Foundations of R Software
Prof. Shalabh
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture - 25
Sequences

Hello friend. Welcome to the course Foundations of R Software. And, from this lecture we are going to begin with a new topic. And, all these topics are going to be related to the management of data. You know this R software is useful for computation, simulation as well as managing the data sets.

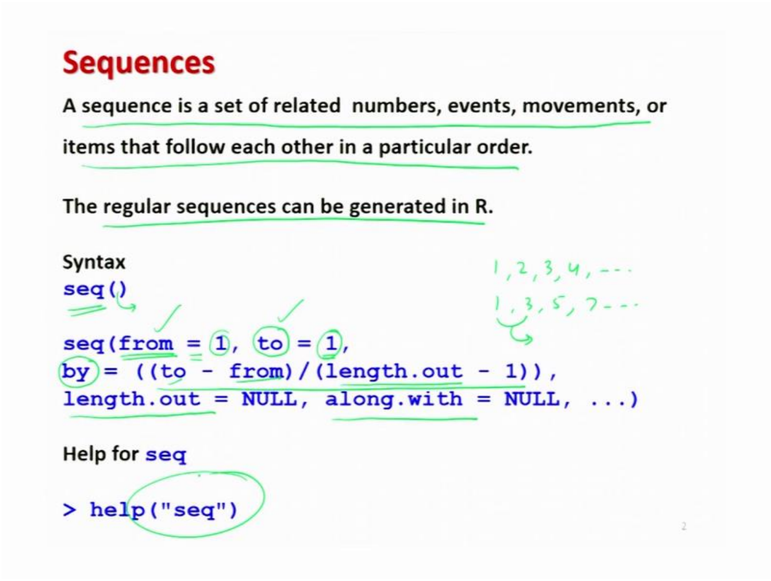
So, what really happened that when we are trying to handle the databases, then many times we have to manipulate them. And, also when we are trying to do over computation, simulation etc.; many times we also need to generate a particular type of values. Now, beginning in this direction we are going to start a topic on Sequence.

So, in this lecture and in the next two lecture, I am going to talk about sequences. So, now, if you try to understand the literal meaning of a sequence, what is that? Sequence is some arrangement of numbers in a particular in a proper way which is pre-defined. That means, you need to define the way you want to arrange your numbers in a particular fashion.

So, how to do all these things in the R software, that is our main motive. In this lecture and in the further lectures, I will try to take some more operations and topics which are related to the sequence of numbers. Now, there are many many operations which are possible in the sequence.

So, how to get it done and some of the most representative examples which are a quite popular and more useful, I will try to explain you with the help of several examples. So, let us try to begin this lecture and try to understand how you can handle the sequence in the R software.

(Refer Slide Time: 02:29



Sequences

A sequence is a set of related numbers, events, movements, or items that follow each other in a particular order.

The regular sequences can be generated in R.

Syntax

```
seq()

seq(from = 1, to = 1,
by = ((to - from)/(length.out - 1)),
length.out = NULL, along.with = NULL, ...)
```

1, 2, 3, 4, - - -

1, 3, 5, 7 - - -

Help for `seq`

```
> help("seq")
```

So, now if you try to see here, what is the sequence? So, sequence is essentially a set of related number, events, movements or even item that follow each other in a particular order, particular fashion, particular way. And, the usual regular sequences can be generated in R without any difficulty. And, the command to generate such sequences is s e q.

And, inside the parentheses you have to write the different types of values, parameters and related options. Now, there are many many options in this syntax s e q, all in lower case alphabets. But, I am going to consider here some popular options which I personally feel that they are going to be useful when you are trying to handle the numerical values.

So, now if you try to see one of the basic operations in the sequence will be that you would like to fix that what is the value from which the sequence should start. And, in order to control it, we have to give here an option here like this inside the parenthesis from. And, then from equal to whatever value you want; here I am writing 1, but you can write anything which is allowed.

