Advanced Financial Instruments for Sustainable Business and Decentralized Markets

Prof. Abhinava Tripathi

Department of Management Sciences

Indian Institute of Technology-Kanpur

Week 1

Lecture – 2: **Fundamentals of R**

In this lesson, we will learn about the Fundamentals of R. In this video, we will discuss what R programming is and it is uses? We learn how to download and install R and RStudio. Then we will families ourselves with the interface of RStudio. We will understand basic working etiquettes with R, such as creation of working directory, setting up and changing the working directory, creation of new projects and working with projects.

We will also learn installation of packages and adding them to the current working directory. Next, we will talk about working with inbuilt functions in R, we will also discuss about arithmetic and logical operators. Lastly, we will discuss working with different data types in R, such as vectors, we will also understand, the construction of user defined functions. The course will help set up a building block for learners and act as a platform to implement more complex tasks in advanced topics using R. **(Video Ends: 00:01:03)**

(Refer Slide Time: 00:01:04)

In this video, we will discuss what R programming is and it is uses? **(Video Starts: 00:01:09)** We learn how to download and install R and RStudio? Then we will families ourselves with the interface of RStudio. So, we will start with introduction of R and how to download and install it. R is an independent open source tool and offers free implementation, it is an extension of the S language, it has huge computing power, it is an essential tool when it comes to data science and research.

A key advantage of R system is that it is free, it is easily downloadable and installed and as we will describe it shortly it is easy to use. R is a statistical computing environment that includes an interpreter for the R programming language with which the user programmer can interact in a conversational manner. R is one of the several programming environments used to develop statistical applications others include Gauss, Julia, Python, Stata, etc.

The best way to obtain R is by downloading it from the internet by simply looking for R project and as we will see by just simply clicking on the relevant web page to download. It will mention all the country's name and their corresponding crime packages. We can check the link for India as well, on this link provided we can download R, we can download the latest version available.

R runs under all the commonly available computer operating systems like windows, macOS, Linux and Unix and pre-compiled binary distributions of R are available for these systems. Once we have done R installation, we will install RStudio, the link is shown here to download RStudio. RStudio is an interactive development environment (IDE) for R, like R, RStudio is

open source software freely available on the internet for Windows, macOS, Linux and Unix systems.

Regardless of R computing system, the user experience in RStudio is nearly identical, RStudio has excellent facilities for writing R programs and packages. But more to the point for us it provides a convenient environment to conduct statistical data analysis. We can select RStudio and select the appropriate installer which is appropriate for R operating system and we can then start working with R.

Let us have a look at these sources where we can download R, RStudio and also there is a facility to directly work on R without installing it by clicking on R cloud. **(Video Ends: 00:03:38)**

(Refer Slide Time: 00:03:39)

In this video, we will explore CRAN. **(Video Starts: 00:03:44)** Kindly have a look at this web page and this link https://cran.r-project.org/. On this web page you have all the relevant information related to R is available for example you can see, what is the latest news? What are the latest updates? What are the latest releases? Let us say you want to download the latest version of R for your operating system.

Maybe Linux, macOS or Windows you can click here on one of these links. Let us say if I have Windows, I can click on this download R with Windows. Here I will find the latest R for Windows system available, I can click on it and download it for the first time. Also, there are

lot of other functionalities for example, if you want to download or have a look at the R journal, you can click on this R journal button.

And you will be taken to this page where all the new updates, new publications about R latest packages and everything will be available on this page. Similarly, if you want to see what are the latest happenings in R? What is the new news or those kind of thing? You can click on what's new and you will be taken to this page. All the latest announcements related to R and other things will be available.

Similarly, there are a lot of other manuals, FAQs and other information for example, if you want to specifically search something related to R, you can click on this search R-Search and you will be taken to this page, where you can search something specific to R. Also, once you install download install R you also need to install RStudio. So, you need to go to this webpage rstudio.com and here all the latest RStudio versions will be there.

You will be given the basic functionalities with free version and also there are some advanced versions. And you will be given a comparative view with these and along with the free version in my experience, the free version provides all the capabilities and computing power. So, you can download this free version here or you can download it directly from here download RStudio for Windows.

If you want to if you have some other operating system, you can download it for macOS or Ubuntu or other versions, the latest version is available here, you can download and install it, it is quite easy. Once you download R and RStudio, you are good to go, although some of us may be using new device or may sometimes may be working where it is not installed those kind of devices.

So, there also you have the option to go to this page RStudio cloud. RStudio cloud you can directly work you need to make a free account. So, once you make your free account RStudio cloud this kind of web page will open and there you can directly start working. Although some limitations are there in terms of space and capacity and speed there are some limitations.

But still, it will give you exactly the same interface and something similar in terms of basic computing power, it will offer you lot of similarities with the actual installed R. So, with this

there is a lot of flexibility and freedom working with R. Whether you have installed on your device or you have not installed but you are working with R cloud as well. **(Video Ends: 00:06:30)**

(Refer Slide Time: 00:06:31)

Let us get familiarized with the interface of RStudio. **(Video Starts: 00:06:35)** When you start RStudio, you will find a window like this, an interface which is basically a RStudio interface, the RStudio interface shown here is quite customizable. Right now, you can see four panes here and as you become familiar with it, you can very well decide to reconfigure these panes.

Basically, the RStudio visual as per your preferences for example, you can move these tabs or different panes and change the placement of these panes to your liking. RStudio has four main panes as you can see right now, each in a quadrant of R screen. This one is called source editor, lower one is called console then this one is your history or workspace browser and this one contains the plots, help, files etc.

