Introduction to R, Part I

Basic math, variables, and variable types

Tim Frasier

Is a (powerful) calculator

Calculator

Standard symbols

Symbol	Function
+	Add
-	Subtract
*	Multiply
/	Divide
sqrt	Square root
^	Power

Calculator
Standard symbols

```
1 + 1
[1] 2
```

```
17 - 15
[1] 2
```

```
2 * 1
[1] 2
```

```
4 / 2
[1] 2
```

Calculator
Standard symbols

```
sqrt(9)
```
```
[1] 3
```

```
5^6
```
```
[1] 15625
```

```
9^(1/2)
```
```
[1] 3
```

Calculator
Parentheses can clarify order of processes

```
1 + 1 * 20 / 10 * 3
[1] 7
```

```
1 + ((1 * 20) / (10 * 3))
[1] 1.666667
```

If order of operations is not indicated by parentheses, R proceeds in the following order:

1. Power
2. Multiplication & division (left to right)
3. Addition & subtraction (left to right)

Calculator
Parentheses can clarify order of processes

```
1 + 1 * 20 / 10 * 3
[1] 7
```

```
1 + ((1 * 20) / (10 * 3))
[1] 1.666667
```

No limit to the degree of “nesting” with parentheses, as long as you can keep it straight yourself

Variables

Variables

- Can assign any value, object, or process to a variable
- The variable name just becomes a “code name” for whatever you have assigned to it

Variables

```
peanutbutter = 5
```

- Assigns the value “`5`” to the variable “`peanutbutter`”

Variables

```
peanutbutter = 5
```

- Assigns the value "5" to the variable "`peanutbutter`"

Used for assignment!!!

Does **not** mean equals (that would be ==).

Can also use <-

Variables

```
peanutbutter = 5
```

- Assigns the value "5" to the variable "peanutbutter"

- Can check by typing peanutbutter at the R prompt and pressing <ENTER>

```
peanutbutter
[1] 5
```

Variables

```
jelly = 9
```

- Assigns the value "`9`" to the variable "`jelly`"
- Can check by typing `jelly` at the R prompt and pressing `<ENTER>`

```
jelly
[1] 9
```

Variables

- Can assign the result of a process to a variable (without knowing what that result is ahead of time)

```
bread = (6 + 8) / 2
```

```
bread
[1] 7
```

Variables

- Can use variables in any way that you would use their values

```
peanutbutter + jelly + bread
[1] 21
```

```
lunch = peanutbutter + jelly + bread
```

```
(sqrt(jelly) + bread) / peanutbutter^3
[1] 0.08
```

Variables

Naming habits

- It may be tempting to give your variables simple names
  - a, b, c, etc.
  - one, two, three, etc.

Don’t do this!!!

- Should be clear and meaningful
  - Make your code clear in 1 week, 1 month, 1 year, etc.
  - Make your code clear to others who may be using (or grading) it

Variable Types: Vectors

Variable Types

Vectors

- An ordered list of elements of the same type.
- Created using the combine function (`c`)

Variable Types

Vectors

- An ordered list of elements of the same type.
- Created using the combine function (`c`)
- A vector of numbers

```
numList = c(1, 2, 3, 4, 5)
```

```
numList
[1] 1 2 3 4 5
```

Variable Types

Vectors

- Can use colon to indicate all integers in between two values

```
numList = 1:5
```

```
numList
[1] 1 2 3 4 5
```

Variable Types

Vectors

- Or combine both

```
numList = c(0, 1:5, 6, 10, 11:13)
```

```
numList
[1] 0 1 2 3 4 5 6 10 11 12 13
```

Variable Types

Vectors

- Can also generate vectors with the sequence function (`seq`)
- Can specify by how much to increment list

```
numList = seq(from = 1, to = 5, by = 0.5)
```

```
numList
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
```

Variable Types

Vectors

- Can also generate vectors with the sequence function (`seq`)
- Or, can specify how long the vector should be, and R will increment values accordingly

```
numList = seq(from = 10, to = 100, length = 5)
```

```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```

Variable Types

Vectors

- Can also generate vectors with the replicate function (`rep`)

```
numList = rep(1, times = 10)
```

```
numList
[1] 1 1 1 1 1 1 1 1 1 1
```

Variable Types

Vectors

- Can refer to specific positions of a vector using their number
  - First, let's make an interesting list

```
numList = seq(from = 10, to = 100, length = 5)
```

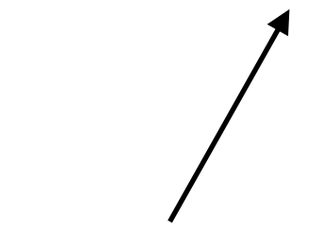
```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```

Variable Types

Vectors

- Can refer to specific positions of a vector using their number

```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```

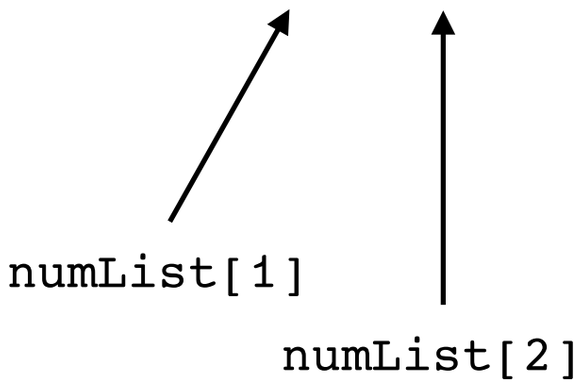


Variable Types

Vectors

- Can refer to specific positions of a vector using their number

