NZSSN Courses: Introduction to R

Session 2 – Subsetting data

Statistical Consulting Centre

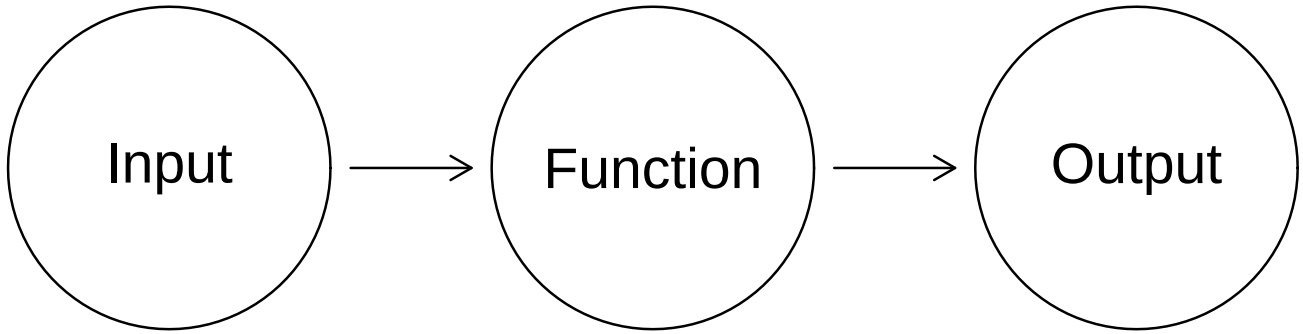
consulting@stat.auckland.ac.nz
The Department of Statistics
The University of Auckland

1 March, 2017

SCIENCE
DEPARTMENT OF STATISTICS

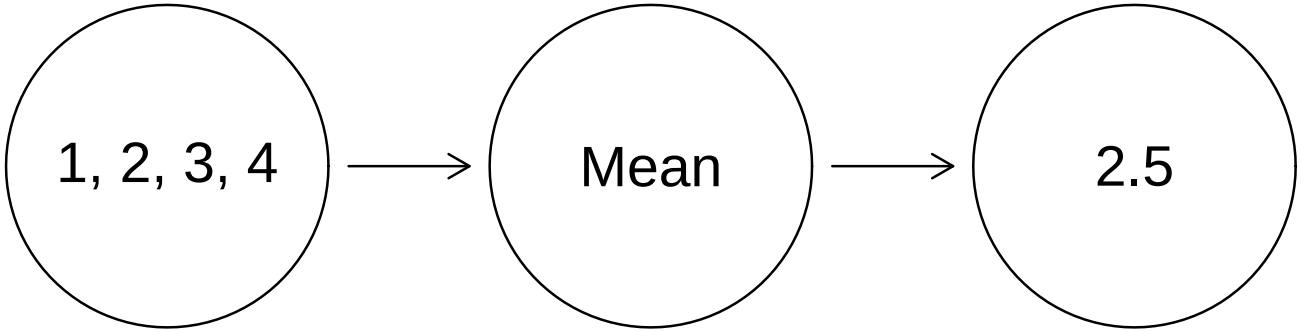
Functions

A function is a relationship between a set of inputs (arguments) and a set of outputs. E.g., the function is fed some information on which it operates, the results of which are the output.



Functions

We have seen many functions, e.g. `log`, `mean`, `table`, `with`, etc.



Working with functions

- Functions can be user-defined, i.e., you can write your own.
- Output is the last line of the function. You can use `return()` to specifiy the output.
- Here is a function calculates the standard error of the mean (SEM).

```
mystder <- function(x){
    mysd <- sd(x, na.rm = TRUE) # Calc std. deviation
    n <- length(x)              # Calc sample size
    mysd/sqrt(n)                # Definition of SEM
}
mystder(issp.df$Age)

[1] 0.5101842
```

- A set of user-defined functions can be bundled together into an R package.

Getting data into R

- Base R includes only functions which read data sets saved in simple file formats, e.g. csv, txt, tab delimited, etc.
- What if your data was saved in another format, e.g. STATA, SPSS, or SAS spreadsheets?
- The haven package for R contains functions that may help! https://cran.r-project.org/web/packages/haven/index.html

```
> library(haven)
> stata <- read_dta("data.dta")
> spss <- read_sav("data.sav")
> sas <- read_sas("data.sas7bdat")
> sasxport <- read_xpt("data.xpt")
```

However, it is always the easiest and safest to read data into R from a csv file.