So now, we will have an overview of all these four quadrants in brief, we will start with this scriptwriter which is the top left pane. The source pane is where you create and edit R scripts, our collections of code. You will notice that when you are typing code in a script here on the source panel for example, codes like this or codes like this R will not actually evaluate the code as we type.

To have R actually evaluate our code, you need to first send the code to console, we can see in the next figure for our reference to create and save an R script in the studio you can go to this

file button. Here you can do a lot of operations with for the script and other operations for example, you can save by clicking here you can save R script. You can open new R script, you can open existing R scripts and so on.

Once a script or a file has opened you can save the file by going to file save, so that can be done. You can keep the same name or you can change the name, once you save automatically the extension dot R which is basic syntax for R files that is automatically added. If you have put a specific code for example this 5*3 to run this specific code, you can click on control enter or you can go to this particular run point and click on this wherever the cursor is this that line will be run.

You could also run a set of lines for example, you can select the set of lines and then either click control enter or click on this run button and run the series of lines of codes so, this is about R script. Next, you also have something called console window which is here on the left button, the console is the heart of R. This is the place where R actually evaluates the code.

At the beginning of the console, you can see a character this is basically a prompt this prompt tells us that R is ready for new code. We can type the code directly into the console after the prompt and get an immediate response for example, if I type 1 + 1 here into the console and press enter, notice an output is produced. You will see that R immediately has given us an output of 2.

But generally, it is preferred to write the code on the script and then run it and later you would like to save it as well. So that is why it is better to write code on script, here you can for practice you can write directly and run but you will not be able to see. If you want to clean this console, you can click on this broom button or you can click or run control + L. So, if you click on control +L or if you click on this broom button all that is there in this console will be clean.

Third is history or environment panel this one on the right of top, the environment tab of this panel shows us the name of all the data objects that are right now there. So, if there are any data objects, let us say I create a data object a = 3, it will be put in this environment tab, so that means it is added to my current working environment. So, all the vectors, matrices and data frames and other data objects that you have defined in the current session they will be appearing here.

We can also see information like the number of observations and rows in a data object, that also is visible here. The tab also has a few clip cable options like import data set so, you can click on this import data set option and different data types you can import by clicking on these from text, excel and other software's like SPSS, SAS and stata. Also, once you click on this import data set option, it will open a graphical user interface for importing data into R as we have said.

So, you can click and input data for example, if I click on text base and then I can click on for example this data file. So, files like CSV file or text file can be read, I can click on this open button to read this. Once I click on open button an overview of this data will be provided some options to change the format. For example, header, footer whether I want to read it as a comma separated file or a period as decimal, comma these kind of options and it will give me a default picture also.

And I can change it by clicking on some of these options and to my liking if the input data is to my convenient, I can click on import and it will be imported. And also, a brief overview, the way data has been read that is also shown to be here, so this is how the environment tab works. Here also you can see the history tab, all the commands that have been run till now they are visible here, I can see them in the past those command that are run.

If you want to clean this, you can click on this clean button but be sure that all the everything that is needed is has been used. And you do not need the objects here then only you click on this and everything will be removed from the current environment. But you need to be very sure that you do not need them further. You can also click on tutorial button to see some help and tutorials that are easily available on main R page.

Now, we will go to the multi-purpose pane which is on the bottom right, here you can see file, plots, packages these buttons and also the help panel. So, there is lot of helpful information is also there. Let us go through these tabs in detail, first you have files tab, the files panel gives you access to the files directly on your hard drive there are number of nice features on the files panel.

For example, you can use it to set your working directory, you can set a working directory here or once you navigate to a folder you want to read a file and save files, you can click more and then save as working directly. For example, if I want to read this Davis. rds file I can click on it, and it will give me an option that should be read it if you click on ok it will be read and the name assigned to as Davis.

Similarly, there are other files that you can read also here you can set working directly as well. In fact, you can set the working directory here by clicking on session button and then set working directory button. Here you can see choose directory, once you click on choose directory some window will be opened. If you want to set this particular folder as your working directory you can click on open or you can change here by going to some different folder.

Once you click on open button notice a setwd command has appeared, it is indicating where the current working directory has been set. And in that particular directory what are the datas, objects and other things that are appearing here on my file tab. Then look at the plot window, all the plots that you have currently plotted will be appearing including some history of plots as well there are buttons for opening the plot.

So, once you have made the plot there will be buttons for opening the plot in separate window a larger window or you can export it in the form of pdf, you can export it in the form of jpeg. You can change the site, you can do a lot of permutations and combinations with your plot to make it more aesthetic on this plot window as we will see short then you have a packages tab.

Here is the list of all the available R packages that are installed on your current drive and indicates whether or not they are currently loaded. In this section the package is unchecked they are not installed if it is checked those are installed but maybe not loaded. The check package the packages that are checked they are both installed as well as loaded. If they are not checked that means they are just installed but not loaded into your current working directory.

So, either you can just click on check like this and load them on current working directory or what you can do is? You can use something called library command and put the name. Let us say if I want to put a car package it will provide me a drop down menu, I can select the car package and I can run with control enter command. Notice, on the console window the car package has been loaded now into my current working directory.

So, it is loaded now and I can start using it. Those that are not checked those packages are though installed but not loaded in my current environment and I may not be able to use their functionalities. Last and another very important window is help window here, on this help menu you have for different R functions you can directly type here for help. For example, anything maybe regression topic, any topic you will type relevant help will appear on this particular panel.