```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```

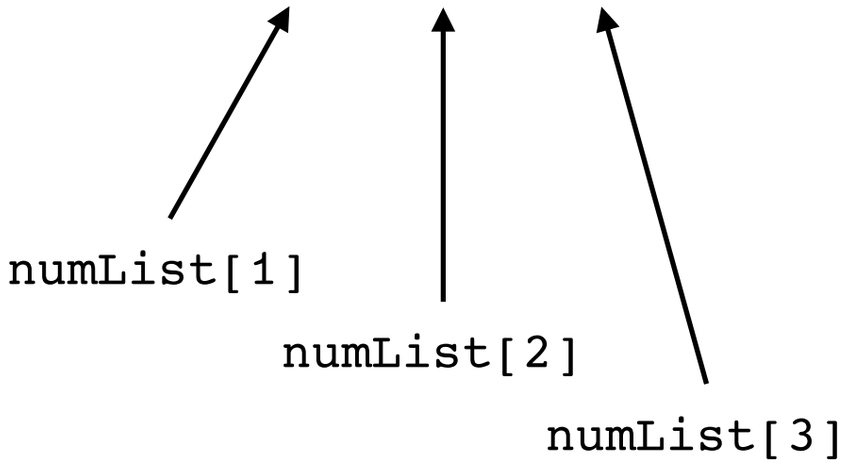


Variable Types

Vectors

- Can refer to specific positions of a vector using their number

```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```

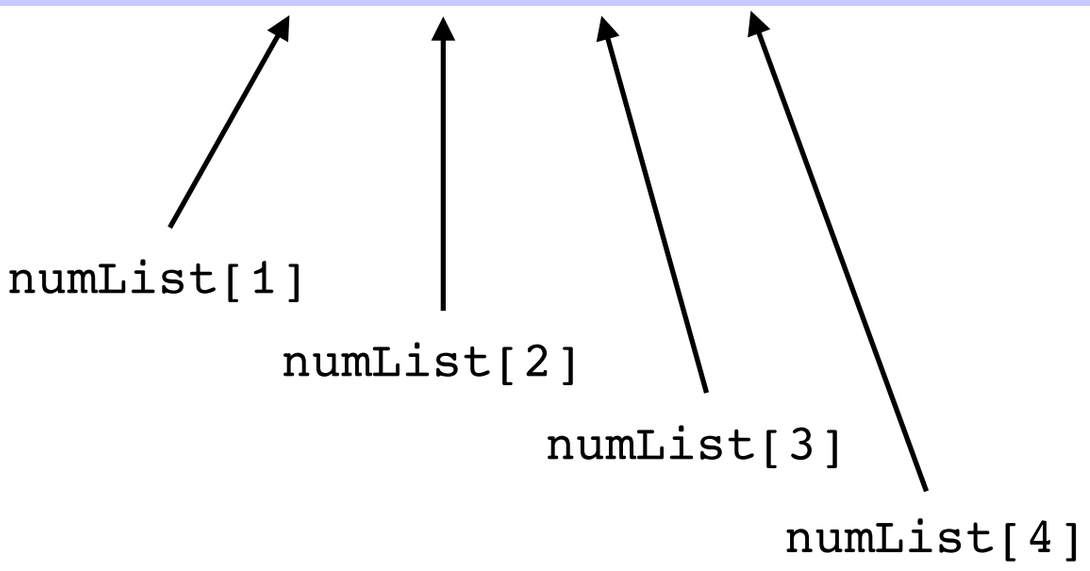


Variable Types

Vectors

- Can refer to specific positions of a vector using their number

