Advanced programming in R: Functions

In this lecture

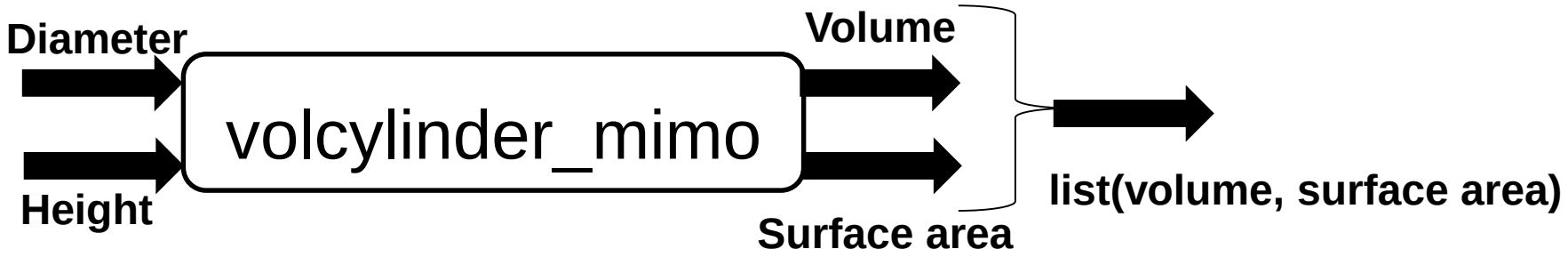
- Functions
  - MIMO
- Source (load) and call (invoke)
- Inline functions
- Looping over objects
  - apply
  - lapply
  - tapply

Multiple input and multiple output functions

Function with multiple inputs and outputs

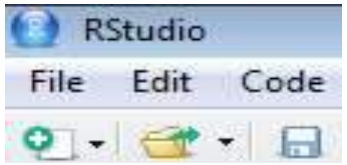
- Functions in R take multiple input objects but returns only one object as output

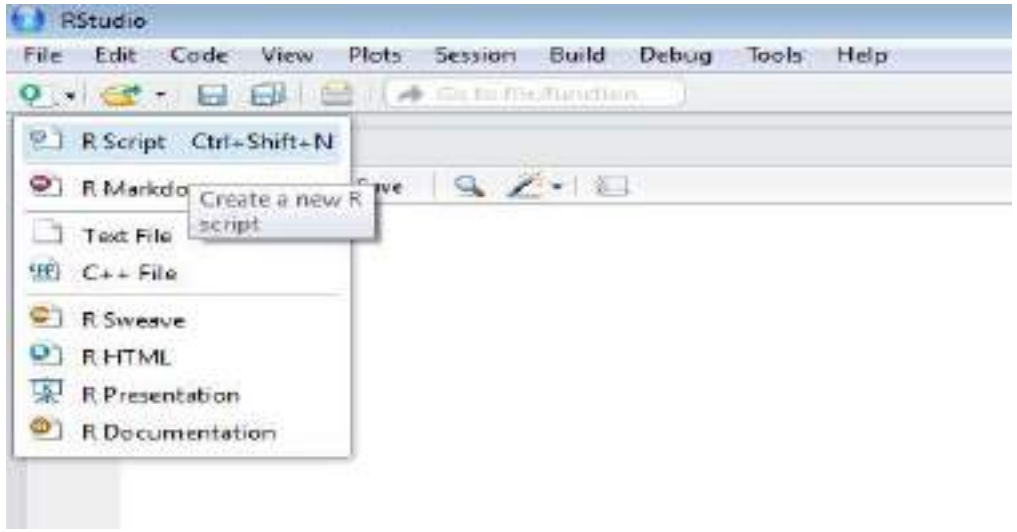
- This however is not a limitation, because a list object (collection of several objects) can be returned by function



Creating and saving

1. Creating a function file





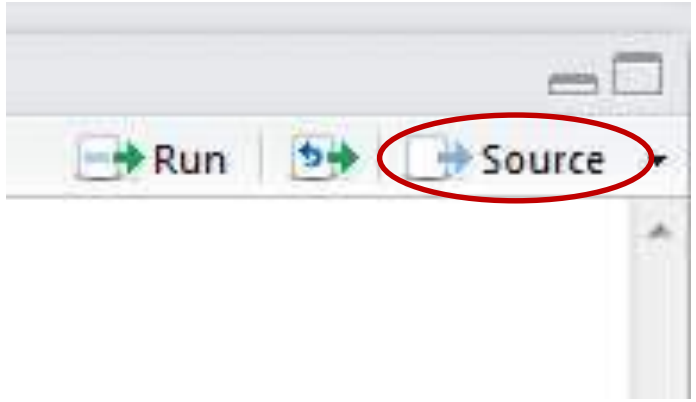

```
1 volcylinder_mimo = function(dia, len){
2   volume = (pi*dia^2)*len/4 ## Volume in metre^3
3   surface_area = pi*dia*len
4   result = list("volume"=volume, "surface_area"=surface_area)
5   return(result)
6 }
```

4:16 volcylinder_mimo

2. Save the function file “volcylinder_mimo”

Loading and invoking

3. Load the function file



4. Execute

```
> source('~/R/R-Workspace/volcylinder_mimo.R')
> result = volcylinder_mimo(10,5)
> result["volume"]
$volume
[1] 392.6991

> result["surface_area"]
$surface_area
[1] 157.0796
```

Inline functions

Example of an inline function

func = function(x) x^2+4*x+4

```
> func = function(x) x^2+4*x+4
>
> func(1)
[1] 9
>
> func(2)
[1] 16
>
> func(-2)
[1] 0
>
> func(0)
[1] 4
```

Looping over objects

- There are a few looping functions that are pretty useful when working interactively on a command line
  - **apply**: Apply a function over the margins of an array or matrix
  - **lapply**: Apply a function over a list or a vector
  - **tapply**: Apply a function over a ragged array
  - **mapply**: Multivariate version of lapply
  - **xxply** (plyr package)

apply function

- Applies a given function over the margins of a given array.
  - Syntax: apply(array, margins, function,…)
  - Here margins refer to the dimension of the array along which the function need to be applied.

```
> A = matrix(1:9,3,3)
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> apply(A,1,sum)
[1] 12 15 18
> apply(A,2,sum)
[1]  6 15 24
```

lapply function

- **lapply** is used to apply a function over a list.
- **lapply** always returns a list of the same length as the input list
  - Syntax: lapply(list, function, …)

```
> A=matrix(1:9,3,3)
> B=matrix(10:18,3,3)
> Mylist=list(A,B)
> determinant = lapply(Mylist,det)
> determinant
[[1]]
[1] 0

[[2]]
[1] 5.329071e-15
```

mapply function

- **mapply** is a multivariate version of **lapply.**
- A function can be applied over several lists simultaneously.
  - Syntax: mapply(fun, list1, list2, …)

```
> source('~/volcylinder.R')
> dia=c(1,2,3,4)
> len=c(7,4,3,2)
> vol = mapply(volcylinder,dia,len)
> vol
[1]  5.497787 12.566371 21.205750 25.132741
```

tapply function

- **tapply** is used to apply a function over subset of vectors given by a combination of factors
  - Syntax: tapply(vector, factors, function, …)

```
Console ~/
> Id = c(1,1,1,1,2,2,2,3,3)
> Values = c(1,2,3,4,5,6,7,8,9)
> tapply(Values,Id,sum)
 1  2  3
10 18 17
>
```

Id	Val	
1	1	sum = 10
1	2	
1	3	
1	4	
2	5	sum = 18
2	6	
2	7	
3	8	sum= 17
3	9	