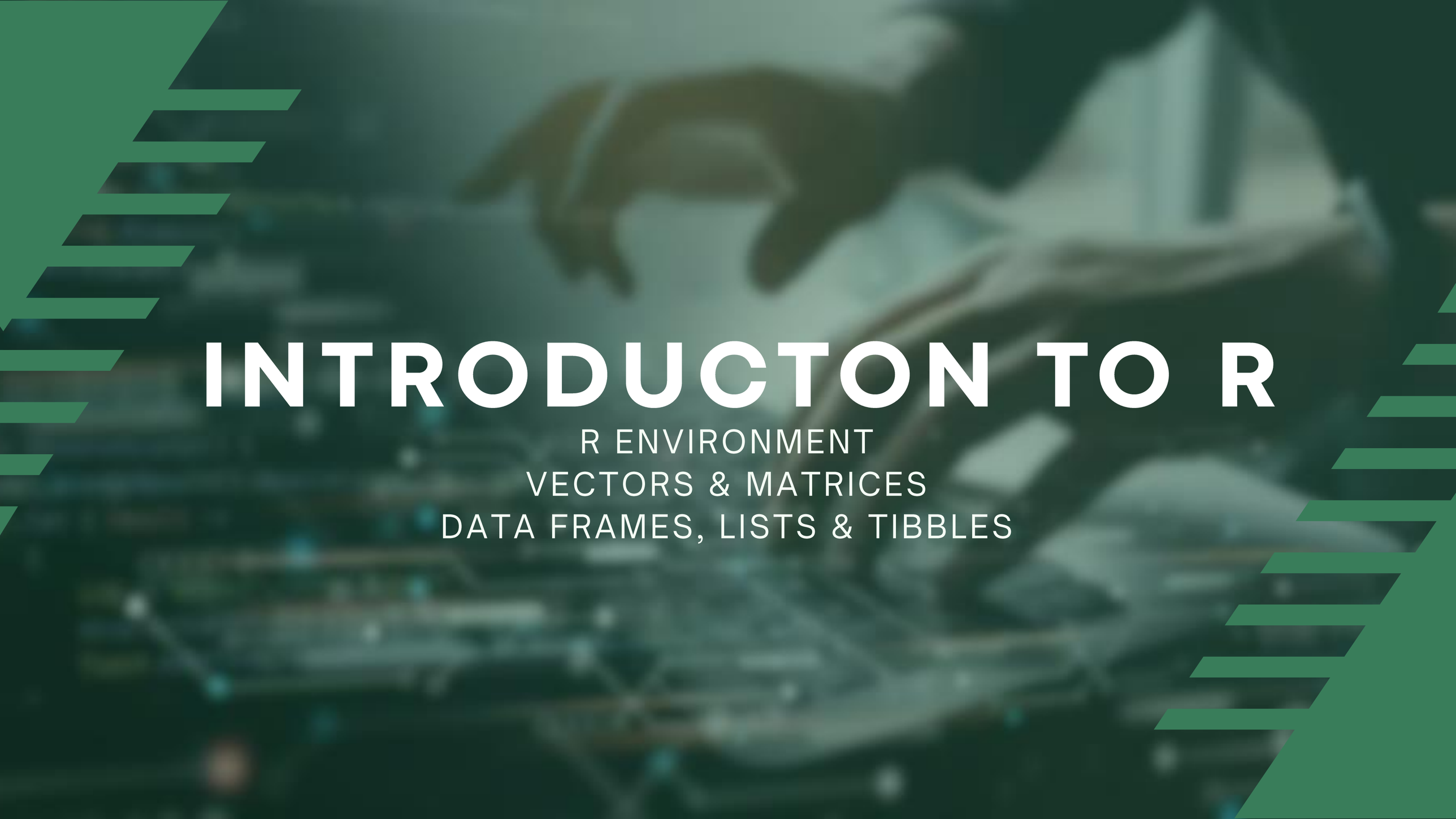
Universiti Kebangsaan Malaysia
The National University of Malaysia

NARATIF BAHARU UKM
UNIVERSITI WATAN KITA

FAKULTI SAINS & TEKNOLOGI

INTRODUCTION TO R

NORATIQAH MOHD ARIFF



INTRODUCTON TO R

R ENVIRONMENT

VECTORS & MATRICES

DATA FRAMES, LISTS & TIBBLES

What is R?