```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```

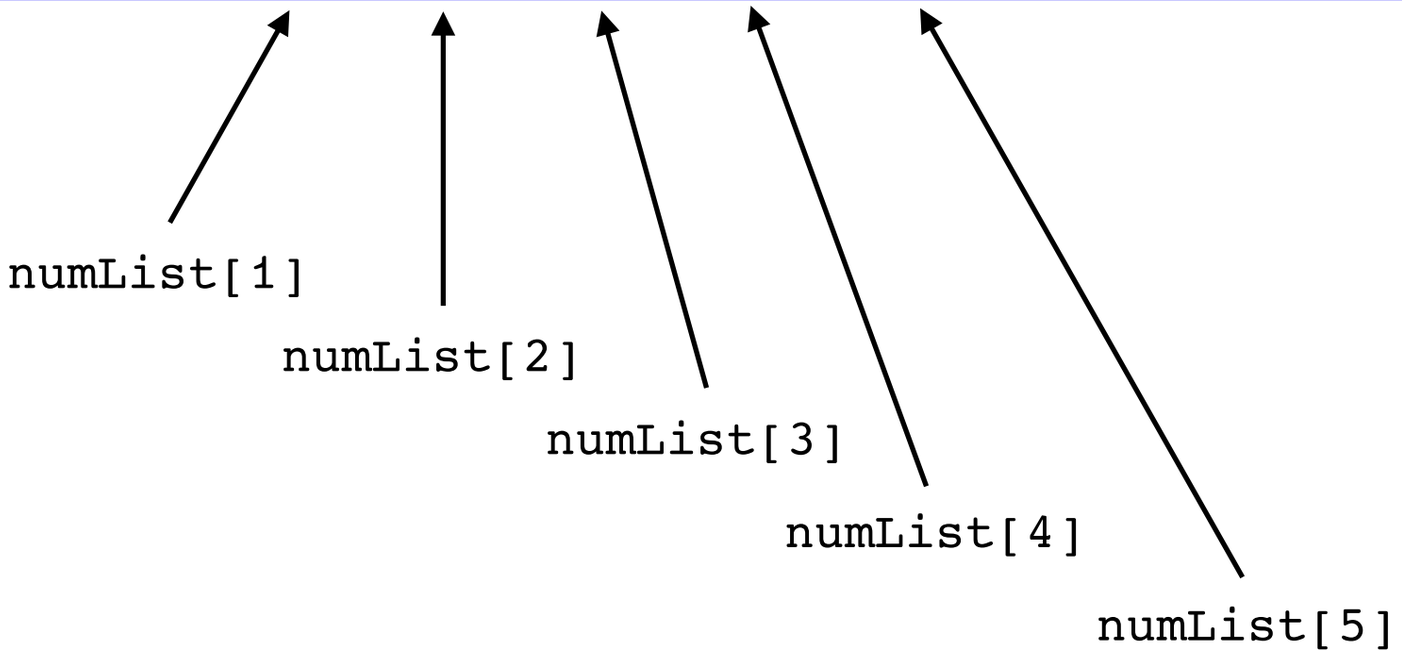


Variable Types

Vectors

- Can refer to specific positions of a vector using their number

```
numList
[1] 10.0 32.5 55.0 77.5 100.0
```



Variable Types

Vectors

- Can refer to specific positions of a vector using their number

```
numList[3]
[1] 55
```

```
numList[5]
[1] 100
```

Variable Types

Vectors

- Can have a vector of any data type

```
letterList = c("A", "B", "C", "D")
```

```
letterList
[1] "A" "B" "C" "D"
```

Note that you need to enclose text
with quotation marks

Variable Types

Vectors

- Can refer to vector positions in the same way

```
letterList[2]
[1] “B”
```

```
letterList[4]
[1] “D”
```

Variable Types: Data Frames

Variable Types

Data Frame

- A “worksheet” of values, organized with instances as rows, and different value types as different columns
  - Like an Excel worksheet
  - All columns must have the same number of rows

Variable Types

Data Frame

- `agefat.csv`
- Age and body fat percentages of 25 normal adults[1]
- Part of the HSAUR package[2]

---

1. Mazess et al. (1984) Am. J. Clin. Nutr. 40: 834
2. Everitt & Hothorn (2006) *A Handbook of Statistical Analysis Using R*

Variable Types

Data Frame

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Each row contains data for a different individual

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Each column contains a different type of data

Each row contains data for a different individual

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

- Read data into R

```
afdata = read.table("agefat.csv", header = TRUE, sep = ",")
```

Clicking on variable here…

RStudio

Go to file/function

Project: (None)

afdata

25 observations of 4 variables

	X	age	fat	sex
1	1	24	15.5	male
2	2	37	20.9	male
3	3	41	18.6	male
4	4	60	28.0	male
5	5	31	34.7	female
6	6	39	30.2	female
7	7	58	21.3	male
8	8	23	9.5	male
9	9	23	27.9	female
10	10	27	7.8	male
11	11	27	17.8	male
12	12	39	31.4	female
13	13	41	25.9	male
14	14	45	27.4	male

Environment | History

Import Dataset | Clear | List

Global Environment

Data

afdata — 25 obs. of 4 variables

Files | Plots | Packages | Help | Viewer

Zoom | Export | Clear All

Console ~/R/

```
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> agefat <- read.table("agefat.csv", header = TRUE, sep = ",")
> View(agefat)
> afdata <- read.table("agefat.csv", header = TRUE, sep = ",")
> View(afdata)
> View(agefat)
> View(agefat)
> afdata <- read.table("agefat.csv", header = TRUE, sep = ",")
> View(afdata)
>
```

…will show you data here

Variable Types

Data Frame

- Can refer to specific parts of a data frame in two ways (for now)
  - Using field names
  - Using row and field numbers

Variable Types

Data Frame

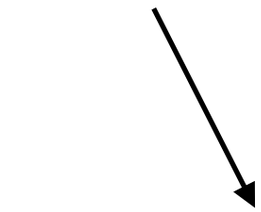
- To refer to a specific column ("field") in a data frame, use:

```
dataframename $ fieldname
```

