Lists

R Data Objects

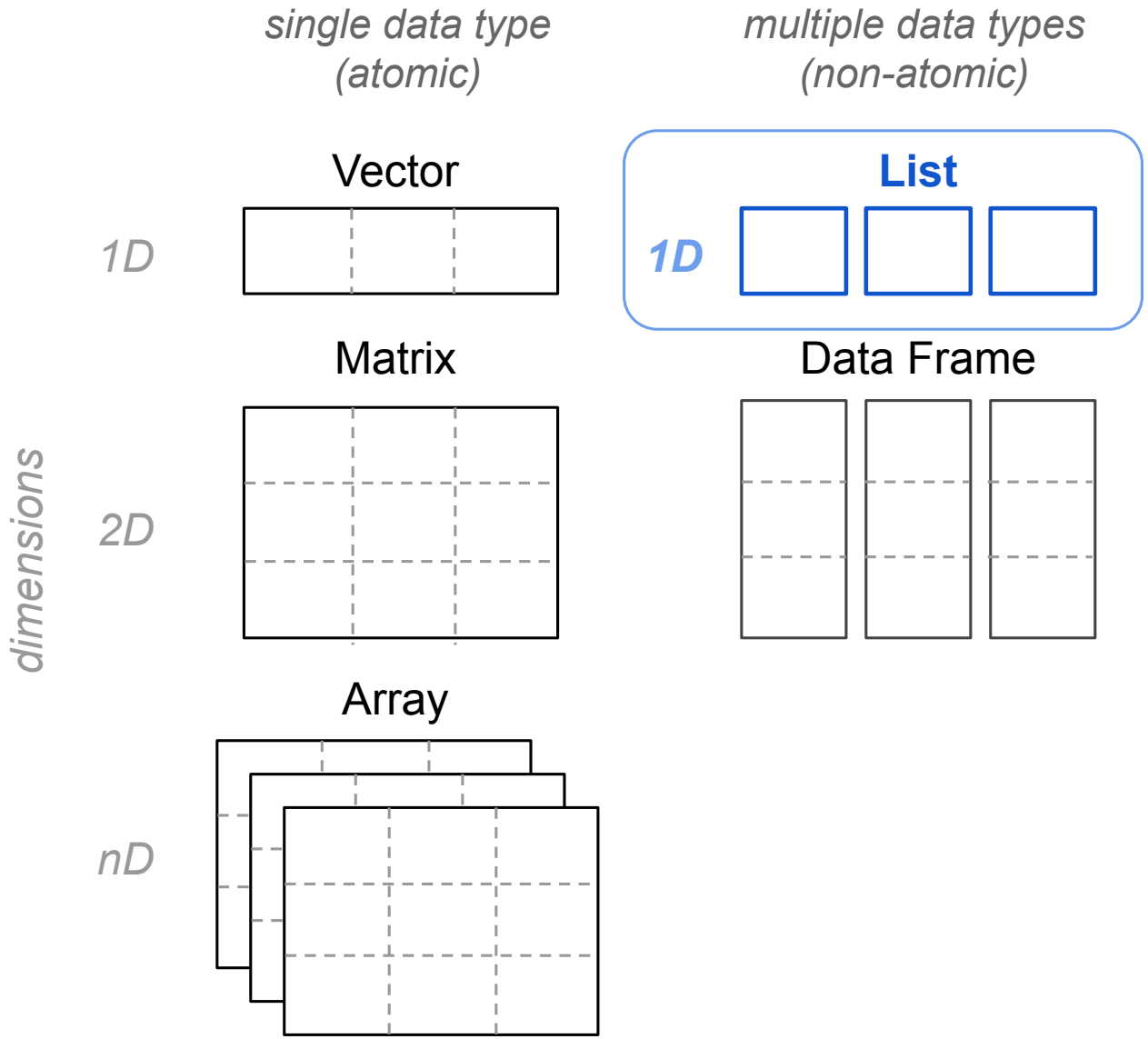
Gaston Sanchez

STAT 33B, Fall 2025

About

In these slides we talk about some basic concepts of R lists.