There is another way if you want to find help you can put question mark and let us say if you want to look at something about regression you can directly type question mark and control enter. So, the if you have put the relevant and correct keyword it will start appearing here. For example, I want to know there is some object called lm, I put control enter all the information about that lm fitting linear models is appearing here.

So, either I can put on this space or I can put the question mark to access the help window. To summarize in this video, we have tried to install and set up R and RStudio in our system. Also we have understood a brief understanding of all the quadrants all the four panes in the studio and we will start doing basic coding in subsequent sessions. **(Video Ends: 00:16:05)**

(Refer Slide Time: 00:16:06)

In the previous video, we have discussed how to install R and RStudio and I familiarized ourselves with the interface of R. In this video, we will understand how to create a working directory and create projects in R. **(Video Starts: 00:16:20)** We will also discuss installing and loading the packages and their uses in R, first we will start with working directory in R. The

working directory in R is the folder where we are working correctly it is a folder where R leads and saves files.

If you want to change R working directory or set a new working directory it can be done in the following steps. First, we will go to this session command, session button click on set working directory then choose working directory and then you can change the current folder or if you want to have a particular suitable location. For example, I want to create my working directory here inside introduction.

And in this particular folder I want to create working directory I can do so. For example, I will set this working directory I will click open button notice, a setwd command has appeared you can very well copy paste this command into your currents file or you can leave it as it is. Once you have set your working directly notice this file tab here the current location is exactly the same that you have set as working directory.

All the files data and everything that is there in that working directory is appearing here now. For example, you can directly load data from here or read data from here by clicking one of these data and code files. So now, we had just through this step set R working directory where we wanted to. It is also possible to do this directly using the functions setwd which stands for set working directory.

For windows, the command would looks something like this let me copy paste it for us. So, notice the syntax of this command you have the setwd command and the location or the address where we wanted to set R working directory. Though it is rather lengthy, you can have a small working directory location also where the address is not so lengthy. Setting a working directory is the best practice to help us store the coding data in a particular directory and remove the risk of remembering the directory location.

So, even after six to nine months later when you are working on this file, you need not remember where was the location where you saved the data and you were working? If you have copy pasted this code in that script itself. So, if you copy paste this setwd line along with the location you need not remember the exact location. Now, everything that you write in the code will automatically be saved in that particular directory.

Once you have set the working directory appropriately. And to know that it has actually happened there is a file tab as we have seen here in the fourth window. The multipurpose pane which shows the address of your current working directory where all the files under the working directory will appear. We can directly open the files from here by clicking on them on a particular file and click on import.

For example, here I can click on the data files and import them now, if you want to know your current R working directory just type the command getwd. So, if I type the command getwd it will produce the current working directory this getwd stands for get working directory. If you want to set a default working directory, a default working directory is a folder where RStudio goes every time you open it you can change it.

You can change the default working directory from the RStudio menu, you can go to the tools global options and then you can click on browse and set your permanent working directory wherever you want to. So, it will be the default directory every time you open R. Also, you can close your current R, RStudio session every time you close R, RStudio you will be asked whether you want to save the data.

So, if you click on this close button, you will be asked to save your data from R if you want to save you can click on save. You can check or uncheck whichever script files or working space files you want to save you can click on save select. If you uncheck that that file will not be saved and the older version will be only remaining. If you want to update it you rather put save selected and check mark on all these files.

So, this is how you close your R and current working RStudio session. Next another very important entity is projects in RStudio. So, when you are working in projects you want to save it open old projects. Projects is a collection of script, data, intermediate, outputs and so on. So, let us say in my current working directory there is a project called macro, there are two ways to load this project either you directly click on it.

Once you click on it, it will ask you, whether you want to load it in your current working environment? If you click on yes a load command will run it will put all these codes, data everything in that particular project on your current environment or you can do it throw

manually by coding. You can simply run load and give the name of the project which is macro.RData and it will load it automatically.

load("macro.RData")

So, these are two ways to load the project if you want to open it you can directly open new project or open existing project in this session or new session that way you can open it, you can also save the projects as well. So, if you want to save a project you can save the project as well. Projects in RStudio in R are means of organizing your work so that both you and your RStudio can keep track of all the files relevant to a particular activity.

For example, a student may want to use R in several different projects such as statistics course, a sociology course or a thesis research project, Using RStudio projects, to organize your work will keep the files for distinct activities separate from each other. To create a project, you can click on this file and then you project to start a new project from RStudio menu. In the resulting sequence of dialog boxes successively selecting existing directory or new directory depending upon whether or not the directory already exists. **(Video Ends: 00:22:17)**

(Refer Slide Time: 00:22:18)

Let us understand about creating projects and working with packages in R. **(Video Starts: 00:22:23)** Using RStudio projects to organize your work will keep the files for distinct activities separate from each other. It is always advisable to save your work in the form of projects where you can save all the data files, codes and other intermediate outputs. So, let us see how to create new projects?

Go to file click on new project now let us say you want to create a new directory or you can also open in existing directory if there is an existing project. Let us say you want to create a new project in new directory, you give it a name let us say we give it a name called new p, new project we create this project this is our new project that we have just started. In this project let us say we have some code files.

Let us say we run some codes these are some code files maybe from some other places I have this code file. I also have some output, let us say I click on this macro data to load this on my current environment. I have now this macro data loaded which I clicked here by clicking here I loaded this macro data on my environment. I also clicked on some of the codes like session one and opened the code.