Variable Types

Data Frame

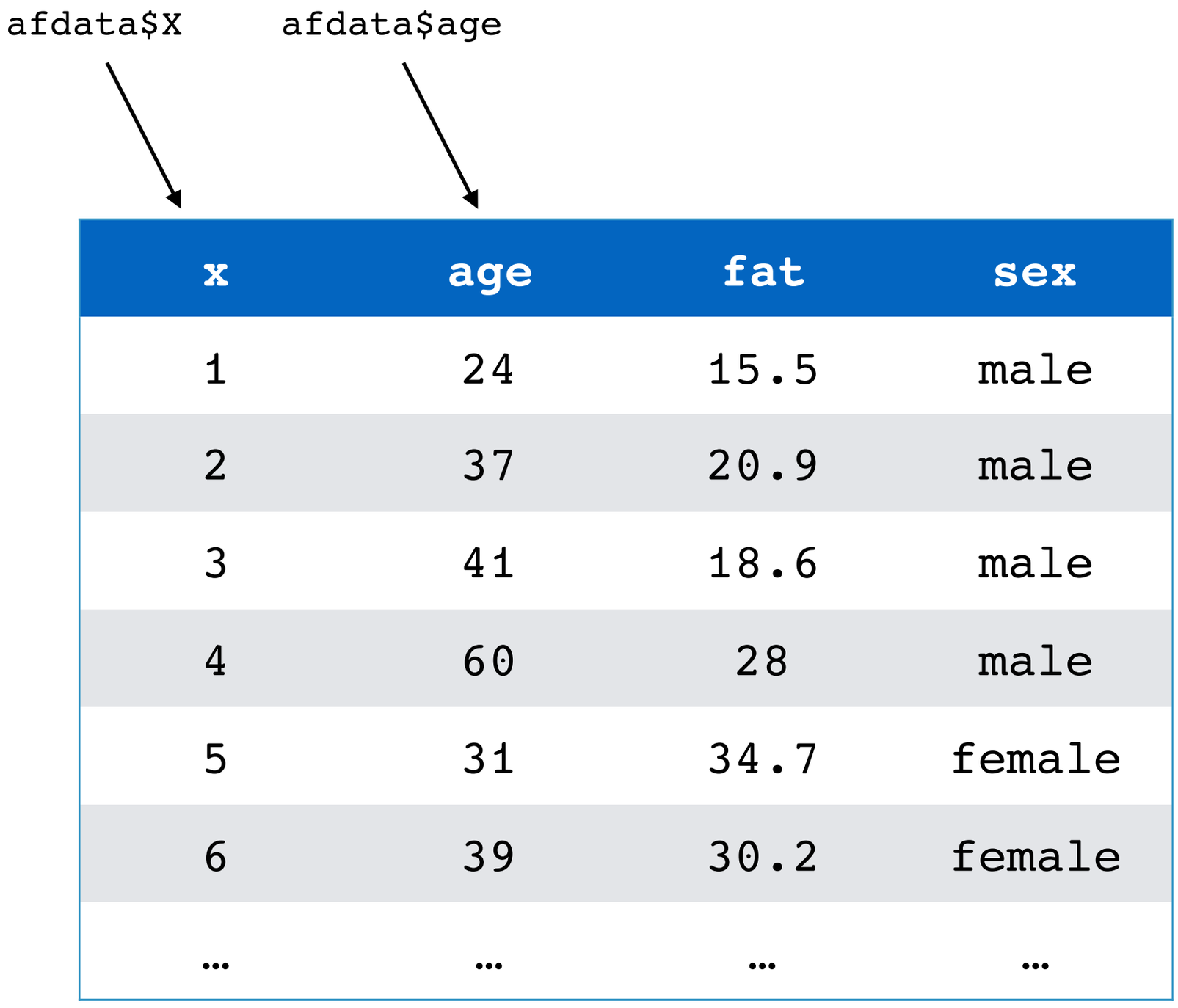
afdata$X



x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

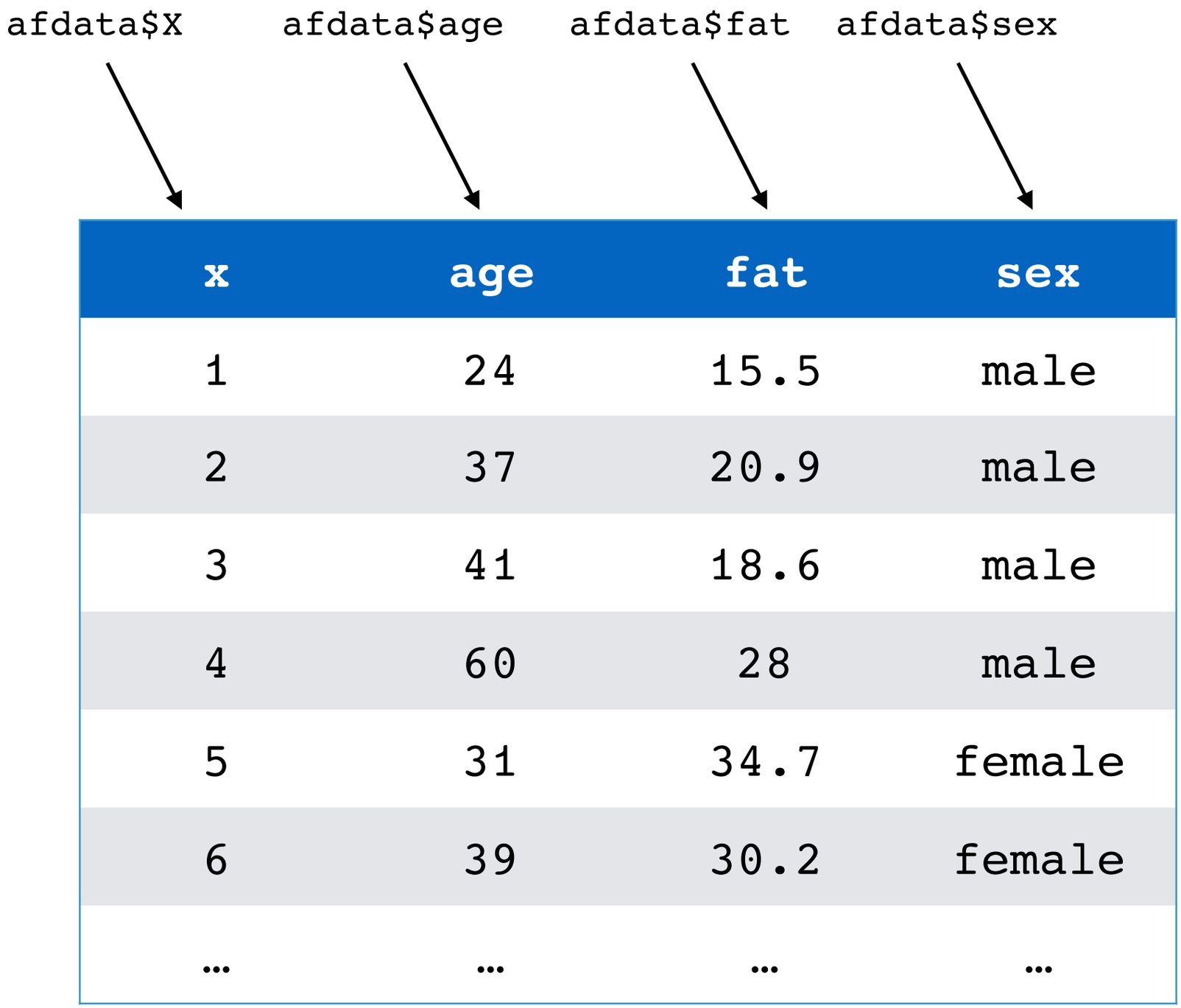
Data Frame



x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame



x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

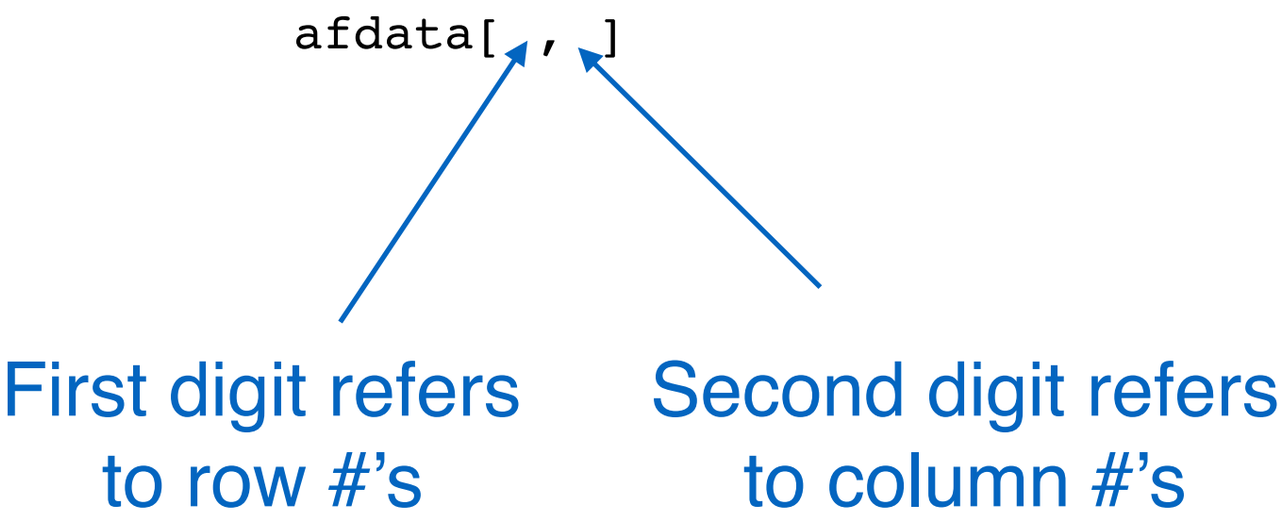
- Can also refer to specific rows, columns, and cells, using brackets

```
afdata[ , ]
```

Variable Types

Data Frame

- Can also refer to specific rows, columns, and cells, using brackets



Blank means “all”

Variable Types

Data Frame

`afdata[1, ]`

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

`afdata[2, ]`

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

```
afdata[3, ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

```
afdata[, 1]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

```
afdata[, 2]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

```
afdata[, 3]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

`afdata[6, 2]`

Variable Types

Data Frame

?

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

```
afdata[5:6, 2:3]
```

Variable Types

Data Frame

- Can combine methods to obtain different subsets of the data

Symbol	Function
==	Equal to
!=	Not equal to
>	Greater than
<	Less Than
>=	Greater than or equal to
<=	Less than or equal to
&	And
|	Or

Variable Types

Data Frame

Data for all males

```
afdata[afdata$sex == “male”, ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for all females

```
afdata[afdata$sex == “female”, ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for everyone 30 or older

```
afdata[afdata$age >= 30, ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for all males 30 or older

?

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for all males 30 or older

```
afdata[afdata$sex == “male” & afdata$age >= 30, ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for all males or individuals 30 or older

```
afdata[afdata$sex == “male” | afdata$age >= 30, ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for all males, or females 30 or older

?

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types

Data Frame

Data for all males, or females 30 or older

```
afdata[afdata$sex == “male” | (afdata$sex == “female” & afdata$age >= 30), ]
```

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Variable Types: Matrix & Array

Variable Types

Matrix

- A two-dimensional array of values of the same type
  - Whereas data frames can store data of different types in different fields

Variable Types

Matrix

```
matrix(1:6, nrow = 2, ncol = 3, byrow = TRUE)
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

Variable Types

Matrix

```
matrix(1:6, nrow = 2, ncol = 3, byrow = FALSE)
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Variable Types

Matrix

- Can refer to specific rows, columns, and cells using brackets, as with data frames

```
mymatrix = matrix(1:6, nrow = 2, ncol = 3, byrow = FALSE)
```

Variable Types

Matrix

```
mymatrix

     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Variable Types

Matrix

```
mymatrix
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
mymatrix[1, ]
```