Basic Data Objects in R



R lists

- A list is the most general data structure in R
- Recall that an R list is a generic (non-atomic) vector
- Lists can contain any other type of data structure
- Lists can even contain other lists

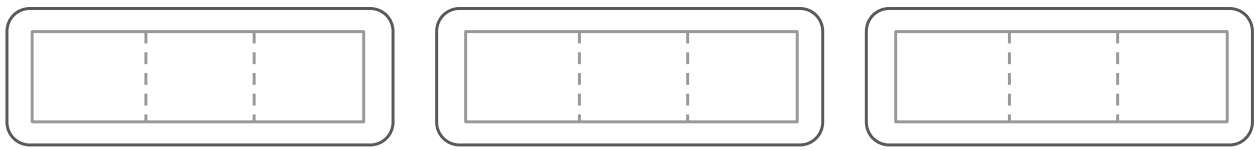
Example

To create a list we use the function `list()`

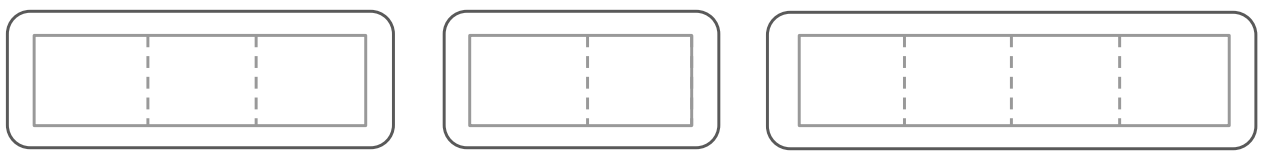
```r
# list of vectors (of equal length)
lis1 <- list(
  1:3,
  c(TRUE, FALSE, TRUE),
  c("a", "b", "c")
)

# list of vectors (of different length)
lis2 <- list(
  1:3,
  c(TRUE, FALSE),
  c("a", "b", "c", "d")
)
```

You can think of a list as a set of non-contiguous cells or boxes; each box in turn contains a certain R object.

List of Vectors (of equal length)



List of Vectors (of different length)



R lists

- Lists are a special type of vector

```
lst <- vector(mode = "list")
```

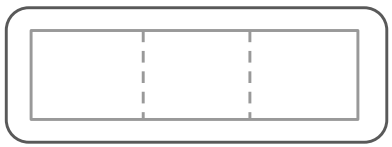
- Lists are vectors in the sense of being a one-dimensional object
- Lists are NOT atomic structures

Example

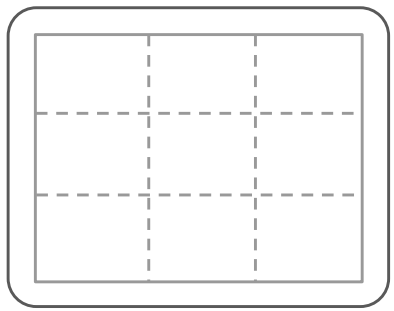
Again, a list can contain any number and any kind of R objects:

```r
# list of various objects
lis3 <- list(
  1:3,
  matrix(1:9, nrow = 3, ncol = 3),
  list(1:2, c(FALSE, TRUE), c("a", "b"))
)
```