Now, let us say I want to save them all together the data, the code file and other things in the form of a project. For that what I need to do is? I need to click on this new project button and now I can close this project it will ask me to save the project, I will save it, once I save this project the project is saved in the directory called new project. In future if you want to open this you can go to file you can click on open project or you can directly check on recent projects where you have saved this file in the directory new project.

Once you click on that all the codes, data files everything will be loaded exactly in the manner when you are working and you can use this in the manner you want. So, this is how you create new projects, save new projects, inside new working directories or existing working directories as well. Now, we will come to the packages, packages is also a very important aspect of R codes. R comes with a lot of extensive capabilities that are right out of the box.

Some of these most exciting features are available as optional modules that we can download and install. There are more than 5,500 user contributed modules called packages that we can download from the CRAN website that we saw earlier. We will use many of these optional packages in this session, what are these packages? Packages are collections of R functions, data and compile code in a well-defined format.

The directory where packages are stored on R computers called the library, R comes with a standard set of packages, the base packages or data sets utilities, graphics. And methods these base packages provide a wide range of functions and data sets that are available by default.

Other packages are available for download installation, once installed they must be loaded into the current session in order to be used.

You can use the command search to tell you which packages are loaded and ready to use, to install a package for the first time you go to this tool button then click on install.packages, window will appear here you can click the name of the relevant package. For example, if you want to name the package let us say you want to download package car, you can select this package a drop from the drop down menu and then click on install.

Once you click on install, the relevant package will be downloaded, notice the console window install.packages command is appearing. Now, the car packages once this is installed, car package will appear in your list of packages, I can search that very easily car once I search that you notice the car package is there but still it is unchecked. The full form and all the details are available about this package here it is called companion to apply it regression, the car package.

If you want to add it to your current working directory, you can check it once you check on it, it will be added to your current working library. Notice that in the console window there is a library car command appearing, you can yourself straight away when you are working and if you have already installed this car package. You can click on the library command and just click car.

And straight away running this library car will also have the same effect of adding this car package to your current work directory. So, not only one you can install more than one package for example, you can go to the install package and add many packages. For example, you can add let us say dplyr, you can add lubridate, maybe stringer. So, in this manner you can keep on adding more and more packages, all these packages will be appearing, you can pack zoo, and so on.

Once I click on this and I click on install, notice a command with all these packages notice install. packages all these packages together it will be installing in my current path, all these four packages are installed. In future if I want to install in some other system, I can straight away copy paste this command in my RStudio window, script window and run it to have the same effect install all these packages.

Once you install this you can add them to your current working library by putting in the library command maybe dplyr, run this command and it will put the library update the library with dplyr. We can also similarly add lubridate, all the packages that you have installed that are not part of base R package they will be added to your current library**.** So, in this manner you can add many, many libraries that you need.

So, in this video, to summarize we have learned how to set a working directory, create projects on RStudio? We have also installed a number of packages and seen how to install the packages that we need for subsequent sessions for data analysis. And also, we have added some of these packages to R current working library. **(Video Ends: 00:28:09)**

(Refer Slide Time: 00:28:09)

Now, we will start with basic coding, let us start with basic operators in R. **(Video Starts: 00:28:14)** An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. In this video, we will discuss several types of operators in R programming language and their usage. We will start with arithmetic operators, arithmetic operations in R include addition, subtraction, multiplication and division.

As these are binary operations they need two operands they are simple and similar to operations in any other programming language. The following illustration tells about arithmetic operation between two variables in R. So, we will start with arithmetic operators, a very simple operation would be an assignment operation where I assign “a” value of 3 to a and I press ctrl + enter.

If I want to see what is inside a, I can simply press a and ctrl + enter to get the output, and the value inside “a” which is 3 is printed. Similarly, I can create a new variable which is “b” and assign a value of 5 by this assignment operator that is equal to, I can press ctrl + enter and generate the output like this. I can press type “b”, press ctrl + enter, and generate the output this is a simple assignment operation.

```
a+b
b-a
a*b
a/b
a^b
```

Similarly, I can perform addition as well, I can press a + b, control enter and I get the summation that is 8 so, this is an addition operation. Similarly, I can perform subtraction operations to b – a which is 2, multiplication 15, or division as well. All these symbols mathematical symbols are quite common to our daily usage so, we are not new to these. Also, if you want to create an exponential operator that is a to the power b.

You can do that, let us say 2^3 or 4^4 and very easily the output is produced. You can also put more complex operations like 4^3-3^2, these are common arithmetic operators. In addition to the common arithmetic operators, the package in the standard R distribution includes hundreds of function that are programmed for mathematical operations for manipulating data, statistical data analysis, making graphs, working with files, and other purposes.

Functional arguments are values passed to functions and these are specified within parentheses after the function name. Here is a quick review of the logarithm function which plays a key role in statistical data analysis. The basic way of doing logarithm in R is with the log function in the format of the log then the value, maybe 100 then the base at which we want, this returns the logarithm of the value at the given base which is to in this case.

```
log(100,10)
```

By default, this function produces a natural logarithm of value. To obtain information about the function, we can press the help function or directly press the question mark with the log (?log), will get help here, we can directly use this help tab here to obtain help. Now, if you

want to compute the log another way to do that is to use this, help log and you will get the help available in the log.