Packages

- Currently, the CRAN package repository features 10,098 available packages (17 Feb. 2017). There are about 12,383 CRAN, BioConductor and Github packages in total.
- To install packages from the `R` GUI, click on `Packages` → `Install Package(s) ...` → `New Zealand (or whatever region you are located)` → `Package name`
- Or, you can type `install.packages("`*`package name`*`")`, e.g. `install.packages("haven")`.
- After the installation, use `library(`*`package name`*`)` to load it into `R`. Note: Installation is performed only once; however, it must be loaded (i.e. use the command `library(`*`package name`*`)`) in *every* `R` session.

Two-way frequency tables

```
income.gender.tab <- with(issp.df,
                          table(Income, Gender))
income.gender.tab
```

	Gender		
Income	Female	Male	NA, refused
$10000 or less	177	57	4
$10001-$15000	115	35	2
$15001-$20000	49	29	2
$20001-$25000	65	50	0
$25001-$30000	71	48	2
$30001-$40000	59	70	4
$40001-$50000	27	47	2
$50001-$70000	7	27	1
$70001-$100000	4	35	2
NAV; NAP No own income	33	20	3

Two-way frequency tables

We can convert the counts to percentages, i.e.

```
round(prop.table(income.gender.tab, 2) * 100, 1)
```

```
                          Gender
Income                     Female Male NA, refused
  $10000 or less             29.2 13.6         18.2
  $10001-$15000              18.9  8.4          9.1
  $15001-$20000               8.1  6.9          9.1
  $20001-$25000              10.7 12.0          0.0
  $25001-$30000              11.7 11.5          9.1
  $30001-$40000               9.7 16.7         18.2
  $40001-$50000               4.4 11.2          9.1
  $50001-$70000               1.2  6.5          4.5
  $70001-$100000              0.7  8.4          9.1
  NAV; NAP No own income      5.4  4.8         13.6
```

`which` Income is unavailable or none?

Let's use R's powerful subsetting capabilities to select those cases for which the value of Income is "NAV; NAP No own income".

```
index <- which(issp.df$Income
               == "NAV; NAP No own income")
index
```

```
 [1]    9   13   19   22   27   37   77  122  132
[10]  133  141  148  186  197  207  221  222  239
[19]  242  277  283  310  345  354  362  382  383
[28]  390  404  416  438  444  454  463  480  482
[37]  537  551  569  608  629  640  650  758  806
[46]  821  891  910  934  939  944  977  984  986
[55] 1027 1042
```

How many are there?

```
# Use length() to count the number of elements in "index".
length(index)

[1] 56
```

Who are they?

```
# Use square brackets to extract IDs corresponding
# to the cases numbers contained in "index"
issp.df$ID[index]

 [1] 1900097 1900139 1900181 1900205 1900241 1900331
 [7] 1900667 1901345 1901435 1901441 1901531 1901597
[13] 1901028 1901118 1901202 1901328 1901346 1901490
[19] 1901568 1900668 1900578 1900110 1900033 1900129
[25] 1900189 1900435 1900441 1900501 1900675 1900795
[31] 1900981 1901059 1901155 1901227 1901359 1901395
[37] 1900226 1900358 1900562 1900946 1901126 1901234
[43] 1901318 1900647 1901151 1901295 1900372 1900600
[49] 1900888 1900936 1900984 1901290 1901344 1901362
[55] 1900464 1901475
```

Subsetting

Square brackets [] are used to extract subsets of data.

```
# First element only
issp.df$Gender[1]

[1] "Female"

# All but the first element
issp.df$Gender[-1]

   [1] "Male"         "Female"       "Female"
   [4] "Female"       "Male"         "Female"
   [7] "Male"         "NA, refused"  "Male"
  [10] "Female"       "Female"       "Female"
  [13] "Female"       "Female"       "Male"
  [16] "Female"       "Female"       "Male"
  [19] "Female"       "Female"       "Male"
  [22] "Female"       "Female"       "Female"
```

