INST0065
Data Visualization and GIS

Week 3: moving forward with R…

Dr. Oliver Duke-Williams

o.duke-williams@ucl.ac.uk
(Please use Moodle forums for messages about this module)

Twitter: @oliver_dw

INFORMATION STUDIES

Celebrating 100 years

This week

- Recap of last week's introduction to R
- More data structures
- Data frames
- Extending R
- Plot libraries

Last week

- RStudio makes it easy for us to use R
- R is a general purpose language
  - It has variables, loops, branches etc
  - But it is designed for statistical processing

R – data types

- R recognises a number of data types
  - Numeric
  - Logical
  - Character

  - Complex ($a + b\ i;\ i^2 = -1$)
  - Integer
  - Raw

R – data structures

- Last week, we looked only at *vectors*
- Vectors are similar to arrays in other programming languages
  - However, we start index numbers at 1, rather than 0
- 'Simple' values are vectors with one element

```
x <- 1
```

Recycling

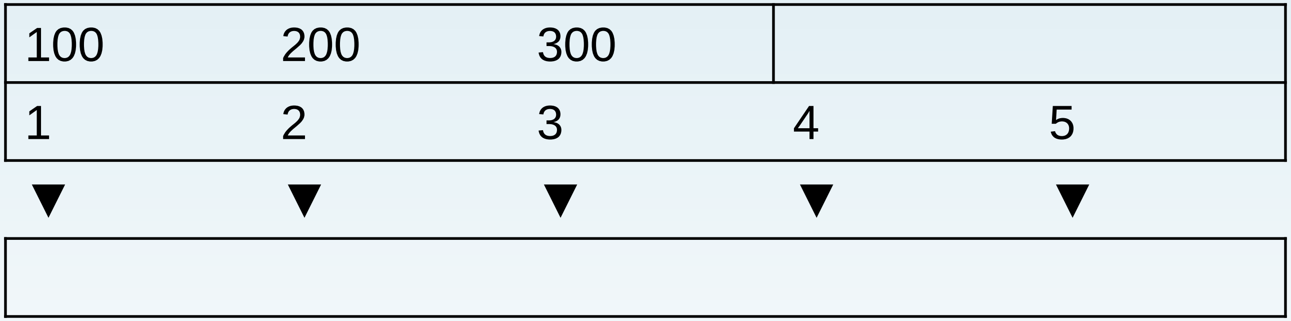
- Example from last week:
  - Sum of 1 + ½ + 1/3 + … + 1/n (note that 1=1/1)
- This requires the denominators 1,2,3,…,n
  - R can generate a sequence1:n easily

- Suppose n is 4
  - 1:4 is the vector 1,2,3,4
  - sum(1:4) gives us the sum of this vector (=10)
- sum(1/1:4) will give us an answer
  - Without knowing more, we might think it will give 1/(1+2+3+4)
  - But! R will try to make vectors in an operation the same size!
  - The numerator (1) will be recycled
  - sum(1/1 + ½ + 1/3 + … + 1/n)
- Similarly, sum(1 + 1:4) is calculated as 1(+1) + 2(+1) + 3(+1) + 4(+1) rather than 1+10

```
> myVec <- c(100,200,300)+c(1,2,3,4,5)+c(10,20,30,40,50,60,70)
Warning messages:
1: In c(100, 200, 300) + c(1, 2, 3, 4, 5) :
 longer object length is not a multiple of shorter object length
2: In c(100, 200, 300) + c(1, 2, 3, 4, 5) + c(10, 20, 30, 40, 50, 60, :
longer object length is not a multiple of shorter object length
> myVec
[1] 111 222 333 144 255 161 272
```

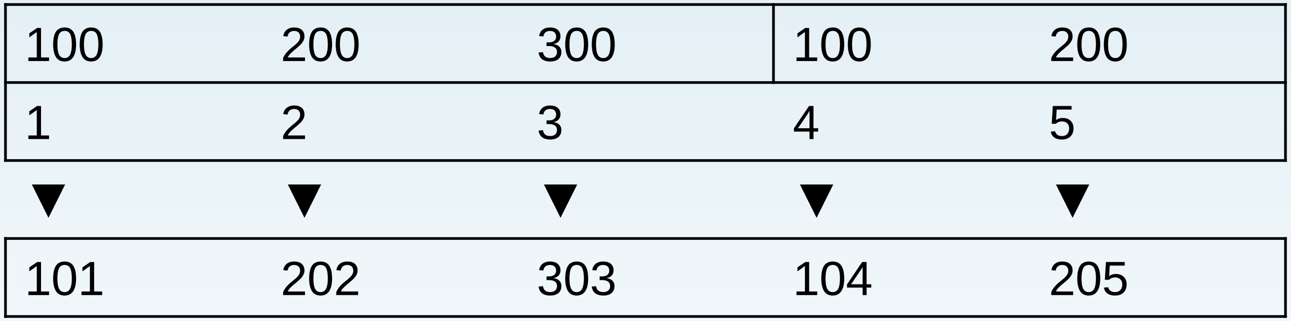
```
c(100,200,300)+c(1,2,3,4,5)+c(10,20,30,40,50,60,70)
```

```
c(100,200,300)+c(1,2,3,4,5)+c(10,20,30,40,50,60,70)
```