Variable Types

Matrix

```
mymatrix
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
mymatrix[2, ]
```

Variable Types

Matrix

```
mymatrix
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
mymatrix[, 1]
```

Variable Types

Matrix

```
mymatrix
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
mymatrix[, 2]
```

Variable Types

Matrix

```
mymatrix
```

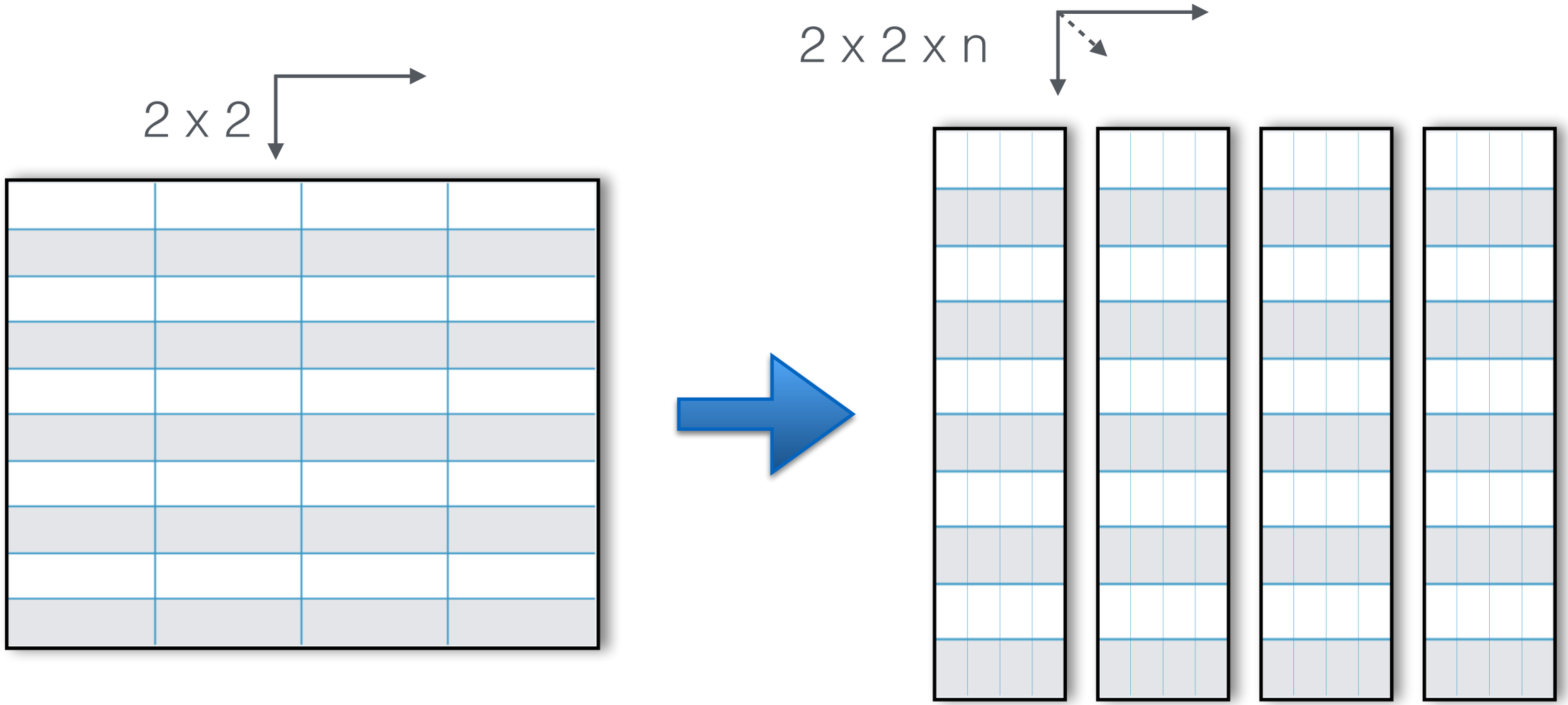
```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
mymatrix[2, 2:3]
```

Variable Types

Array

- Continuation of a matrix to multiple dimensions
  - Multiple related matrices organized in a specific fashion



Variable Types

Array

- Dimension argument (`dim`) takes 3 arguments (in this order!)
  - Rows
  - Columns
  - “Layers”

```
myarray = array(1:24, dim = c(3, 4, 2))
```

Variable Types

Array

```
myarray

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

Variable Types

Array

- Can refer to specific layers, rows, columns, and cells using brackets, as with data frames and matrices

Variable Types

Array

```
myarray

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
myarray[, , 1]
```

Variable Types

Array

```
myarray

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
myarray[, , 2]
```

Variable Types

Array

```
myarray

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
myarray[1, , 2]
```

Variable Types

Array

```
myarray

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
myarray[, 2, 2]
```

Variable Types

Array

```
myarray

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12

, , 2

     [,1] [,2] [,3] [,4]
[1,]   13   16   19   22
[2,]   14   17   20   23
[3,]   15   18   21   24
```

```
myarray[, 2, 1:2]
```

Variable Types: Factor

Variable Types

Factor

- A vector where the elements are "categorical" values that could also be ordered

- Values stored internally as integers with labeled levels

Variable Types

Factor