Subsetting

```
#Elements 3 through 8
issp.df$Gender[3:8]

[1] "Female" "Female" "Female" "Male"   "Female" "Male"

#Elements 3 and 8
issp.df$Gender[c(3, 8)]

[1] "Female" "Male"
```

More on subsetting

Subsetting two-dimensional arrays, such as data frames, requires the use of *two indices.*

```
#First row or record
issp.df[1, ]

       ID       Q1    Q2                      Q3    Q4
1 1900073 disagree agree neither agree nor dis agree
                Q5           Q6           Q7           Q8
1 think themselves always wrong always wrong always wrong
  Gender Age  Marital.Status       Education
1 Female  56 marr,liv as mar Secondary compl
  Working.hours.per.week         Income       Ethnicity
1                NAV;NAP $10000 or less European/Pakeha
```

More on subsetting

```
#Second column or variable
issp.df[, 2]
```

```
   [1] "disagree"                "strongly disagree"
   [3] "disagree"                "cant choose, dk"
   [5] "disagree"                "disagree"
   [7] "strongly agree"          "neither agree nor dis"
   [9] "na, refused"             "disagree"
  [11] "agree"                   "neither agree nor dis"
  [13] "agree"                   "agree"
  [15] "agree"                   "agree"
  [17] "strongly agree"          "neither agree nor dis"
  [19] "strongly agree"          "neither agree nor dis"
  [21] "disagree"                "agree"
  [23] "disagree"                "agree"
  [25] "strongly agree"          "agree"
  [27] "disagree"                "neither agree nor dis"
```

More on subsetting

```
#Some rows and columns
issp.df[30:40, c(1, 10:12)]
```

```
         ID      Gender Age  Marital.Status
30 1900277        Male  50 marr,liv as mar
31 1900283 NA, refused  NA     NA, refused
32 1900295        Male  48 marr,liv as mar
33 1900307      Female  69          widowed
34 1900313      Female  NA marr,liv as mar
35 1900319        Male  46 marr,liv as mar
36 1900325      Female  46 marr,liv as mar
37 1900331        Male  44 marr,liv as mar
38 1900337      Female  36 marr,liv as mar
39 1900343        Male  32 marr,liv as mar
40 1900349      Female  90 marr,liv as mar
```

More on subsetting

```
#Rows by number, columns by name
issp.df[1:10, c("ID", "Ethnicity", "Education")]
```

```
         ID       Ethnicity                 Education
1   1900073 European/Pakeha          Secondary compl
2   1900013 European/Pakeha        University degree
3   1900025 European/Pakeha Incpl university,other
4   1900037 European/Pakeha Incpl university,other
5   1900043 European/Pakeha          Incpl secondary
6   1900061 European/Pakeha Incpl university,other
7   1900079 European/Pakeha Incpl university,other
8   1900085 European/Pakeha Incpl university,other
9   1900097             NAV                    <NA>
10  1900115 European/Pakeha Incpl university,other
```

Subsetting in calculations

Gender frequencies:

```
table(issp.df$Gender)
```

```
     Female        Male NA, refused
        607         418            22
```

Let's exclude those records with missing gender.

```
exclude.rows <- which(issp.df$Gender == "NA, refused")
exclude.rows
```

```
 [1]    9   31   49   72   79   98  141  226  269  271
[11]  377  382  522  538  540  705  759  760  829  881
[21] 1025 1035
```

Subsetting in calculations

```
issp.df$Gender[exclude.rows]

 [1] "NA, refused" "NA, refused" "NA, refused"
 [4] "NA, refused" "NA, refused" "NA, refused"
 [7] "NA, refused" "NA, refused" "NA, refused"
[10] "NA, refused" "NA, refused" "NA, refused"
[13] "NA, refused" "NA, refused" "NA, refused"
[16] "NA, refused" "NA, refused" "NA, refused"
[19] "NA, refused" "NA, refused" "NA, refused"
[22] "NA, refused"

table(issp.df$Gender[-exclude.rows])


Female   Male
   607    418
```

Subsetting in calculations

```
with(issp.df, table(Income, Gender))
```

	Gender		
Income	Female	Male	NA, refused
$10000 or less	177	57	4
$10001-$15000	115	35	2
$15001-$20000	49	29	2
$20001-$25000	65	50	0
$25001-$30000	71	48	2
$30001-$40000	59	70	4
$40001-$50000	27	47	2
$50001-$70000	7	27	1
$70001-$100000	4	35	2
NAV; NAP No own income	33	20	3