- An open-source statistical environment which uses programming syntax with many built-in statistical functions.
- It was designed by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand.

70%

Data Miners

- It’s the most-used data science language after SQL.
- R operates seamlessly on different platforms like Windows, macOS, and Linux, making it easily accessible to users regardless of the operating system they use.

INTRODUCTION TO R: Basic Things to Know

- Download/ Install R
  - Go to R webpage. You can google R.
  - Download R and install in your computer.
- Start R
  - Double click on R shortcut icon/ Go to **Start** → **Program** → **R**
- Base-R is the basic software which contains the R programming language.
- RStudio is a software that makes R programming easier.

INTRODUCTION TO R: Basic Things to Know

- Getting to know R
  - R console (the workspace)
  - script
  - variables (objects)
  - assignment operator (<-, =,->)
  - function
  - arguments
  - comment operator (#)
  - ( ) for functions, [ ] for vectors

INTRODUCTION TO R:
Basic Things to Know

- Quit R
  - `q( )` or **File → Exit.**
- Help in R
  - `help ( )` or type "`?`" in front of the name of the function you need help with
- `apropos( )` if you are not sure about the name of the function
- List all active variables in the workspace
  - ls ( )
- Remove variables
  - `rm ( )` to remove specific variable or `rm (list=ls( ))` to remove all variables
  - You can also go to **Misc → Remove all objects**

INTRODUCTION TO R: Basic Things to Know

- Set working directory
  - Use `setwd` to set your own working directory (where all your files will be save)
  - `setwd("C:/rclass/") OR setwd("C:\\rclass\\")`
- Get working directory
  - Use `getwd` to get the current working directory

INTRODUCTION TO R: Basic Things to Know

- Packages
  - Install packages manually by first downloading the zip file from the CRAN webpage, then **Packages → Install package(s) from local zip file**.
  - Use CRAN directly in R, **Packages → Set CRAN mirror**, then **Packages → Install package(s)**.
  - `update.packages()` offers to download and install packages which have a (suitable) later version.

INTRODUCTION TO R: Basic Things to Know

- Save your work
  - Save workspace, **File** → **Save workspace** (save everything)
  - Save script, click on active script, then go to **File** → **Save as** (save edited commands and comments)
  - Save history, **File** → **Save history** (save all commands)
  - Copy and paste anything from R to any file suitable, text file, Microsoft word, Excel and so on.
- Load previous work
  - **Load workspace**, **Open script**, **Load history** from **File**.

INTRODUCTION TO R:
Class of Objects

- Numeric: Decimal values are called “numerics” in R. It is the default computational data type.
- Integer: Integer numbers
- Character: A character object is used to represent string values.
- Logical: A list of TRUE/FALSE.
- Vector: A list of items that are of the same type.
- Matrix: A two-dimensional data set with columns and rows.
- Complex: A number with imaginary value i.

INTRODUCTION TO R: Class of Objects

- The `class()` function check the class of an object.
- You can also check specific class using:
  - `is.numeric(), is.integer(), is.character(), is.logical (), is.vector(), is.matrix(), is.complex()`
- You can coerce a variable into a specific class using:
  - `as.numeric(), as.integer(), as.character(), as.logical (), as.vector(), as.matrix(), as.complex()`

INTRODUCTION TO R: Arithmetic Operations

- R includes the usual arithmetic operations such as +, -, *, / and ^.
- These operators have the standard precedence, with exponentiation (^) highest and addition/ subtraction (+, -) lowest. As usual, the order of these precedence can be control by the use of parentheses ( ).