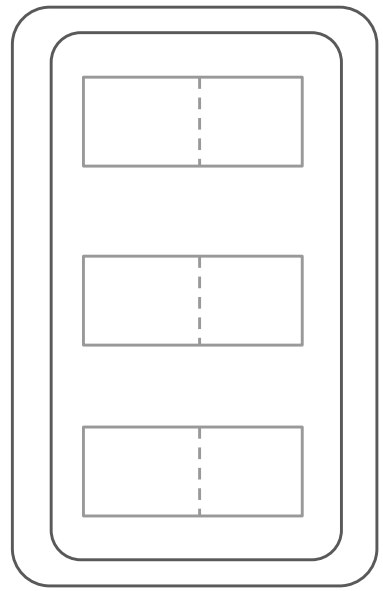
List of various objects



vector

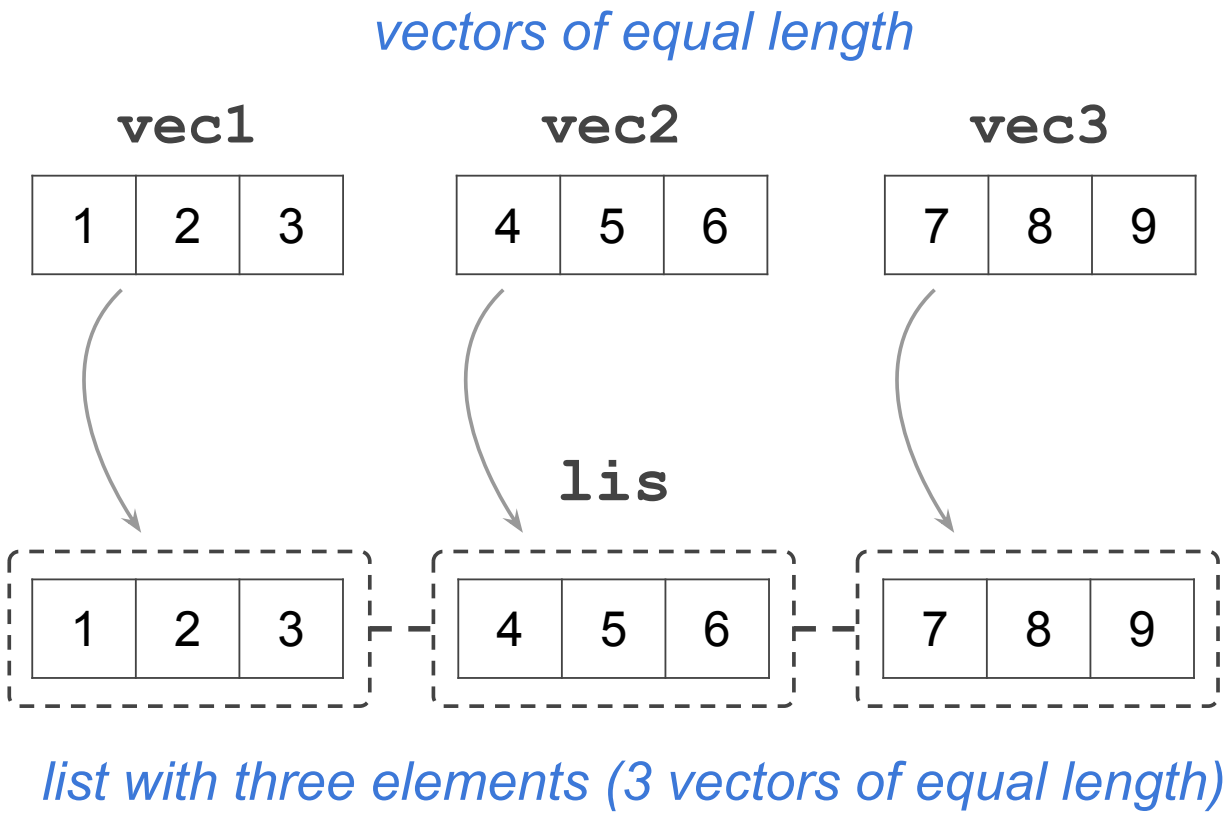


matrix



Other lists

Example: list of unnamed elements

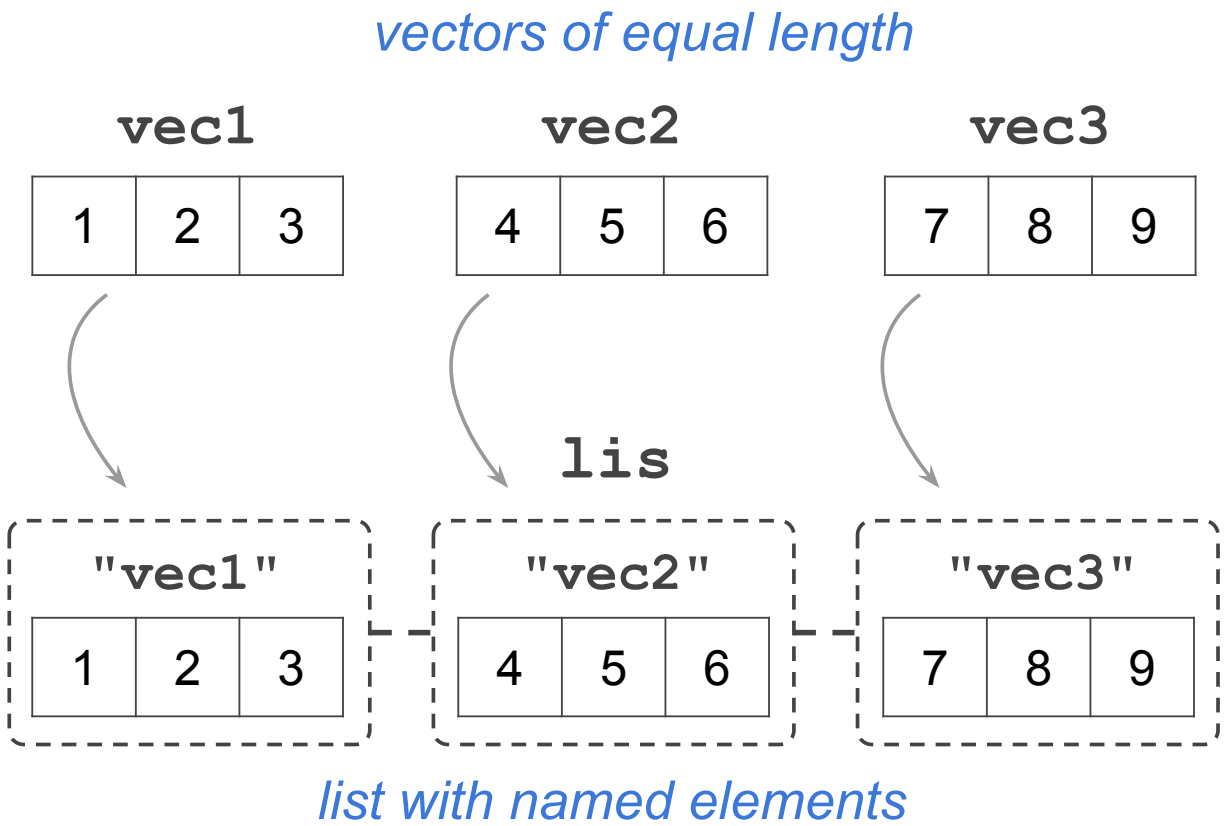


Example: list of unnamed elements

```
vec1 = 1:3
vec2 = 4:6
vec3 = 7:9

lis <- list(vec1, vec2, vec3)
lis
```

```
## [[1]]
## [1] 1 2 3
##
## [[2]]
## [1] 4 5 6
##
## [[3]]
## [1] 7 8 9
```

Example: list of named elements



Example: list of named elements

```r
vec1 = 1:3
vec2 = 4:6
vec3 = 7:9

# list with named elements (highly recommendable)
lis <- list("vec1" = vec1, "vec2" = vec2, "vec3" = vec3)
lis
```

```
## $vec1
## [1] 1 2 3
##
## $vec2
## [1] 4 5 6
##
## $vec3
## [1] 7 8 9
```

No vectorization with R lists

Lists are very convenient because they allow you to store multiple kinds of objects in a single place. This super power of lists comes with a price: you lose vectorization.

```
# vectorization reminder
vec = c(2, 4, 6, 8)
sqrt(vec)
```