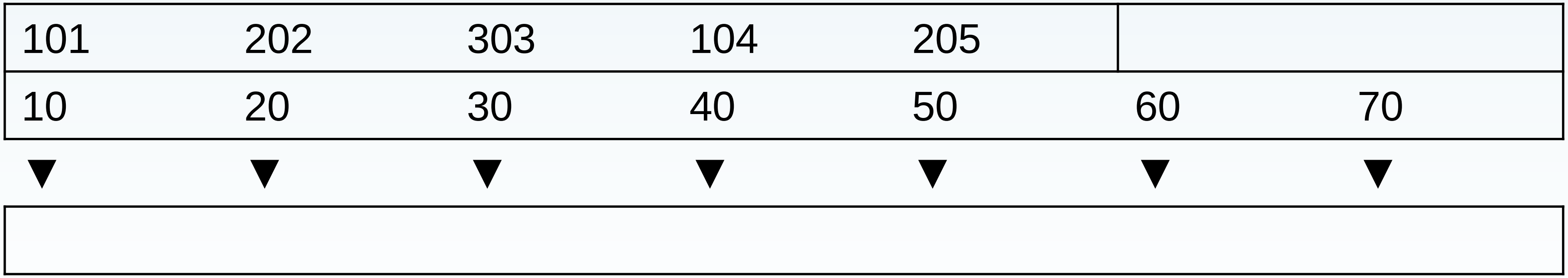
100	200	300		
1	2	3	4	5
▼	▼	▼	▼	▼

```
c(100,200,300)+c(1,2,3,4,5)+c(10,20,30,40,50,60,70)
```



100	200	300	100	200
1	2	3	4	5
▼	▼	▼	▼	▼
101	202	303	104	205

```
c(100,200,300)+c(1,2,3,4,5)+c(10,20,30,40,50,60,70)
```



101	202	303	104	205		
10	20	30	40	50	60	70
▼	▼	▼	▼	▼	▼	▼

```
c(100,200,300)+c(1,2,3,4,5)+c(10,20,30,40,50,60,70)
```

101	202	303	104	205	101	202
10	20	30	40	50	60	70
▼	▼	▼	▼	▼	▼	▼
101	202	303	104	205	161	272

R – data structures

- Lists
  - Each element can be different data type
  - New elements can be directly added
  - Elements can be named

R – data structures

- Matrices
  - A matrix is a two dimensional vector
  - Same properties as vectors
    - All entries are the same data type
    - Size is fixed

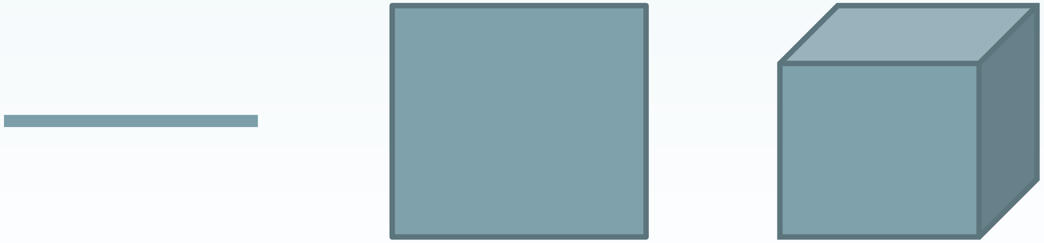
R – data structures

- Arrays
  - Arrays are multi-dimensional vectors

- Vectors, matrices, arrays
  - A vector is a 1 dimensional array
  - A matrix is a 2 dimensional array
  - An array is an n-dimensional space