```
x = c(“high”, “medium”, “low”, “high”, “medium”)
```

Variable Types

Factor

```
x = c(“high”, “medium”, “low”, “high”, “medium”)
```

```
xf = factor(x)
```

```
xf
```

```
[1] high medium low high medium
Levels: high low medium
```

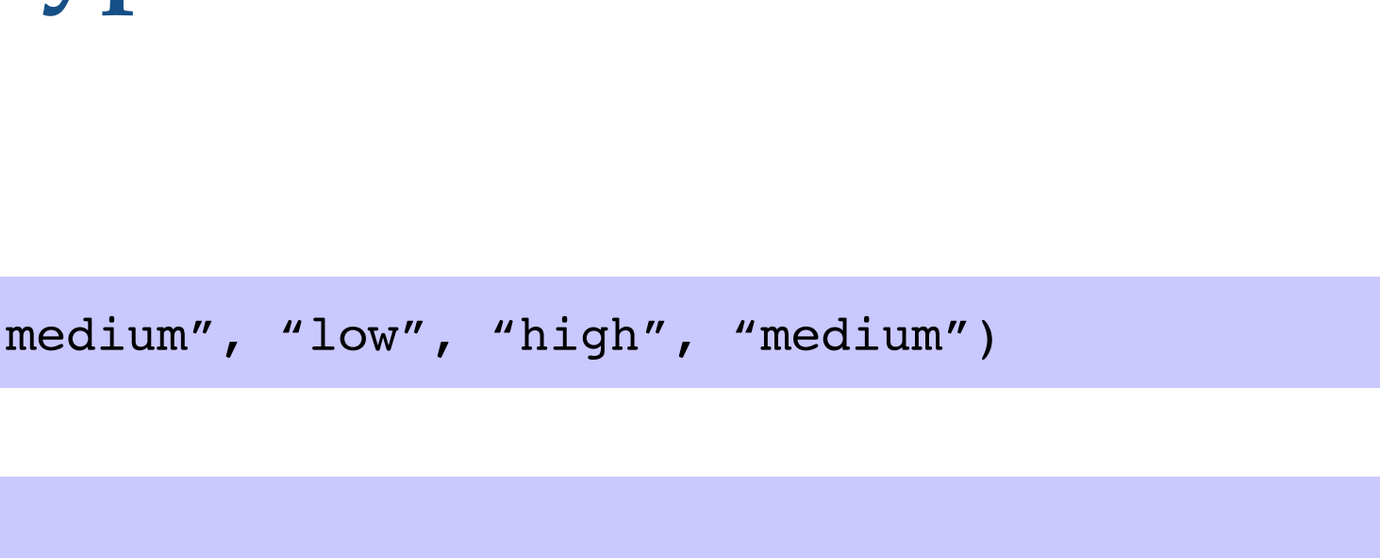
Variable Types

Factor

```
x = c(“high”, “medium”, “low”, “high”, “medium”)
```

```
xf = factor(x)
```

```
xf
[1] high medium low high medium
Levels: high low medium
```



Variable Types

Factor

```
x = c(“high”, “medium”, “low”, “high”, “medium”)
```

```
xf = factor(x)
```

```
xf
```

```
[1] high medium low high medium
Levels: high low medium
```

```
as.numeric(xf)
```

```
[1] 1 3 2 1 3
```

Variable Types

Factor

```
x = c("high", "medium", "low", "high", "medium")
```

```
xf = factor(x)
```

```
xf

[1] high medium low high medium
Levels: high low medium
```

```
as.numeric(xf)

[1] 1 3 2 1 3
```

By default, R numbers levels alphabetically

high	1
low	2
medium	3

Variable Types

Factor

- Is possible to control numerical order though

```
xfo = factor(x, levels = c("low", "medium", "high"), ordered = TRUE)
```

Variable Types

Factor

- Is possible to control numerical order though

```
xfo = factor(x, levels = c("low", "medium", "high"), ordered = TRUE)
```

```
xfo

[1] high medium low high medium
Levels: low < medium < high
```

```
as.numeric(xfo)

[1] 3 2 1 3 2
```

Variable Types:
List

Almost done!

Variable Types

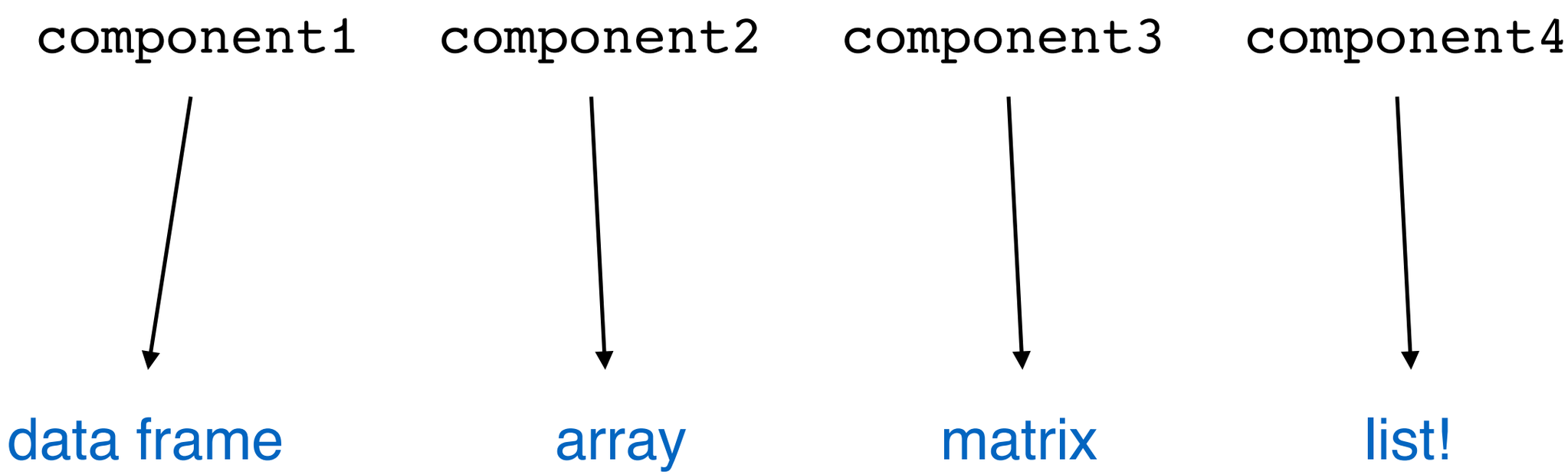
List

- A vector where the components can be different types
  - A list of things, not a list of values (that would just be a vector)

Variable Types

List

- A vector where the components can be different types
  - A list of things, not a list of values (that would just be a vector)



Variable Types

List

- Lets make 3 different variable types and combine them into a list