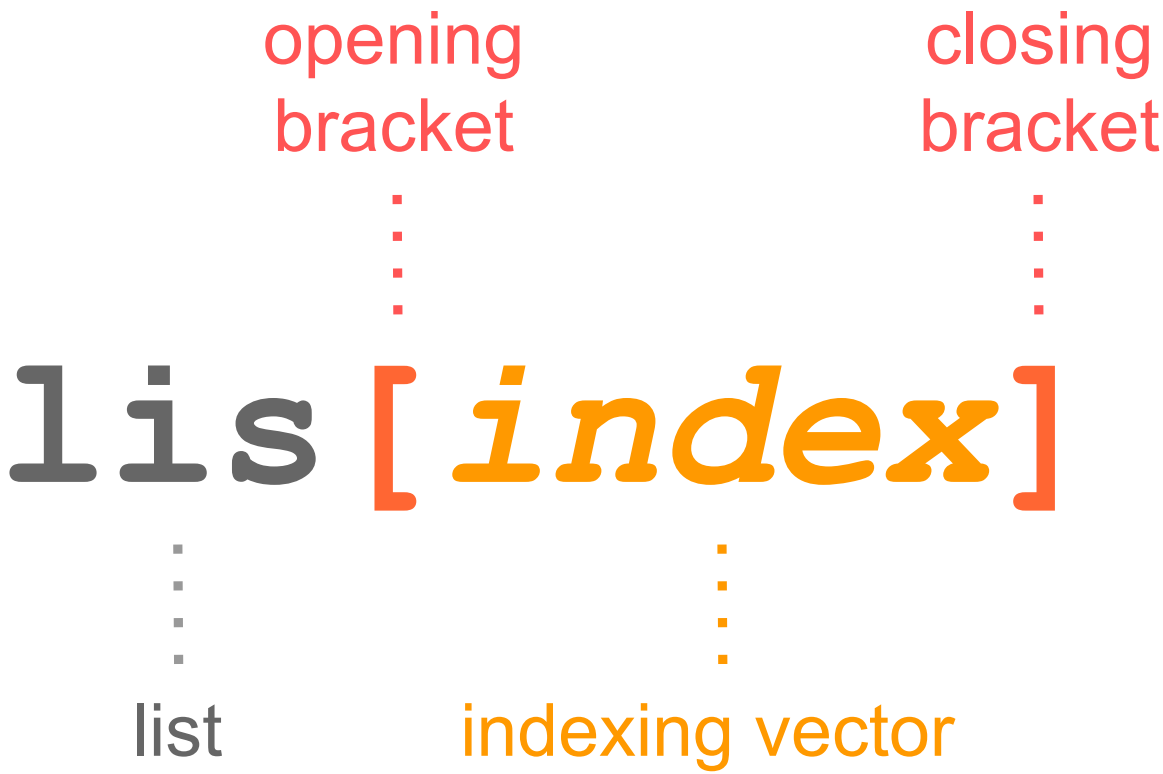
```
## [1] 1.414214 2.000000 2.449490 2.828427
```

```
# no vectorization with lists :(
lst = list(2, 4, 6, 8)
sqrt(lst)
```

```
## Error in sqrt(lst): non-numeric argument to mathematical function
```