Now, let us say we want to see what is log function? We can press log and with tab completion all the similar functions with log format or log syntax will appear. Let us look at the first one which is log along with the syntax, the tab completion gives us the entire syntax we need to just press tab and all the syntax will appear in front of us. It shows that we can type x which is the value we can also specify the base.

If we do not specify the base, the default base is exp bracket 1 which is exponential or natural base. So, let us say we start with the log of 9, at a base of 3, we get a value of 2 this is the basic logarithm function with 9 as the value and 3 as the base, the results are 2 because 9 is the square of 3. Similarly, I can type log and 5 which will give me the value of log 5 at a natural base which is 1.609.

I can also compute similarly multiple values like log of 100 at the base of 10 or another way to do this would be log10(100) and I will get the logarithm of 100 at a base of 10. So, here we have a comparison of the base 10 logarithms of 100 obtained by the basic logarithm function and it is shortcut, for both cases the answer is 2 because 100 is 10 squared. Next, we have logical operators, logical operators are used to carry out Boolean operations like AND, OR etc.

Let us look at operators and their descriptions first then we will proceed with their implementation in R. A simple ampersand this ampersand symbol (&) is logical AND operation then this pipe operator (|) this is called pipe symbol this is logical OR. So, operators & and | perform element wise operations producing results having length of the longer operand and double ampersand (&&) and double piper operator (||) examine only the first element of the operand resulting in a single length logical vector.

The logical and operator returns true if both the operands are TRUE and returns FALSE otherwise. For example, look at this 5==5, now in this case if I run this the output would be produced as true because both left and RHS and LHS are true. However, if I put left and right operands as 4==5 it will produce an output of false because one side is not equal to other side.

Also let us say I want to assign this TRUE or FALSE to another value like this here I am assigning a value of true to the value of a and now, if I want to check what is inside a. I can run control enter on a and I will get TRUE because a true value is assigned to it. Notice that I can also reflect TRUE with T and FALSE with F so, for example F would represent a value of FALSE and R considers FALSE as 0.

So, if I multiply this with 60 or any numerical, I will get a value of 0. Similarly, TRUE or T is considered as 1 by R and therefore if I multiply it with a numeric value let us say 60 then in that case, I will observe a value of 60. Similarly, I can also check greater than or equal to and similar other operations for example, 5 > 2 this since this is true, I will get a value of TRUE. If I put a value of not equal, this exclamation mark in front of equal to this is not equal to. (!= denotes not equal to)

So, if I check whether 4 is not equal to 5 this will give me a TRUE value where because 4 is not equal to 5. So, exclamation mark along with any symbol is reverse of that that is opposite of that so, if it is attached to equal to sign it is not equal to. Also, I can have greater than equal to sign so, if I apply 5 greater equal to 4 this will be TRUE because 5 is indeed greater than 4, so that greater than condition is satisfied.

Similarly, if I want to check whether 3 is less than equal to 4 it will be TRUE because 3 is less than 4, the logical or pipe symbol returns the Boolean value TRUE if either or both operands are true and returns false otherwise. For example, if I put TRUE & TRUE both of them are true then I will get a TRUE because it is and operator so, both of these conditions need to be true or I can represent that with T&T which is TRUE.

Similarly, if I use FALSE and TRUE this will give me a FALSE because one of them is false. If I use TRUE & FALSE, I will get a FALSE. In this case if I use TRUE along with pipe operator, I will get TRUE because one of the condition is true. Similarly, I can test multiple conditions for example, I can test TRUE along with another TRUE and another TRUE in this case since all the three are true, I will get a TRUE.
If one of them is FALSE let us say I use T&F&T so, one of them is FALSE so, I should get a FALSE. However, if I was using or operation that is pipe symbol and one of them was FALSE then even in that case, I would have got a value of TRUE because at least one of them is true.

In fact, if I only one of the value was TRUE and all of them were FALSE I would still get a value of TRUE.

To summarize in this video, we have learned the use of arithmetic, relational and logical operations in R. This strengthens our fundamental understanding of R and will be helpful throughout our journey with R. **(Video Ends: 00:37:08)**

(Refer Slide Time: 00:37:09)

Now, let us start with working with vectors. **(Video Starts: 00:37:13)** In this video, we will discuss about vectors a basic data structure that plays an essential role in R programming. We will learn vector creation, vector manipulation, vector element recycling, vector element sorting and indexing vectors in R. A simple way to construct a vector in R is with c operation like this, c().

We start with vector creation a simple way to create a vector in R with this c(), this syntax which combines it's elements, many other functions also return vectors as results. For example, the sequence operator seq() generates consecutive whole numbers, while the sequence function seq. This seq function does much the same but more flexibly, we will discuss all of them one by one. So, first we will start by creating a vector using this function.

So, we can write this c(1, 2, 3, 4) this will combine all the four elements and create a new vector with four elements 1, 2, 3, 4 as we can see in the console by running ctrl enter. Now, I can assign these values or this vector and give it a name let us say V if I run this command c

(1, 2, 3, 4) and assign it to a vector name V. The new vector V will be created and if I run control enter it will carry all the four elements.

V=c(1,2,3,4)

In this way, I can create an assignment operation by assigning a vector and giving it a name. If you want to see some initial elements of this vector, you can use head command and type head (V). Similarly, if you type tail command you get some of the last elements but in this case since the vector length is very small it is only four then we are getting all the elements. In R, vector is a very important data class, a vector can be of numeric, character or factor in nature.