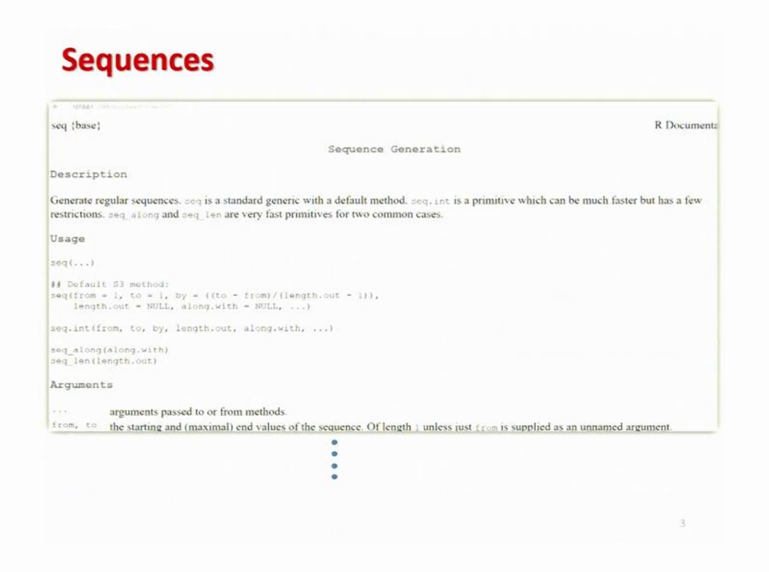
And, then you would like to fix that how long the sequence will go; that means, the what should be the last value of the sequence. So, that can be controlled by this option here to t o to. And, you try to write down here to is equal to whatever the value you want, here I am writing here 1, but you can choose anything. So, this from and to they are going to control the starting and end point of the sequence.

Now, the usual way in which the sequence is going to progress is by 1 unit like as 1, 2, 3, 4 etc. Now, you have different options that you can choose the sequence should be like 1, 2, 3, 4 etc. or like 1, 3, 5, 7 etc. or this can be something else whatever you want. So, basically if you try to see you need to inform here somewhere between the 2 numbers, that how the sequence should progress and by what value it should increase.

So, in order to control this jump, we have an option here by b y by and then we try to define here the value by which the sequence has to progress more. The usual way in which it will progress that is defined by the value at to, value at from, divided by length dot out minus 1, right. And, then means there are some other option like l e n g t h dot o u t length out along with a l o n g dot w i t h etc. And, there is a long list of such options.

So, I will try to consider here couple of examples and through which I will try to show you that how these options can be used. But, I would request you that please try to look at the help on this topic sequence s e q and try to understand that how the things are going to actually work.

(Refer Slide Time: 05:50)



Sequences

seq {base} R Documenta

Sequence Generation

Description

Generate regular sequences. seq is a standard generic with a default method. seq.int is a primitive which can be much faster but has a few restrictions. seq_along and seq_len are very fast primitives for two common cases.

Usage

```
seq(...)

## Default S3 method:
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)),
    length.out = NULL, along.with = NULL, ...)

seq.int(from, to, by, length.out, along.with, ...)

seq_along(along.with)
seq_len(length.out)
```

Arguments

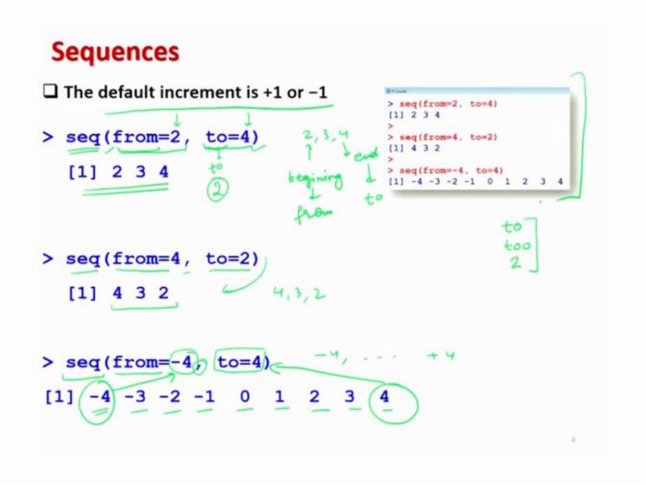
...	arguments passed to or from methods.
from, to	the starting and (maximal) end values of the sequence. Of length 1 unless just from is supplied as an unnamed argument.