Subsetting and Indexing

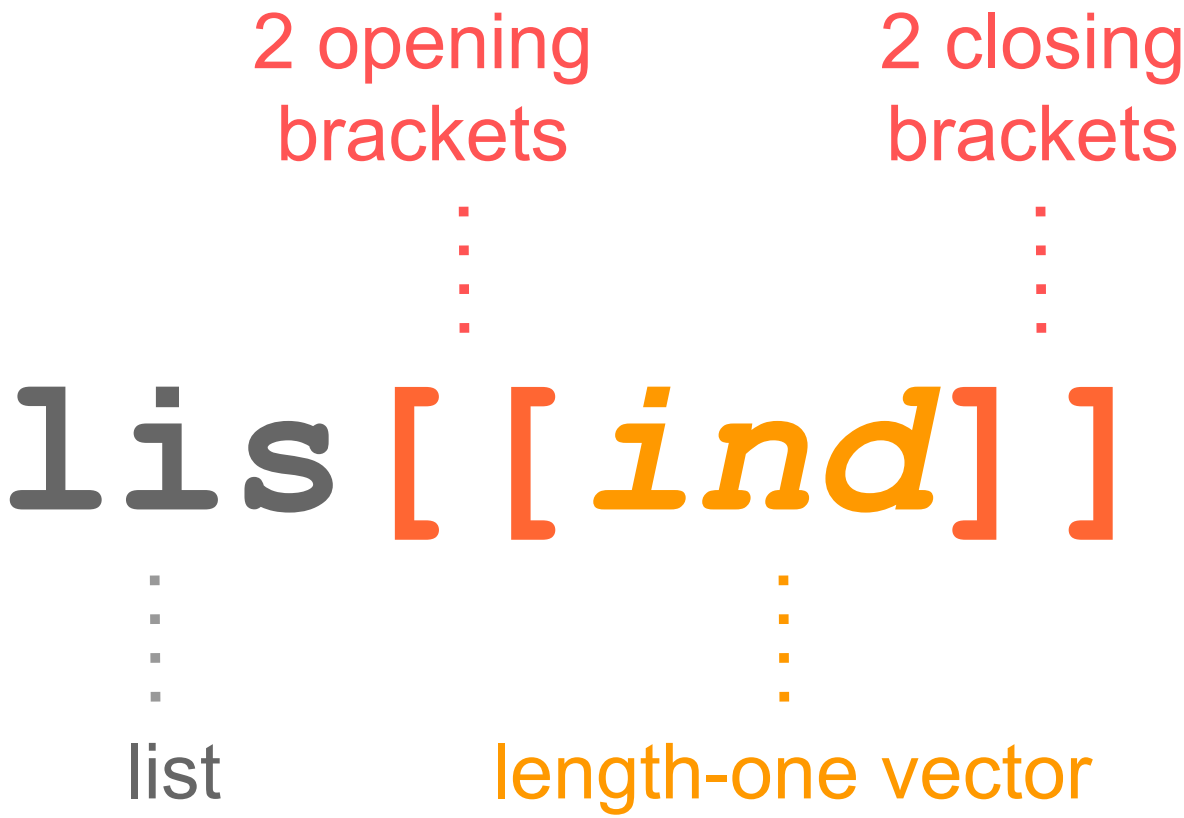
Single Brackets

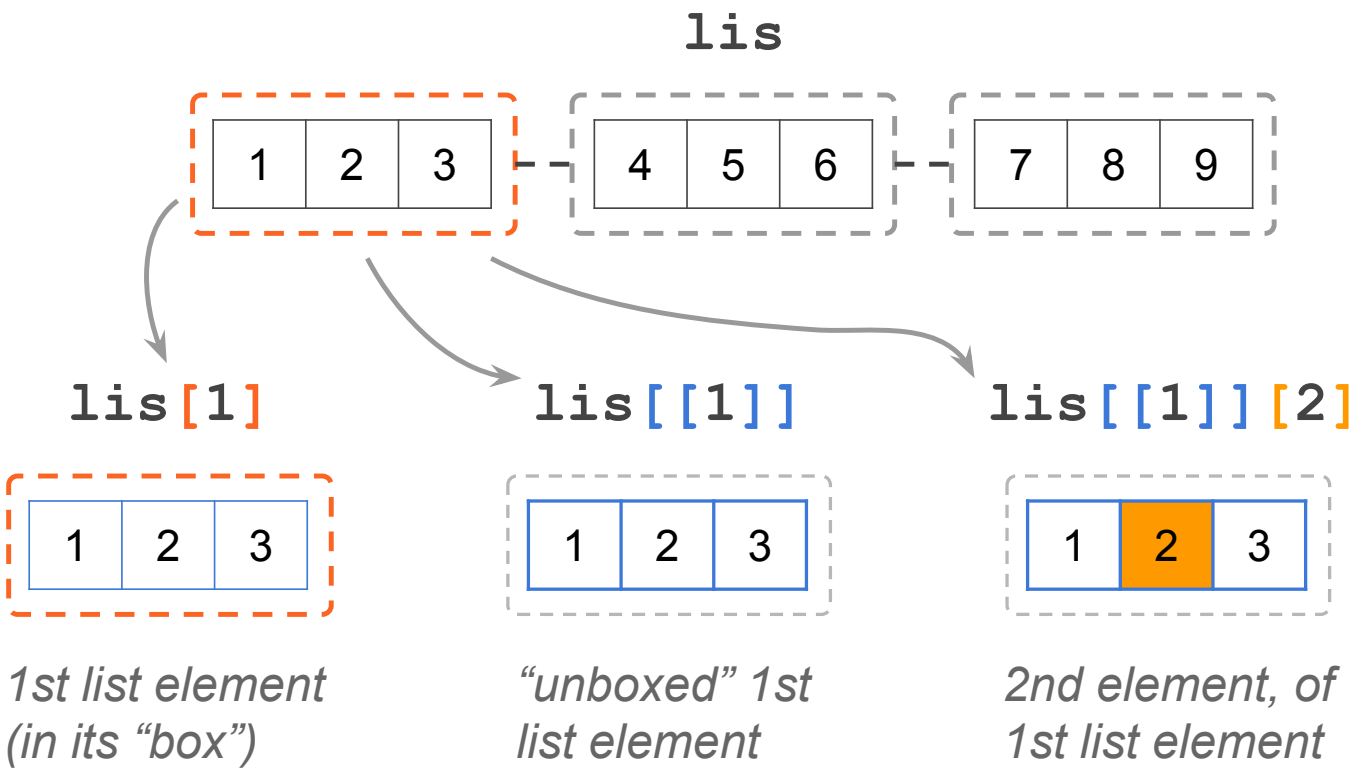


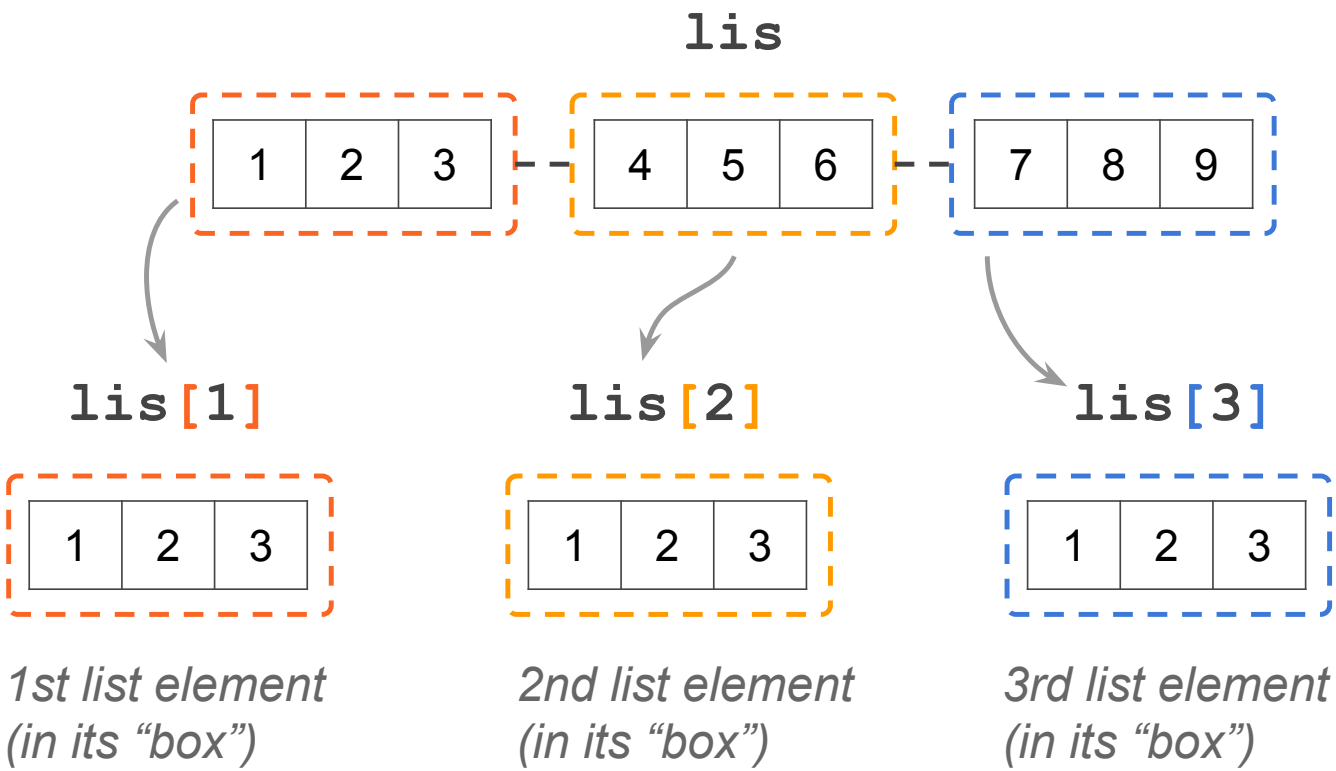
Bracket Notation for lists

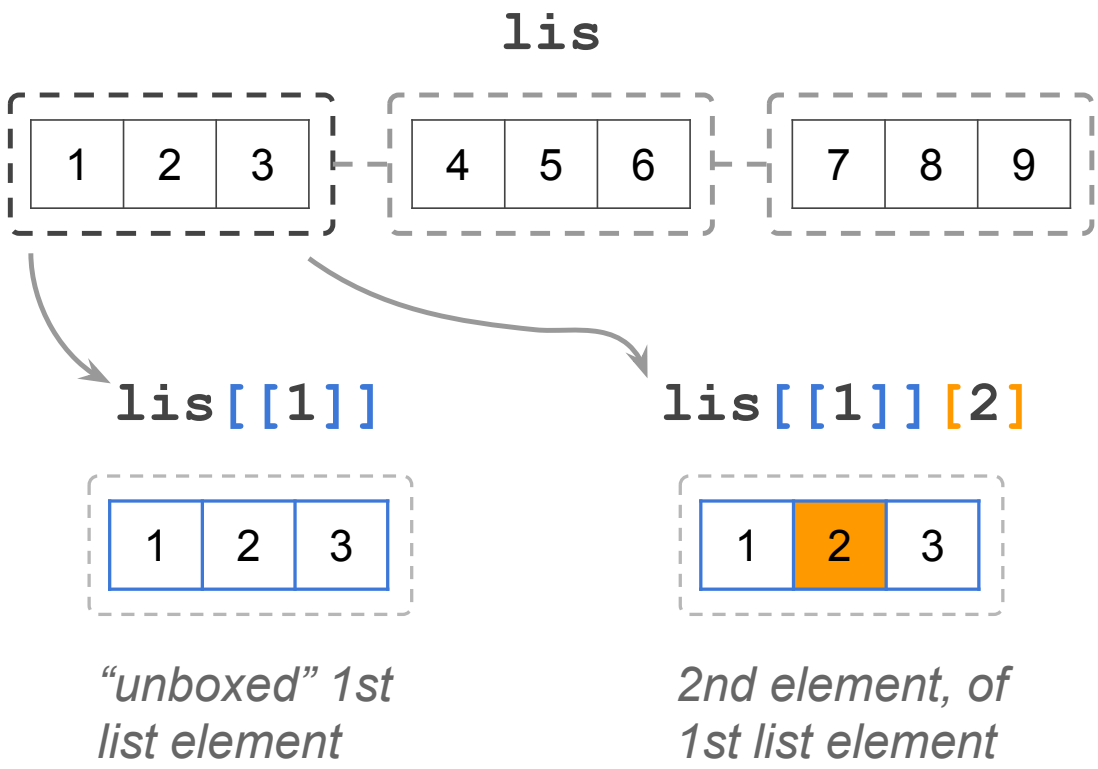
- To extract values from R lists use brackets: `[ ]`
- Inside the brackets specify an indexing vector.
- Vector(s) of indices can be numbers, logicals, and sometimes names (i.e. when you have a list with named elements).

Double Brackets




lis
1 2 3
4 5 6
7 8 9
lis[1]
lis[[1]]
lis[[1]][2]
1 2 3
1 2 3
1 2 3
1st list element
(in its “box”)
“unboxed” 1st
list element
2nd element, of
1st list element


lis
1 2 3
4 5 6
7 8 9
lis[1]
lis[2]
lis[3]
1 2 3
1 2 3
1 2 3
1st list element
(in its “box”)
2nd list element
(in its “box”)
3rd list element
(in its “box”)


lis
1
2
3
4
5
6
7
8
9
lis[[1]]
lis[[1]][2]
1
2
3
1
2
3
“unboxed” 1st list element
2nd element, of 1st list element

Dollar Operator