     0.776   1
## 4 A2ML1     0.859  0.859  0.678   0.441  NA     0.469   0
## 5 A4GALT    NA     0.844  0.821   0.890  0.513  0.814   0
## 6 A4GNT      0.680  0.680  0.538   0.402  NA     NA       0
```

方法二：使用 join ...

首先对 methylation 数据进行处理

```
meth3 <-  
  meth %>% pivot_wider( names_from = "site", values_from = "methylation_score" );  
  
head(meth3);
```

```
## # A tibble: 6 x 7  
##   gene   `1stExon` `5UTR`    Body TSS1500 TSS200 `3UTR`  
##   <chr>     <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>  
## 1 A1BG      0.799   0.799   0.811    0.852   0.850    NA  
## 2 A1CF      0.0946  0.118   0.00798   0.110   NA        NA  
## 3 A2M       0.899   NA      0.716    0.654   NA        0.776  
## 4 A2ML1     0.859   0.859   0.678    0.441   NA        0.469  
## 5 A4GALT    NA      0.844   0.821    0.890   0.513    0.814  
## 6 A4GNT     0.680   0.680   0.538    0.402   NA        NA
```

dplyr::join

```
comb2 <- left_join( meth3, expr2, by = "gene" ) %>% select( -group );  
head(comb2);
```

```
## # A tibble: 6 x 8  
##   gene   `1stExon` `5UTR`     Body TSS1500 TSS200 `3UTR` value  
##   <chr>     <dbl>    <dbl>     <dbl>    <dbl>    <dbl>    <dbl>  
## 1 A1BG      0.799    0.799    0.811     0.852    0.850     NA      1  
## 2 A1CF      0.0946   0.118    0.00798    0.110    NA        NA      0  
## 3 A2M       0.899    NA        0.716     0.654    NA        0.776    1  
## 4 A2ML1     0.859    0.859    0.678     0.441    NA        0.469    0  
## 5 A4GALT    NA        0.844    0.821     0.890    0.513     0.814    0  
## 6 A4GNT     0.680    0.680    0.538     0.402    NA        NA      0
```

注意 left_join 的语法

join 详解

- `left_join()`: return all rows from x, and all columns from x and y.
Rows in x with no match in y will have NA values in the new columns.
If there are multiple matches between x and y, all combinations of the matches are returned.
- `inner_join()`
- `right_join()`
- `full_join()`

更多请见: <https://dplyr.tidyverse.org/reference/join.html>

section 3: Exercise & home work

小结

本次提要

- ① 3 个生信任务的 R 解决方案
- ② factors 的更多应用 (forcats)

下次预告

- Strings and regular expression

important

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

练习 & 作业

- Exercises and homework 目录下 talk06-homework.Rmd 文件；
- 完成时间：见钉群的要求