3

And, if you try to look into the help sequence, you will get here this type of say screenshot and then you can see here there are many many details of various operations in the sequence. So, this exercise as I am leaving up to you and I request you once again that you please at least try to read that what are the different options.

I am not asking you to remember each and everything always, but at least once you read it you will come to know that what are the different possibilities. And, based on that whenever you need them, you can look into the help and can get the right syntax.

(Refer Slide Time: 06:24)



Sequences

❑ The default increment is +1 or −1

```
> seq(from=2, to=4)
  [1] 2 3 4

> seq(from=4, to=2)
  [1] 4 3 2

> seq(from=-4, to=4)
[1] -4 -3 -2 -1  0  1  2  3  4
```

```
> seq(from=2, to=4)
[1] 2 3 4
>
> seq(from=4, to=2)
[1] 4 3 2
>
> seq(from=-4, to=4)
[1] -4 -3 -2 -1  0  1  2  3  4
```

But, now let me try to take here some examples and through those example I try to show you that how the outcome in a sequence can be controlled depending on your wish, depending on your requirement. So, one point which you have to keep in mind is that the default increment in the sequence is plus 1 or minus 1, 1 unit either positive or negative.

And, in case if you want to generate such a sequence where you want to make an increment or decrement of 1 unit, then you have to write here like this seq. And, then inside the parenthesis you write here from equal to whatever value you want from where you want to start the sequence and to t o. And, then you try to write down here to equal to whatever point up to which you want to have the sequence.

So, for example, I want to have here a sequence like 2, 3 and 4. So, you can see here this is my beginning point and this is my end point. So, this beginning point will be indicated by the option from and end point is going to be indicated by the option to. So, I can write down here from equal to 2 comma to equal to 4. Please understand I am saying here t o to, not 2 like this number. And, yeah this you have to be watchful during the entire lecture, that in case if I am using the to like t o to or t double o too or the number 2.

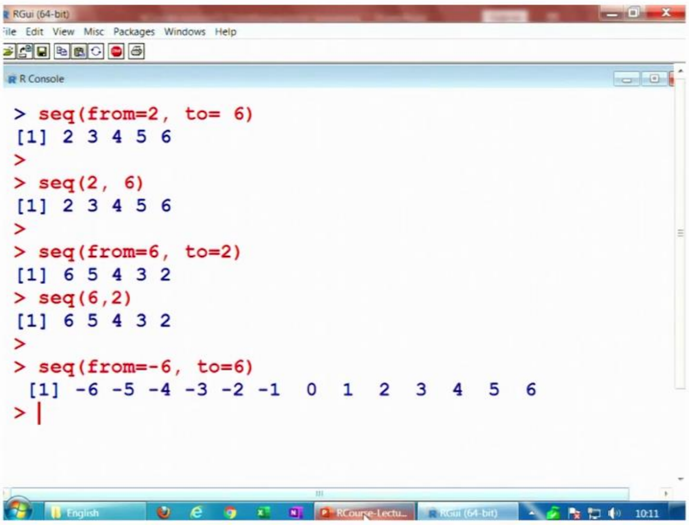
My audio is always going to be only to. So, just be careful, watchful. So, now if you try to execute this command here, it will give you this type of outcome 2 3 4. And, it is not a condition that the value which has to be given in the from is always have to be smaller than the value which is to be given in the to command. Even if you want to have a sequence like 4 3 2 which is decreasing by 1 unit; you can also write here seq.

And, inside the parenthesis from equal to 4 and comma then t o to equal to number 2. And, once you operate it, you will get here a value like here 4 3 2. And, in this sequence command it is also possible that the values which you are going to consider in the from and to option, they can be negative also. So, if you try to see here I want to generate here a sequence which is beginning from -4 and then it is coming up to +4.

So, for that I try to give here the command seq and inside the parenthesis I try to write from equal to -4, then comma and then t o to equal to 4. So, now what will happen? This sequence will start from -4 and then there is a difference of 1 unit and the sequence will progress further, like its -4 -3 -2 -1 and 0 and then after that 1 2 3 4.