And as we will see or we will try to understand the nature of all these different kinds of vectors. You can also create multiple elements in a vector, you can use colon operator with numeric data to create sequences. For example, if you write 5:13 it will create a sequence from 5 to 13, we can assign this sequence to a new vector called V1 like this. So, the sequence is assigned to a variable name V1.

A new vector V1 is created I can type V1 and then ctrl enter to see all the elements inside it. If the final element specified does not belong to the sequence then it is discarded for example, let us look at this operation V = 3.8:11.4. And let us see what is inside this variable V and you can see that a vector from 3.8 sequence 4.8, 5.8 and so on is created and it goes up to 10.8, the final element is discarded, we can also use a seq operator to create a sequence.

For example, if you want to create a sequence from 5 to 9 by incrementing it with 0.4

Seq(5,9, by=0.4)

I can write a function like this seq and then specify the starting point 5 ending point 9 and then I can specify the space or the increments that is 0.4. If I run or press ctrl enter you notice starts from 5 then 5.4 then 5.8. So, there are increments of 0.4 up till 9 this way I can create even non-integer sequence.

Another example would be seq sequence command 0 starting point 1 and then I can specify the interval which is 0.1. So, it will create a sequence with an interval of 0.1 starting from 0 ending at 1. I can also instead of specifying the increments I can specify the number of elements in that for example, I can specify the sequence command starting from 0 ending at 1 and I can specify the length of this sequence as 11,

seq(0,1, length=11)
let us look at the output.

So, it has produced 11 numbers in the sequence, a sequence of 11 numbers and automatically distributed them with a gap of 0.1 which is derived from the length of the sequence. Next, we move to vector manipulation, in R two vectors of same length can be added, subtracted, multiplied or divided giving the result as a vector output. Let us understand with a simple example, let us create two vectors first vector v1 using combine operator c and 3, 8, 4, 5, 0, 11.

So, we have assigned a certain element to it we are combining them and giving them a name v1. Then we will create a vector v2 and give it certain names starting from 4, 11, 0, 8, 1, 2 will give it the name v2. Now, I can perform a vector addition operation and similarly notice each element is added one by one. For example, first element is added 4 + 3 =7 second element 11 + 8 =19 and so on so forth up till the last element which is 11 + 2 which is 13.

Similarly, I can perform vector subtraction by simply writing v1–v2 and each element it is position by position will be subtracted that is 3 – 4 =– 1, 8 – 11 =– 3 and so on up to 11 – 2 which is 9. Similarly, I can perform vector multiplication v1*v2 and again they will be multiplied look at each corresponding position that is 4 by 3 which is 12, 11 into 8 which is equal to 88 and so on up till 11* 2 which is 22.

There is also something called vector element recycling what is this? In vector recycling if we apply arithmetic operations to two vectors of unequal length then the elements of the shorter vector are recycled to complete the operation. For example, I create two vectors of an equivalent, vector 1 has larger number of elements for example it has 3, 8, 4, 5, 0, 11 so, it has six elements.

Now, I create another vector give it a name v2 and only two elements inside it 4, 11 now, if I perform an addition let us say v1 + v2 notice that there is no error message. Instead, R has created or recognized v2 as a variable which is 4, 11 it has automatically given it a length of 6 which is matching the length of v1 by replicating the sequence like this, v2= c(4,11,4,11,4,11). So, in this manner it has created eleven elements so that the vector operation is completed.

Similarly, a number of operations can be done for example another similar demonstration can be like this I create four elements here. And then I create another vector but with only three elements and if I perform the addition operation I do not get an error message but instead I get a warning message. But still the operation is performed and notice the first three element are 5 and the fourth element is 8 that means the third element that is this.

c(1,2,3,4)+c(4,3,2)

If you notice, the first element is again recycled to add to the last element of the first factor which is 4 and therefore, 4 + 4 is 8. Note that if the operands are of different lengths then the shorter version of the 2 is extended by repetition as in this case which was v1+v2 above here, where elements in v2 are 2 which is effectively repeated four times. If the length of the longer operand is not a multiple of the length of the shorter one then also a warning message is printed.

Next, we can also perform vector sorting. So, elements in a vector can be sorted using sort function. So, let us say we create a new vector v with these elements 3, 8, 4, 5, 0, 11, – 9 and 304. So, if I run this sort function sort(v), the elements are sorted from lower to higher. I can also sort them in opposite manner that is putting v.

sort(v, decreasing=T)

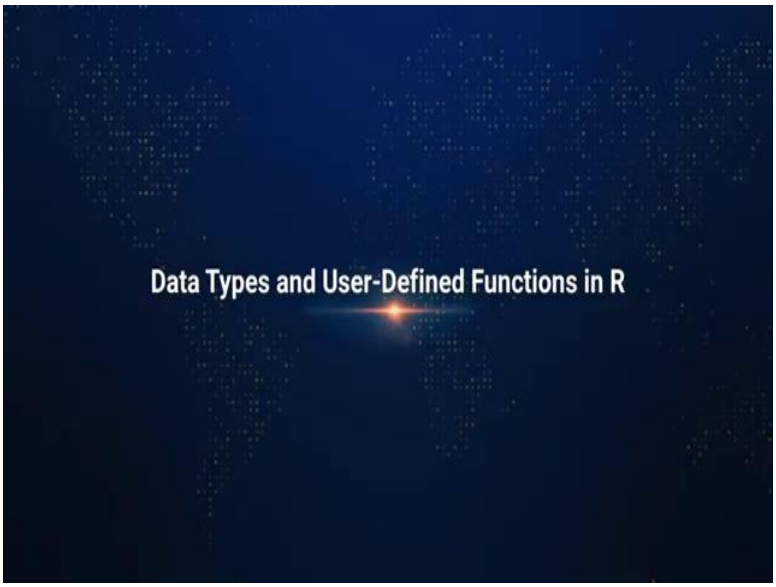
And I can put decreasing instead of increasing it will be generating a sequence of decreasing numbers. Similarly, I can create colour let us put Red, Blue, yellow, violet and I need to give the name so, I will create combination c, now these are assigned to a vector called colour. Now, if I press sort colour notice, the colour are sorted according to their alphabetical sequence so, Blue, Red, violet yellow. Next, we will move to indexing the vectors.