Subsetting in calculations

Produce the last table with known gender and income

```
exclude.rows1 <-
  with(issp.df,
       which(Gender == "NA, refused"  |
             Income == "NAV; NAP No own income"))
issp.df[exclude.rows1, c("Gender", "Income")]
```

```
           Gender                    Income
9     NA, refused NAV; NAP No own income
13         Female NAV; NAP No own income
19           Male NAV; NAP No own income
22           Male NAV; NAP No own income
27         Female NAV; NAP No own income
31    NA, refused           $25001-$30000
37           Male NAV; NAP No own income
49    NA, refused          $10000 or less
72    NA, refused          $70001-$100000
```

Subsetting in calculations

Two-way frequency table, excluding cases of unknown Gender and uknown or no Income.

```
I.G.tab <- with(issp.df[-exclude.rows1, ],
                table(Income, Gender))
I.G.tab
```

```
                Gender
Income           Female Male
  $10000 or less    177   57
  $10001-$15000     115   35
  $15001-$20000      49   29
  $20001-$25000      65   50
  $25001-$30000      71   48
  $30001-$40000      59   70
  $40001-$50000      27   47
  $50001-$70000       7   27
  $70001-$100000      4   25
```

Subsetting in calculations

Convert counts to percentages rounded to 1 decimal place.

```
round(prop.table(I.G.tab)*100, 1)
```

```
                 Gender
Income            Female Male
  $10000 or less    18.2  5.9
  $10001-$15000     11.8  3.6
  $15001-$20000      5.0  3.0
  $20001-$25000      6.7  5.1
  $25001-$30000      7.3  4.9
  $30001-$40000      6.1  7.2
  $40001-$50000      2.8  4.8
  $50001-$70000      0.7  2.8
  $70001-$100000     0.4  3.6
```

Easier way

```
I.G.tab1 <- with(issp.df, table(Income, Gender))
dim(I.G.tab1)

[1] 10  3

row.names(I.G.tab1)

 [1] "$10000 or less"
 [2] "$10001-$15000"
 [3] "$15001-$20000"
 [4] "$20001-$25000"
 [5] "$25001-$30000"
 [6] "$30001-$40000"
 [7] "$40001-$50000"
 [8] "$50001-$70000"
 [9] "$70001-$100000"
[10] "NAV; NAP No own income"
```

Easier way

```
colnames(I.G.tab1)

[1] "Female"       "Male"         "NA, refused"

which(row.names(I.G.tab1) == "NAV; NAP No own income")

[1] 10

which(colnames(I.G.tab1) == "NA, refused")

[1] 3
```

Easier way

```
I.G.tab1[-10, -3]
```

```
                Gender
Income           Female Male
  $10000 or less    177   57
  $10001-$15000     115   35
  $15001-$20000      49   29
  $20001-$25000      65   50
  $25001-$30000      71   48
  $30001-$40000      59   70
  $40001-$50000      27   47
  $50001-$70000       7   27
  $70001-$100000      4   35
```

Easier way

```
round(prop.table(I.G.tab1[-10, -3])*100, 1)
```

```
                 Gender
Income            Female Male
  $10000 or less    18.2  5.9
  $10001-$15000     11.8  3.6
  $15001-$20000      5.0  3.0
  $20001-$25000      6.7  5.1
  $25001-$30000      7.3  4.9
  $30001-$40000      6.1  7.2
  $40001-$50000      2.8  4.8
  $50001-$70000      0.7  2.8
  $70001-$100000     0.4  3.6
```

More subsetting in calculations

```
#Mean age for everybody
with(issp.df, mean(Age, na.rm = TRUE))

[1] 45.77179

#Mean age for male
with(issp.df, mean(Age[Gender == "Male"], na.rm = TRUE))

[1] 45.95204

#Mean age for male whose annual income is
#less than "$10000"
with(issp.df, mean(Age[Gender == "Male" &
                       Income == "$10000 or less"],
                   na.rm = TRUE))

[1] 51.26316
```

Summary

- Making R functions
- Installing and loading R pacakges
- Subsetting vectors and datasets