So, this -4 is corresponding to this from and this 4 is corresponding to this option to right and here you can see this is the screen shot here. So, before I move forward, let me try to give you here these examples on the R console here. So, that you can understand that what is really happening and how it will look like.

(Refer Slide Time: 09:36)



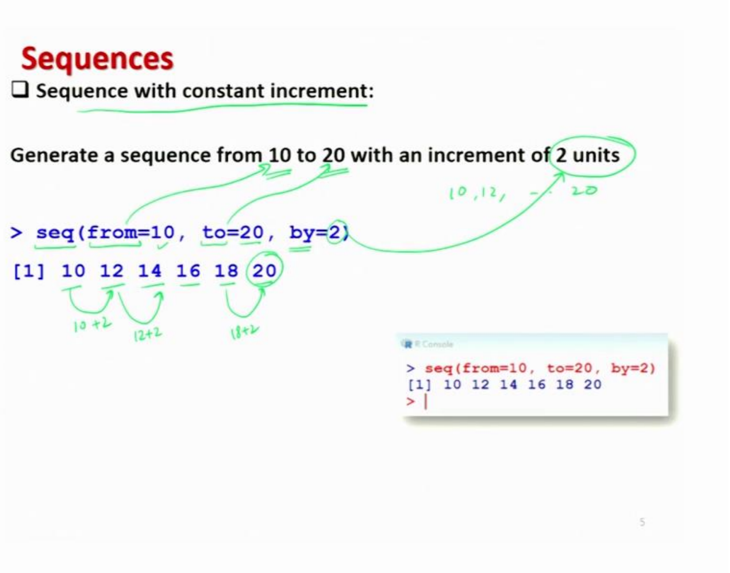
```
> seq(from=2, to= 6)
[1] 2 3 4 5 6
>
> seq(2, 6)
[1] 2 3 4 5 6
>
> seq(from=6, to=2)
[1] 6 5 4 3 2
> seq(6,2)
[1] 6 5 4 3 2
>
> seq(from=-6, to=6)
 [1] -6 -5 -4 -3 -2 -1  0  1  2  3  4  5  6
> |
```

So, I try to write down here sequence from equal to 2 and say here to is equal to say here 6. So, you can see here this is giving you a 2 3 4 5 6, right. And, one option I would like to show you here is that even if I try to write down here seq and simply I try to write down here 2 comma 6, still I will get the same outcome 2 3 4 5 6.

So, in this command seq, the first value is the default value for from and the second value after comma is the default value for the option t o to, right. But, my advice to you all will be that always try to write the complete command so, that you will understand and then there will be no confusion. But, I leave this decision up to you. Now, in case if I want to have a sequence which is beginning say from say 6 and it is going up to here 2.

Then, I have to write from equal to 6 and t o to equal to 2 and you can see here this is here 6 5 4 3 2, right. And, the same thing also you can write like here say the sequence s e q 6 comma 2 and it will give you the same outcome. And, similarly if you want to have a sequence which is beginning from say -6 and it is going up to say 6. So, I can write down here from is equal to -6 and t o to is equal to 6 and this outcome will be here -6 -4 up to here.

(Refer Slide Time: 11:13)



Sequences

❑ Sequence with constant increment:

Generate a sequence from 10 to 20 with an increment of 2 units

```
> seq(from=10, to=20, by=2)
[1] 10 12 14 16 18 20
```

```
> seq(from=10, to=20, by=2)
[1] 10 12 14 16 18 20
> |
```