So, we will start with indexing of the vectors if we wish perhaps to print only one of the vector elements, we can exactly index or specify the location of the element within the square brackets. For example, we had created a layer vector called v and if you want to only print the third element inside that vector I can run v[3] this will print the third element in that vector. So, in this manner we can also specify a vector of indices.

For example, v[3:6], I can specify these four elements from their location starting from 3 up till 6. We can also specify a selected few elements for example, let us say with this I can specify first, third and fifth element, v[c(1,3,5)] So, I provided the location of first, third and fifth element which is 3, 4 and 0 only. If I want to exclude a certain elements, I can also put a minor symbol which will specify or tell R that it needs to exclude these elements.

For example, with this, v[-c(5,7)], I can exclude the elements number 5 and 7. So, if I print now, all the elements will be printed except the elements number 5 and 7. To conclude, to summarize, we have learned how to create and manipulate vectors in R in this video? We have also discussed vector manipulation using vector element recycling when vectors are of unequal length. We have also learned how to sort the vector elements and print specific elements of a vector. **(Video Ends: 00:47:28)**

(Refer Slide Time: 00:47:29)



Now, let us look at data types and user defined functions in R. **(Video Starts: 00:47:33)** To make the best use of R language, we need a strong understanding of the basic data types and data structures and how to operate on them. Each variable or vector in R is associated with specific data type. In this video we will be discussing all the data types and how to check them in R? We will also learn how to create a user defined function in R?

R has five basic data types that is numeric, character, integer, factor and logical data types. We will first define all types of data structures and then we will see how to check it in R, we start with numeric data type. Decimal values are called numeric in R it is the default data type for numbers in R. Character data type, R supports character data types where we have all the alphabets and special characters it stores character values or string values.

Then we have logical data type, R has logical data types that take either a value of TRUE or FALSE, a logical value is often created by a compression, comparison between variables. Integer data type, R supports integer data types which are the set of all integers. We also have

factor variables that is categorical variables like male, female categories, we can create as well as convert a value into different data types using functions like.

For example, you can use as.integer to convert to integer or similarly, you can use as.character(). For example, as.character() to convert to character maybe a numeric or integer to a character. Similarly, you can use as.factor() to convert into factors which is a categorical variable. So, we start discussing about data types in R. First let us illustrate numeric data type, we assign a numeric value to a variable name y, if I want to check the class of this new variable y.

y=5

class(y)

Let us use the class command and notice R has automatically recognized this variable as a numeric variable. I could have also checked the class of variable or rather a vector where two elements within quotes presented like this. Please notice now these variables or values are input in terms of in between quotes and therefore they are recognized as characters by R. So, we are forcing R to recognize these values which are originally integers.

class(c("1","2")

But we are forcing R to recognize them as characters then I can assign a value of 4 to x variable. I can assign a value of 3 to y variable and I can create a logical variable by checking whether x greater than y and assign it to z. Here the value of z would be TRUE of course because x is greater than y and if I want to check the class of z, let us see that class of z is logical which means it is a logical variable.

Let us create a new variable, let us create a vector of characters this is v2=c and I will put four elements here "My", another element is "Name", next element is "is" another element is "Abhinava". So, the character vector has four elements that is "My", "Name", "is", "Abhinava" and if I want to check the class of this variable v2. I can do that I need to assign that see the class of this vector it is a character vector.

Along with finding the class, we can also find some more features about vector like number of characters, structure and overall summary of vector or variable using the following codes. For example, I can summarize this variable or vector v2, notice it shows that it has four elements My name is Abhinava also I can find the number of characters using nchar command.

And it shows me that for all the elements first element has two characters, second element has four characters and so on. I can also deal with factor variables in R, factor in R is a variable that is used to categorize and store the data having a limited number of different categories. Factor in R is like a categorical variable that stores both string and integer values as levels.

For example, let us look at this character vector here which has a set of elements “Male”, “Female”, “Female”, “Male” and so on. So, it has set of number of elements written as “Male”, “Female”, “Female”, “Male” and so on. Now, if I assign this character vector into a new variable called v5 which is a character vector and I tell it I tell R that use this as a factor not as a character vector.

V5=as.factor(c(“M”,”F”,”F”,”M”,……)

So, I will assign this character vector in the form of a factor and now that R has recognized this variable this vector as a factor, I can check the class of this variable this is coming as factor. If you want to check the levels of this factor you can simply type v5 and for a factor variable R will put the elements along with the categories as levels which as we can see here is F and M representing Female and Male.

You can also summarize like we have been summarizing other variables you can summarize this V5. Notice it is showing me that there are six elements in Female category and six elements in Male category. So, this is how you summarize and understand the factor variable in R.

