

Univariate Data - 1. Data Vectors

Young W. Lim

2018-02-05 Mon

1 Univariate Data

- Based on
- Data Vectors

"Using R for Introductory Statistics" John Verzani

I, the copyright holder of this work, hereby publish it under the following licenses: GNU head Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled GNU Free Documentation License.

CC BY SA This file is licensed under the Creative Commons Attribution ShareAlike 3.0 Unported License. In short: you are free to share and make derivative works of the file under the conditions that you appropriately attribute it, and that you distribute it only under a license compatible with this one.

```
x <- c(1, 2, 3, 4, 5)
```

```
length(x)
```

```
sum(x)
```

```
sum(x)/length(x)
```

```
mean(x)
```

```
x - mean(x)
```

```
x^2 / length(x)
```

```
sqrt(x)
```

```
x <- c(1, 2, 3, NA, 5)
sum(x)
sum(x, na.rm = TRUE)
mean(x, na.rm=TRUE)
```

Attributes: names

```
head(precip)
head(sort(precip, decreasing=TRUE))
head(name(precip))

t <- c(a=1, b=2, c=3)
t <- setNames(c(1, 2, 3), c("a", "b", "c"))

t <- c(1, 2, 3)
names(t) <- c("a", "b", "c")
```

```
x <- c(1, "a", "I")
```

```
as.numeric("1")
```

```
as.character(1)
```

```
scan()
```

```
x <- scan("x.dat")
```


Structured Data

```
1:5  
1:length(y)  
0:(length(x)-1)  
seq(0, 100, by=10)  
seq(0, 100, length.out=11)
```

Replication

```
rep(5, times=10)
rep(1:3, times=4)
rep(c(1,2,3), times=c(3, 2, 1))
```

Numerical Indexing

```
x <- [c(1, 2, 3, 4)]  
x[1]  
x[2]  
x[55]
```

Indexing with names

```
precip[ c("Seattle Tacoma", "New York") ] # name as indices  
match( c("Seattle Tacoma", "New York"), names(precip) ) # gives number index  
precip[ "Seattle" ] # NA
```

Assignment through indexing

```
x <- c(1, 2, 3)
x[1] <- 11
x[2:3] <- c(12, 13)
x[6] <- 6 # NA NA for 4, 5
x[2:3] <- 0
x <- 1:10
x[] <- 1:3 # repeating assignment of 1 2 3
n <- 10
x[1 +0:(n-1) %% length(x)] # remainder
```

<code>x[1]</code>	the first element of <code>x</code>
<code>x[]</code>	all elements of <code>x</code>
<code>x[length(x)]</code>	the last element of <code>x</code>
<code>x[c(2,3)]</code>	the second and third elements of <code>x</code>
<code>x[-c(2,3)]</code>	all but the second and third elements of <code>x</code>
<code>x[0]</code>	0-length vector of the same type as <code>x</code>
<code>x[1] <- 5</code>	Assign a value of 5 to the first element of <code>x</code>
<code>x[c(2,3)] <- c(4,5)</code>	assign values to second and third elements of <code>x</code> in the assignment, recycling of the right-hand side may occur. assignment can grow the length of a data vector

Numeric Data Types

```
c(class(1), class(pi), class(seq(1, 5, by=1))) # numeric
```

a

```
sqrt(2) * sqrt(2)  
sqrt(2) - 2
```

Categorical Data Types

```
c("Lincoln", "said", "Seattle\ ")  
  
sprintf("X%s", 1:10)  
  
sprintf("%8d", c(1, 12, 123, 1234))  
  
paste("X", 1:10, sep="")  
  
paste("Hello", "world", sep="-")  
  
paste(c("Seattle", "Tacoma"), c("Texas", "San Antonio"), sep="_")  
  
x <- c(one=1, two=2, three=3)  
out <- paste(names(x), x, sep=":", collapse=" ")  
sprintf("[ %s ]", out)
```


- the levels of factors: a list of all possible categories for the data in the factor
- the default choice through `factor` is the collection of unique values
- can specify more levels through the `levels` argument
- cannot assign a value into a factor unless it matches a level

```
x <- paste("X", rep(1:3, 4), sep="")  
y <- factor(x)
```

```
y[1] <- "X4"
```

Adding a level

```
levels(y) <- c(levels(y), "X4")  
y[1] <- "X4"
```

```
levels(y) <- paste("label:", 1:4, sep="")
```

```
y <- factor(state.name[1:5])
```

```
levels(y) <- c("South", "West", "West", "South", "West")
```

```
y <- factor(state.name)[1:5]
```

```
factor(y, levels=y) # levels are actual values
```

Generating new factors

```
r <- "red"; b <- "blue"; g <- "green"
factor( c(r,r,r,g,g,g,b,b) )

g1(3, 5, labels=c("red", "green", "blue"))

m <- head(Cars93)
out <- m$Orign : m$AirBas

levels(out)
```

Date and time types

```
require(lubridate)

current_time <- now()
class(current_time)

as.numeric(current_time)

month(current_time, label=TRUE)

x <- "02-Feb-2018 16:08:42"
y <- parse_date_time(x, dbYHMS")
year y

now() - day(1)
now() - hours(24)
```

Logical data

```
is.na(1)
is.numeric("one")
is.logical("false")
```

```
3 < pi
"one" == 1
```

```
sqrt(2)*sqrt(2) == 2
```

```
isTRUE(all.equal(sqrt(2)*sqrt(2), 2))
```

```
whale <- c(74, 232, 135, 165, 79)
whale > 100
```

```
whale == 111
```

```
whale < 100 | whale > 200
whale > 100 & whale < 200
```

Functions

```
any(whale > 300)
all(whale > 50)
which(whale < 100 | whale > 200)
74 %in% whale
any(292 == whale)
match(c(292, 293), whale)
```

Coercion and indexing

```
sum(whale > 200)
```

```
whale[ whale > mean(whale) ]
```

```
whale[ whale < mean(whale) - sd(whale) | whale > mean(whale)+sd(whale) ]
```

```
hip_cost[ !is.na(hip_cost) ]
```