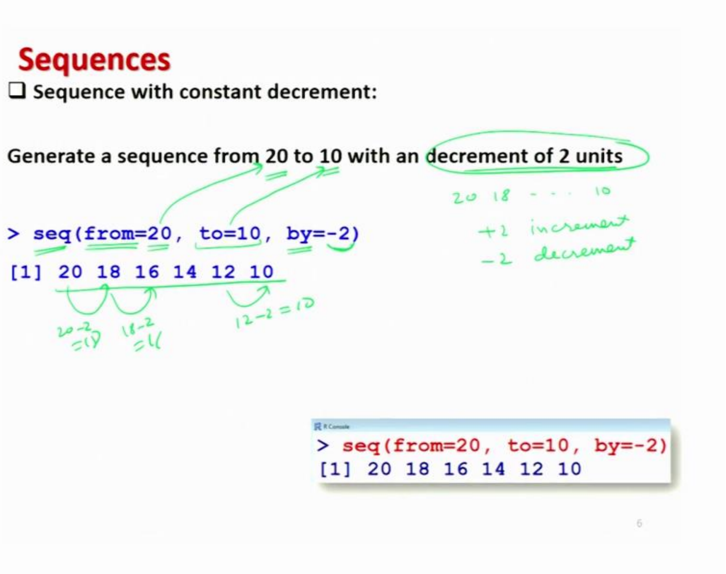
So, now you can see that it is not a very difficult operation to do. And, now I am sure that you will be confident that when I am trying to show you these operations on the computer screen and in the slides, you can believe that yes this is going to work. So, now

I try to take here one more operation that I want to generate a sequence with constant increment. And, my objective is that I do not want the default increment and suppose I want to generate a sequence which starts at 10 and this ends at 20 and the increment in the values is of 2 units.

So, you can say like here 10 12 14 up to here 20 like this one. So, now, how to define this sequence in the R software? So, you can write down here seq and then from equal to 10 and t o to is equal to 20 and then after the comma you write here b y by and by you can write down here 2. Because, this 2 is corresponding to this increment of 2 units and similarly this from is corresponding to this 10 and to is corresponding to this value 20.

So, now this is going to generate a sequence like 10 12 14 16 18 and 20. So, you can see here a from this 10, it at say 10 plus 2 then 12 plus 2 14 and so on. And, this operation will continue up to here 18 plus 2, but then as soon as you reach to the 20, the R will stop here, right.

(Refer Slide Time: 12:37)



Sequences

❑ Sequence with constant decrement:

Generate a sequence from 20 to 10 with an decrement of 2 units

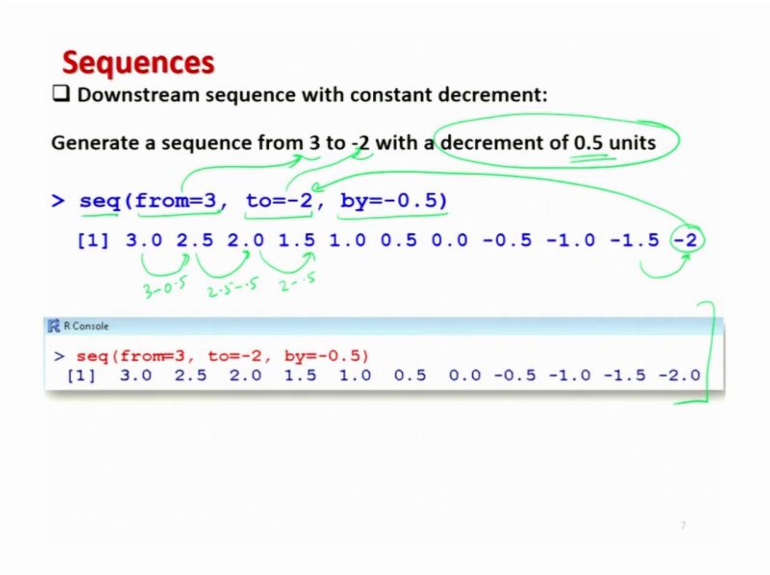
```
> seq(from=20, to=10, by=-2)
[1] 20 18 16 14 12 10
```

```
> seq(from=20, to=10, by=-2)
[1] 20 18 16 14 12 10
```

So, that is not a very difficult thing. Now, similarly I want to have a sequence which is decreasing. So, suppose I want to begin a sequence from 20 and I want to end here at 10 and I need a decrement of 2 units. So, it will be like 20 18 etc. up to here see here 10. So, I use here the command s e q and inside the parenthesis I write down here the from, from will be say 20 then t o to up to here 10.