(Video Ends: 00:52:59)

(Refer Slide Time: 00:52:59)

Let us discuss user defined functions in R. **(Video Starts: 00:53:03)** As we have probably guessed R includes functions for calculating many common statistical summaries such as mean of the numeric vector. Let us consider a simple variable X and assign a number of values to it. So, we are assigning a set of values to this vector X now, if you want to compute the mean of this vector you can simply use an inbuilt function called mean which is 14.

X=c(3,5,7,9,10,12,15,18,20,25,30)

Mean(X)

If there were no mean functions, we would nevertheless have calculated the mean straightforwardly using the built-in sum and length functions like this. For example, I could have computed sum of X and divided it by the length of X, thus I would have obtained the mean of this where the length function returns the number of elements in it's argument. To do this repeatedly, every time we need to compute the mean of a numeric vector would be inconvenient.

Sum(X)/length(X)

myMean=function(X){sum(X)/length((X)}

myMean(X)

And so, in the absence of a standard mean function, we could define our own mean function like this. Here this mean function would compute the mean of the argument X now, notice you need not necessarily put X. Here, you can put Y also this is just a simple argument that needs to be fed in this function. So, I can run this now compute this function for the variable X and we again get the mean of X.

So, R function my mean here is used in the same way as a standard R elements inbuilt function and we can consequently use it in defining other functions as well. For example, the R function standard deviation can compute the standard deviation of a numeric vector. Here is a simple substitute for this SD function which is mySD function and let us use my mean to support this.

So, we will create a new mySD function which will be a substitute to SD function which is sd this function this is the original R build function, we will create R own user defined function like this. And we will use an argument Z and we will supply some of the arguments here square root now, we need to create sum of X minus my now, here we will make use of myMean function, myMean of X raised to the power 2 divided by length X – 1.

Sqrt(sum((X-myMean(X))^2)/(length(X)-1))}

Mysd(X)

sd(X)

So, this is my SD function and we can run this function. Now, I can compute the standard deviation of this X variable 8.52, we could have simply computed the standard deviation of this variable like this as well and we would have got the same values. So now, we have created this user defined function which can be used to compute standard deviation as a substitute of sd function.

Next, we move to conditional statements in R. Conditional expressions are expressions that perform different computations or actions depending on whether a predefined Boolean condition is true or false. Conditional statements include if-else statements if-else separately or a combination of them, let us see the implementation in R. We will start with a simple if-else function.

The if-else function evaluates both expressions, expression 1 and expression 2 and then returns the appropriate values from each based on the element by element value of the condition. If an element passes the condition is true then if-else statement returns the corresponding value of expression 1, otherwise it returns for expression 2. So, this basic syntax for this is like, this you have ifelse first you provide the condition.

```
#ifelse(condition, expression1, expression2)
Gender=c("Male","Female","Male","Female","Male","Female","Male","Female","Male","Female", "Male","Female", "Male","Female")
length(Gender)
for(I in 1: length(Gender)){
ifelse(Gender[i]=="Male", print("M"),print("F"))
}
```

If the condition is true then expression 1 will be evaluated, if condition is false then expression 2 would be evaluated. How do we implement it? Let us say, let us define a variable called Gender and it is a categorical variable so, we will put certain Male and Female elements inside. So, we will put certain elements inside this gender vector, now that we have assigned these values to our gender vector, what we will do is?

We will check which of these values are Male or Female and for that we will use our ifelse function and we will check with whether the value of Gender is Male. In case it is Male then we will print M, if it is Female then we will print F. So, if I run this function, notice for each

M and F Male and Female, we have M F M F as printed there. What is the length of this gender variable?

We can use length function to get the length which is 12. Now, we will make use of for loop to print each element of this Gender variable. We can easily make use of this R for loop, the syntax of the for loop goes like this, to type for and it will have the same effect notice, I in 1 to length of gender. So, we will run as many times as the length of this variable Gender then again inside this we will put ifelse loop.

And now we will put Gender and we will put subset i, so that every time this i counter changes the value of Gender moves from one element to next and again, we will put the same condition Male to check each element one by one. If it is Male it should print M, if it is Female the expression 2 which is here print F should be executed. So, now we will run R for loop and see the output.

And notice every time the counter is run starting from i= 1 to 12 since we have put it to end at length of Gender which is 12. So, twelve times the loop will run and every time it will compare the value of Gender with whether it is Male if it is Male the first expression will be executed it will print M. And if it is false then second expression which is print F so, F will be printed if it is Female.

So, now we have discussed one example of a simple ifelse statement and one example of a for loop, we will also discuss some advanced conditional comments in detail in the next module under data handling section in R. To summarize, we have learned how to quickly check the class of a vector or variable in R? We have also seen how R facilitates us to create user defined functions and smoothens further calculations?

We have seen some basic conditional statements ifelse and for loop and will gradually move to the advanced part in subsequent sessions for data analytics. **(Video Ends: 01:01:19)**
(Refer Slide Time: 01:01:20)

(Video Starts: 01:01:21) To summarize in this video, the learners are familiarized and initiated with our platform. After completing this lesson learners will be able to create a new project and write basic codes such as simple mathematical and logical operations. Learners will be able to create new and different types of data objects such as vectors. The Learners will also be able to create user defined functions and automate basic mathematical and logical tasks.

These topics will help learners and act as building blocks in implementing tasks and assignments pertaining to more advanced topics such as predictive analytics using R programming. In the next lesson, we will discuss the types of data science projects and their life cycle. **(Video Ends: 01:01:58)**