```
a = 1:5
```

```
b = matrix(1:6, nrow = 2, ncol = 3)
```

```
c = “When will this get interesting!?”
```

```
MyList = list(a, b, c)
```

Variable Types

List

- Let's see what this looks like

```
MyList

[[1]]
[1] 1 2 3 4 5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] "When will this get interesting!?"
```

Variable Types

List

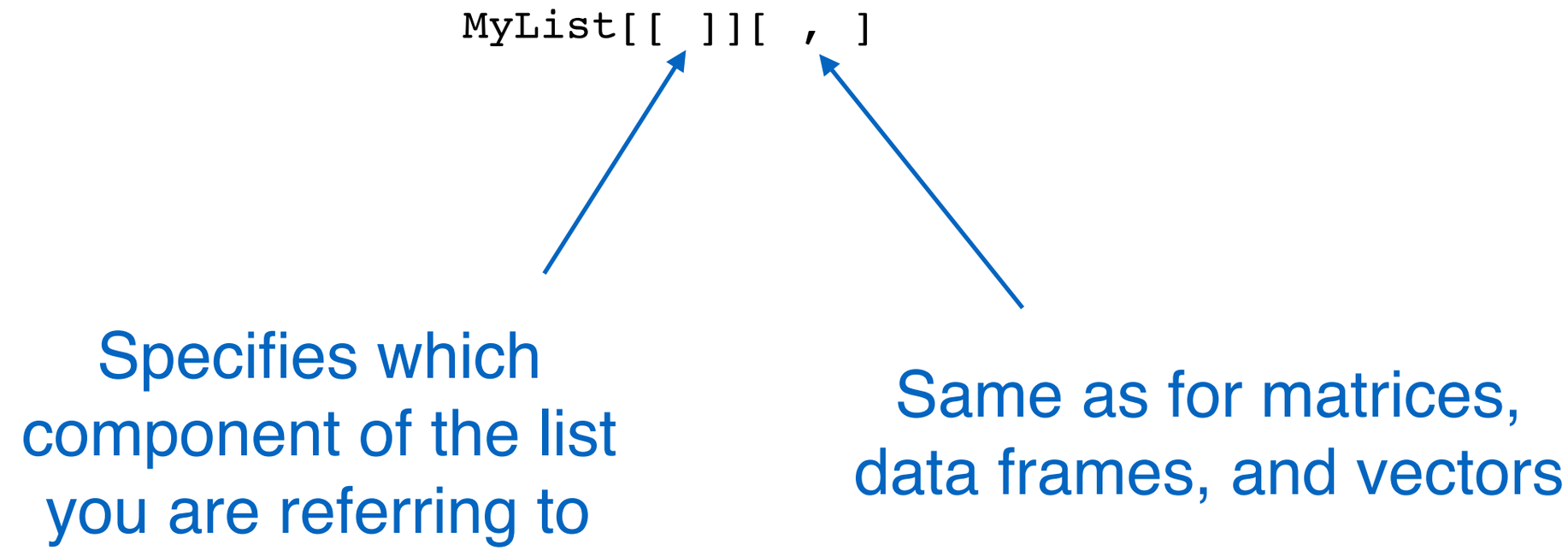
- Refer to different parts of a list in this way

```
MyList[[ ]][ , ]
```

Variable Types

List

- Refer to different parts of a list in this way



Variable Types

List

```
MyList

[[1]]
[1] 1 2 3 4 5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] “When will this get interesting!?”
```

```
MyList[[1]]
```

Variable Types

List

```
MyList

[[1]]
[1] 1 2 3 4 5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] "When will this get interesting!?"
```

MyList[[2]]

Variable Types

List

```
MyList

[[1]]
[1] 1 2 3 4 5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] "When will this get interesting!?"
```

MyList[[3]]

Variable Types

List

```
MyList

[[1]]
[1] 1  2  3  4  5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] “When will this get interesting!?”
```

```
MyList[[1]][2:3]
```

Variable Types

List

```
MyList

[[1]]
[1] 1 2 3 4 5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] “When will this get interesting!?”
```

```
MyList[[2]][1, ]
```

Variable Types

List

```
MyList

[[1]]
[1] 1 2 3 4 5

[[2]]
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

[[3]]
[1] “When will this get interesting!?”
```

```
MyList[[2]][, 2]
```

Missing Data

Last One!

Missing Data

- Can't have blank space in R "files"
- Missing data should be indicated with `NA`
- Need to ensure data are in this format prior to loading into R
  - Find and replace, but be careful!

Questions?