And, then now because I want here a decrement; so, I try to give the value of b y by as minus 2. So, plus 2 will indicate the increment and minus 2 is going to indicate the decrement. And, if you try to do it, you will get here the values like 20 18 16 14 12 and 10. So, you can see here 20 to minus 2 is equal to 18, then 18 minus 2 is equal to 16 and up to here this will continue here 12 minus 2 is equal to 10 and so on. So, once again you can see it is not a very difficult operation.

(Refer Slide Time: 13:41)



Sequences

❑ Downstream sequence with constant decrement:

Generate a sequence from 3 to -2 with a decrement of 0.5 units

```
> seq(from=3, to=-2, by=-0.5)
  [1] 3.0 2.5 2.0 1.5 1.0 0.5 0.0 -0.5 -1.0 -1.5 -2
```

R Console

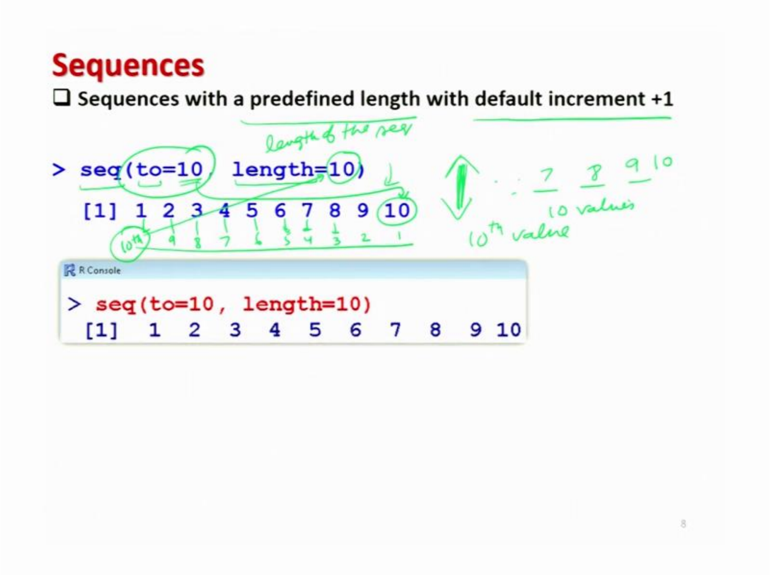
```
> seq(from=3, to=-2, by=-0.5)
 [1]  3.0  2.5  2.0  1.5  1.0  0.5  0.0 -0.5 -1.0 -1.5 -2.0
```

And, now I try to give you here one more option where I want to show you that this decrement is not only an integer value. But, that can be a fractional value also, that can be a fractional value as well as that can be a positive as well as negative value. So, now, you have here different types of choices that you want to increase or decrease your sequence by any value, right.

So, for example, if I want to generate a sequence starting from 3 going up to -2 and I want a constant decrement of 0.5 units, right. That means, the difference between the two consecutive terms should be 0.5 and the sequence has to be decreasing, that mean the values are decreasing. So, I try to give this command here say here s e q and then from 3. So, this is coming from here then t o to which is coming from here -2 and then because there is a decrement of 0.5 units.

So, I try to write down here by is equal to -0.5. And, now what will happen? The sequence will start from 3 and then it will be 3 minus 0.5, this will come out here 2.5. And, then once again 2.5 minus 0.5 and then 2 to 1.5 will become like 2 minus 0.5 and so on. And, this value will continue until you reach to the value minus 2 which is here and if you try to do it in the R console, you will get here this type of screenshot, right.

(Refer Slide Time: 15:15)



Sequences

❑ Sequences with a predefined length with default increment +1

```
> seq(to=10, length=10)
  [1] 1 2 3 4 5 6 7 8 9 10
```

```
> seq(to=10, length=10)
 [1]  1  2  3  4  5  6  7  8  9 10
```