INTRODUCTION TO R:
Common Math Functions

Functions	Details
`sqrt()`	Square root
`abs()`	Absolute value
`sin(), cos(), tan()`	Trigonometric functions (in radians)
`pi`	The value for π = 3.1415926
`exp(), log()`	Exponential and natural logarithm functions
`log10()`	Logarithm with base 10
`factorial()`	Factorial function (!)
`Choose()`	Combination function

INTRODUCTION TO R: Logical Operators

- R also makes use of common logical operators

Operators	Details
==	Equal to
!=	Not equal to
<, <=	Less than, Less than or equal to
>, >=	Greater than, Greater than or equal to
&	Logical AND
|	Logical OR

INTRODUCTION TO R: Vectors: Use `c()` Function

- The function `c()`, known as the catenate or concatenate function, can be used to create vectors from scalar or other vectors.
- Examples:
  - `x<-c(1,3,5,7)` #create vector x from single numbers/ scalars
  - `y<-c(2,4,6,8)` #create vector y from single numbers/ scalars
  - `z<-c(x,y)` #create vector z from vector x and vector y

INTRODUCTION TO R: Vectors: Use ":" Operator and `seq()` Function

- The colon operator can be used to generate sequence of numbers which increase or decrease by 1.
- Examples:
  - `1:10` #increasing sequence
  - `10:1` #decreasing sequence
- The function `seq( )` can be used to sequence of numbers starting and stopping at specified values with increments defined by user.
- This function contains 3 arguments, the first argument is the starting point, the second argument is the stopping point and the last argument is the increment.
- Examples:
  - `seq(0,1,0.1)` #values between 0 and 1 with an increment of 0.1
  - `seq(20,5,-4)` #values between 5 and 20 with an increment of -4 (The sequence is in decreasing order)

INTRODUCTION TO R: Vectors: Use `rep()` Function

- The function `rep( )` stands for repeat or replicate.
- The function contains 2 arguments, the first argument is a value or a vector, the second argument is the number of times the value or each element of the vector (in the first argument) is replicated.
- Examples:
  - `rep(3,5)` #repeat the number '3' five times
  - `rep(c(1,3,6,9),3)` #repeat the vector 3 times
  - `rep(1:3, c(2,2,2))` #repeat each element of sequence 2 times

INTRODUCTION TO R:
Vectors: Arithmetic

- The same functions used for scalars can be used on vectors. However, the length of the vectors must be carefully observed, or some vector arithmetic will result in errors.
- The square brackets [ ] are used to identify specific elements in vectors.
- Some useful functions for vectors are as follows:

Functions	Details
`length()`	The size of the vector
`sum()`	Calculates the total of all values in the vector
`prod()`	Calculates the product of all values in the vector
`cumsum(),cumprod()`	Calculates the cumulative sums or products in the vector

INTRODUCTION TO R: Vectors: Index

- Use to retrieve members of a vector
- Example:
  - `x<-c(3,2,5,7,1)`
  - `> x[3]`

    `[1] 5`
  - `> x[-2]`

    `[1] 3 5 7 1`
  - `> x[c(1,4,2)]`

    `[1] 3 7 2`
  - `> x[x<=3]`

    `[1] 3 2 1`

INTRODUCTION TO R: Matrices: Use `matrix()` Function

- Single values are written in the form of a vector and the function `matrix()` is used on the vector to create a matrix.
- The function has 4 arguments, the first argument is the vector, the second argument `nrow` is the number of rows, the third argument `ncol` is the number of columns and the last argument `byrow` is a logical operator where `TRUE` indicates the values in the vector is fill in by row or otherwise (by default the values will be fill in by column).
- Examples:
  - `y<-c(3,6,14,90,54,2,8,65,28,45)` #create vector y
  - `Y<-matrix(y,nrow=5,ncol=2)` #fill in by column
  - `Y<-matrix(y,nrow=2)` #fill in by column
  - `Y<-matrix(y,nrow=5,ncol=2,byrow=TRUE)` #fill in by row

INTRODUCTION TO R: Matrices: Use `cbind()` or `rbind()` Function

- Use the function `cbind()` or `rbind()`.
- These two functions stand for column bind and row bind to combine vectors and matrices in order to form a new matrix.
- Examples:
  - `x<-c(56,8)` #vector x
  - `y<-c(2,16)` #vector y
  - `A<-rbind(x,y)` #combine x and y as rows
  - `B<-cbind(x,y)` #combine x and y as columns