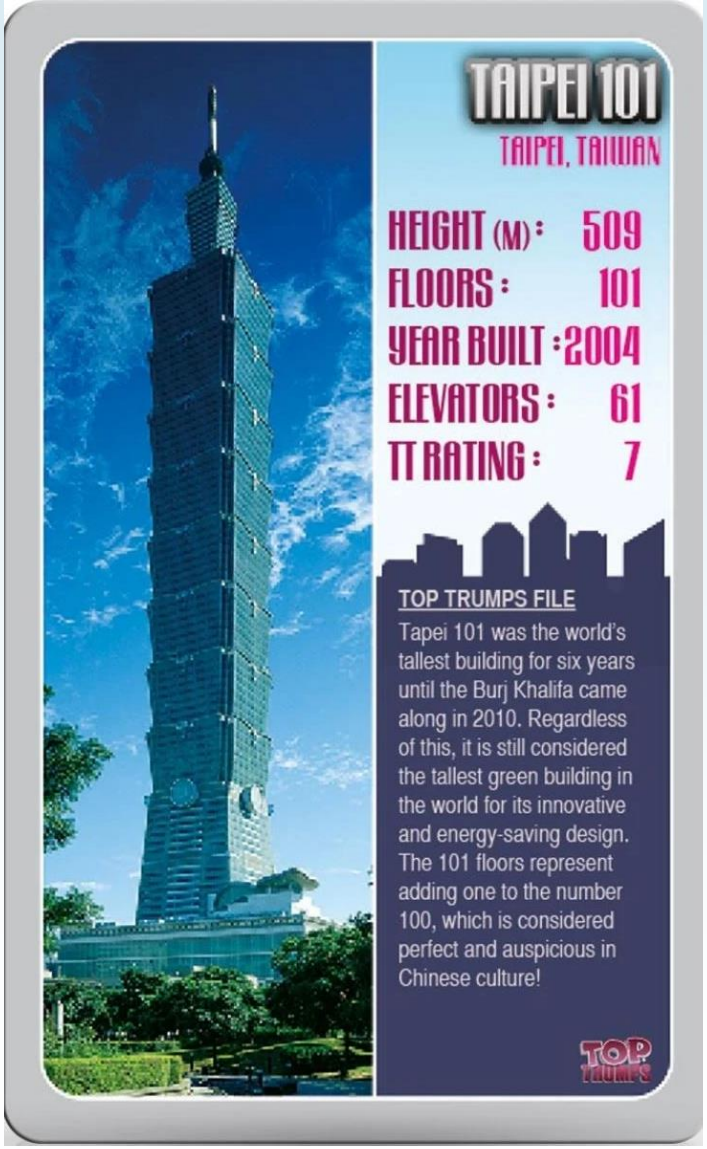
'n-dimensional space'

- It is easy for us to conceptualize 1, 2, and 3 dimensional objects
  - – We can mentally visualize these as physical dimensions



'n-dimensional space'

- When we talk about data, a dimension is simply one of the recorded characteristics
  - Entities can be described with any number of characteristics, whether numerical, categorical, continuous, discrete etc



TAIPEI 101

TAIPEI, TAIWAN

HEIGHT (M):	509
FLOORS:	101
YEAR BUILT:	2004
ELEVATORS:	61
TT RATING:	7

TOP TRUMPS FILE

Tapei 101 was the world's tallest building for six years until the Burj Khalifa came along in 2010. Regardless of this, it is still considered the tallest green building in the world for its innovative and energy-saving design. The 101 floors represent adding one to the number 100, which is considered perfect and auspicious in Chinese culture!

R – data structures

- Data frames
  - Data frames are used for typical tabular data
  - They consist of rows and columns, each column has a single data type
  - Data frames can be merged using similar principles to SQL joins

R – data structures

- We will focus on
  - Vectors
  - Matrices
  - Data frames

Matrices

- We can create a matrix using the *matrix* command

```
> myMat <- matrix(data=1:12, nrow=3, ncol=4, byrow=FALSE)
> myMat
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

Basic operations on matrices

- We can do mathematical operations on matrices if they are the same size
    - The operations +,-,/,* work on each cell

- Consider three matrices
- They all have 12 cells
- myMat1 and myMat3 are 3 rows by 4 cols
- myMat2 is 4 rows by 3 cols

```
> myMat1 <- matrix(1:12,3,4,FALSE)
> myMat2 <- matrix(13:24,4,3,FALSE)
> myMat3 <- matrix(25:36,3,4,FALSE)
> myMat1
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> myMat2
     [,1] [,2] [,3]
[1,]   13   17   21
[2,]   14   18   22
[3,]   15   19   23
[4,]   16   20   24
> myMat3
     [,1] [,2] [,3] [,4]
[1,]   25   28   31   34
[2,]   26   29   32   35
[3,]   27   30   33   36
```

- We can add or multiply (etc) myMat1 and myMat3
- We get an error message if we try to operate on myMat1 and myMat2

```
> myMat1 + myMat3
     [,1] [,2] [,3] [,4]
[1,]   26   32   38   44
[2,]   28   34   40   46
[3,]   30   36   42   48
> myMat1 * myMat3
     [,1] [,2] [,3] [,4]
[1,]   25  112  217  340
[2,]   52  145  256  385
[3,]   81  180  297  432
> myMat1 + myMat2
Error in myMat + myMat2 : non-conformable arrays
```

Subsets of matrices

- We can refer to individual cells
  - – myMat1[2,3]
    - This syntax is similar to array syntax in other languages
- We can refer to subsets
  - – myMat1[1:2,3:4]
  - – myMat1[1:2,]

```
> myMat1[1:2,3:4]
     [,1] [,2]
[1,]    7   10
[2,]    8   11
```