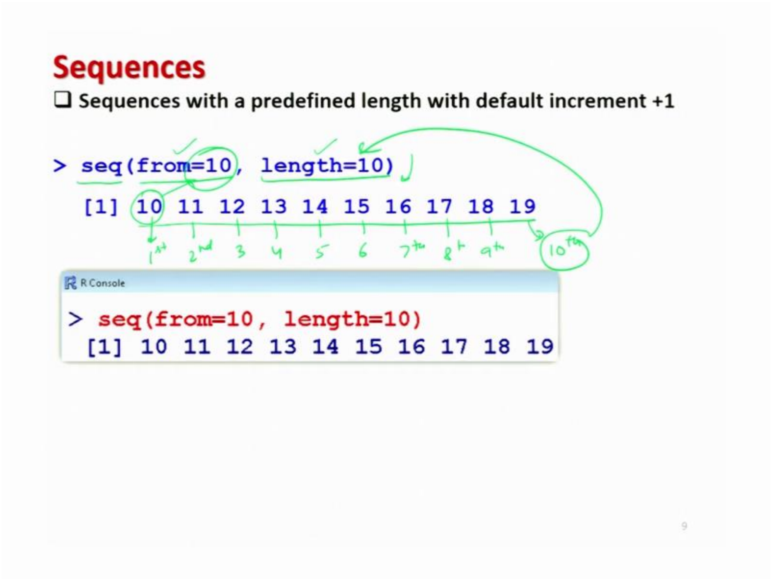
Now, I try to give you here one more option, that you simply know that I want to have a sequence of pre-defined length; that means, you know that how many elements you want in the sequence. And, you simply know one of the values with the to or say from and suppose just for the sake of simplicity, I will say that we want the default increment of plus 1 right.

So, in such a case what you would try to do? Suppose, I try to generate here a sequence that goes up to the last point say here 10 so; that means, before that I want here 10 values. So, they will become here 9 8 7 and so on and the sequence is going to stop here somewhere as soon as you get the 10th value.

Now, if you try to see that without doing any manual calculation, you can generate this sequence in the R software by using the command s e q. And, inside the parenthesis you write a t o equal to 10 and then l e n g t h that is the length and this is actually the length of the sequence or the number of elements you want in the sequence. So, that is going to be here 10.

So, what will happen here, that as soon as you execute you will get here this type of outcome and if you try to see here the last point, this to here is like here so, this 10. And, then after that you come here this is my 1st value, this is my 2nd value, this is my 3rd value, this is my 4th value, this is my 5th value, this is 6th, this is 7th, this is 8, this is 9th and this is my here 10th value. So, this 10 is corresponding to this 10 which is the length of the vector or length of the sequence, ok.

(Refer Slide Time: 17:04)



Sequences

❑ Sequences with a predefined length with default increment +1

```
> seq(from=10, length=10)
  [1] 10 11 12 13 14 15 16 17 18 19
```

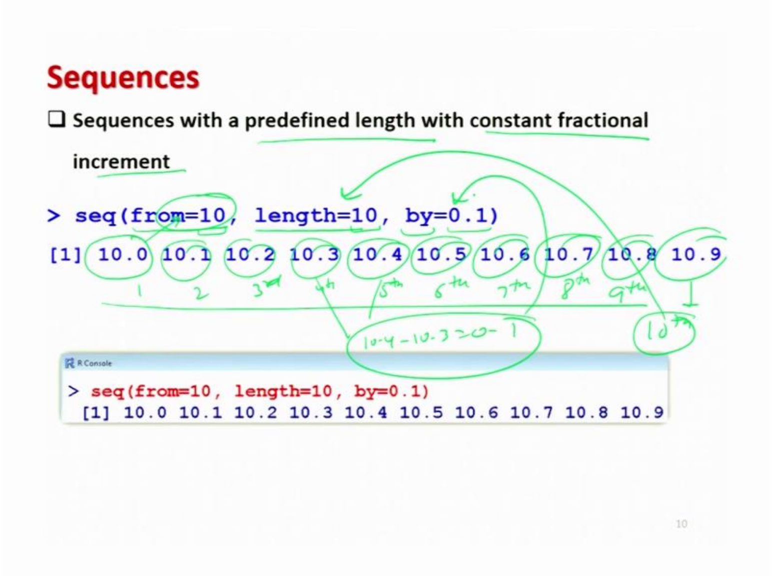
R Console

```
> seq(from=10, length=10)
 [1] 10 11 12 13 14 15 16 17 18 19
```

So, you can see here these things are quite easy to do. And, similarly I try to take here one more example where you know that you want to generate a sequence of the length 10, but you want to start from 10. Means earlier your end point was 10, but now I want to have the starting point as 10. So, if you try to write down here say s e q, inside the parenthesis you write from is equal to 10 and the l e n g t h all in lower case alphabet length is equal to 10.