INTRODUCTION TO R: Matrices: Operations

- The same arithmetic and operations used for scalars and vectors can be used on matrices. It will apply itself on each element of the matrix.
- The dimension of the matrices must be carefully observed, or some matrix operations will result in errors.
- The square brackets [ ] are used to identify specific elements in matrices.
- Some useful functions for matrices are as follows:

Functions	Details
`dim()`	Dimension of the matrix
`%*%`	Matrix multiplication
`t()`	Matrix transpose
`det()`	Determinant of a square matrix
`solve()`	Matrix inverse

INTRODUCTION TO R: Matrices: Index

- Use to retrieve members of a matrix
- Example:
  - `x<-c(3,2,5,7,1)`
  - `y<-c(6,7,2,1,4)`
  - `z<-rbind(x,y)`
  - `z`

```
  [,1] [,2] [,3] [,4] [,5]
x    3    2    5    7    1
y    6    7    2    1    4
```

  - `z[1,]`

```
[1] 3 2 5 7 1
```

  - `z[,c(3,1)]`

```
  [,1] [,2]
x    5    3
y    2    6
```

  - `z[2,4]`

```
y
1
```

INTRODUCTION TO R: Vectors and Matrices: Use `scan()` Function

- When data to enter is too long.
- We should use the `scan()` function.
- Most appropriate when all data is of the same mode.
- By default, it expects all of its input to be numeric data, this can be overridden with the `what=` argument.
- Example:
  - `myscan<-scan()`
  - `myscan<-scan(what="")`
  - `myscan<-matrix(scan(),ncol=3,byrow=TRUE)`

INTRODUCTION TO R: Data Frames: Use `data.frame()` Function

- A data frame is a table or a two-dimensional array-like structure in which each column contains values of one variable and each row contains one set of values from each column.
- A data frame is a matrix-like structure, where the columns can be of different types.
- `data.frame()` function is similar to `matrix()` and `cbind()` function.
- Variables in a data frame are accessed using either the "`$`" sign or the `attach()` function.

INTRODUCTION TO R: Data Frames: Use `data.frame()` Function

- Use `str()` function to get a detail summary of the data.
- Example:
    - `x<-seq(1,4,0.5)`
    - `y<-c(1:3,rep(c(5,6),2))`
    - `dat<-data.frame(x,y)`
    - `str(dat)`

INTRODUCTION TO R: Data Frame: Entering Data Interactively

- Create a blank data frame for a variable
- `myx<-data.frame()`
- `fix(myx)`
- We can also key in the data into the blank data frame or edit an existing data frame by using the following method:

  Edit → Data Editor → type variable name

INTRODUCTION TO R:
Lists: Use `list()` Function

- Lists are the R objects which contain elements of different types like – numbers, strings, vectors and another list inside it.
- A list can also contain a matrix or a function as its elements.
- A list combines data of different class and length.
- The function `alist()` is an easy means of creating a list with empty elements.
- Example:
    - `a<-UKDriverDeaths`
    - `b<-USAccDeaths`
    - `c<-fdeaths`
    - `Deaths<-list(a,b,c)`

INTRODUCTION TO R: Assigning Names

- Use the `names()` function to assign names to vectors, variables in a data frame and variables in a list.
- Example:
  - ```
    x<-c(1,2,3,4,5)
        names(x)<-c("one","two","three","four","five")
    ```
  - ```
    dat<-data.frame(x,y)
        names(dat)<-c("myX","myY")
    ```
  - ```
    names(Deaths)<-c("List1","List2","List3")
    ```
- Assign names to each row in a matrix using `rownames()` and to each column in a matrix using colnames()
- Example:
  - ```
    rownames(z)<-c("row1","row2")
    ```
  - ```
    colnames(z)<-c("col1","col2","col3","col4","col5")
    ```
    ```

INTRODUCTION TO R: Tibbles

- A `tibble` is a modern reimagining of the `data.frame`, keeping what time has proven to be effective and throwing out what is not.
- It is in the `tidyverse` environment.
- It shows only the first ten rows with all the columns that can fit on the screen.
- Each column also shows the data types.

THANK YOU