And, as soon as you execute it, you will get here this outcome. So, if you try to see here, this is beginning at 10. So, this 10 is corresponding to this value here from. And, now this is my here 1st value, 11 is my 2nd value, 12 is 3rd, 13th is 4, 14 is 5th, 15th is 6th, 16th is 7th, 17th is 8th, 18th is 9th and 19 here is the 10th value. So, this 10 is corresponding to this length 10 right. So, there are 10 elements in this sequence and which are beginning at 10. So, you can see here it is not a very difficult thing to generate such a sequence in the R software.

(Refer Slide Time: 18:05)



Sequences

❑ Sequences with a predefined length with constant fractional increment

```
> seq(from=10, length=10, by=0.1)
[1] 10.0 10.1 10.2 10.3 10.4 10.5 10.6 10.7 10.8 10.9
```

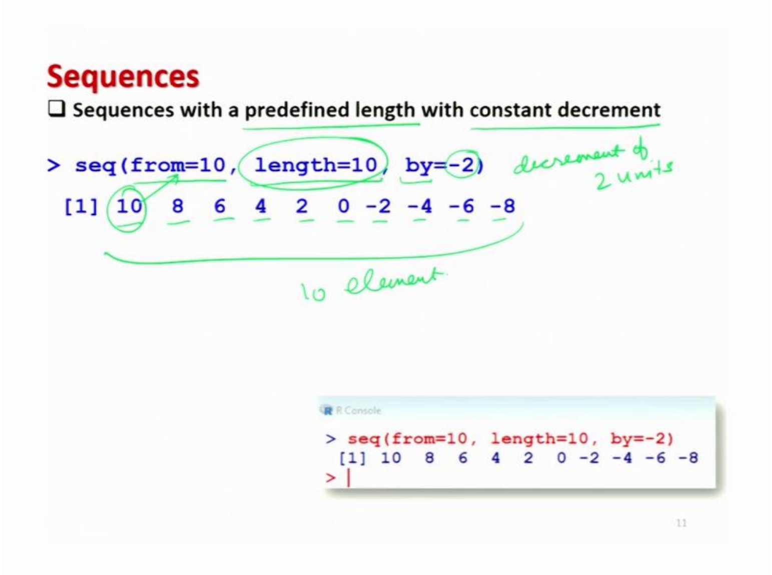
```
> seq(from=10, length=10, by=0.1)
 [1] 10.0 10.1 10.2 10.3 10.4 10.5 10.6 10.7 10.8 10.9
```

And, now I try to take here a combination. This combination is I want to generate a sequence of pre-defined length which has a constant rational increment. So, now you know that you just have used this command here, where you have used the sequence command with from and length. So, now, in this command I will use here the command here by, this option here by. So, suppose I want to generate a sequence which starts at the value 10 and it has 10 number of elements and these numbers are going to be increasing by the fraction of 0.1 length.

So, I have to write here from is equal to 10, l e n g t h, length is equal to 10 and by is equal to b y by is equal to 0.1 and this will give you here this type of outcome. So, you can see here this is beginning at 10. So, this corresponding to this from 10 and now this is my 1st value, this in 2nd, this is 3rd, this is 4th, this is 5th, this is 6th, this is 7th, this is here 8th, this is here 9th and this is here 10th value.

So, this 10 is going to be corresponding to this length 10 and the difference between the two consecutive terms, if you try to see. Suppose for example, 10.4 minus 10.3 is 0.1; so, this is corresponding to this by. So, this is how you can generate the sequence without any problem.

(Refer Slide Time: 19:28)



Sequences

❑ Sequences with a predefined length with constant decrement

```
> seq(from=10, length=10, by=-2)
 [1] 10  8  6  4  2  0 -2 -4 -6 -8
```

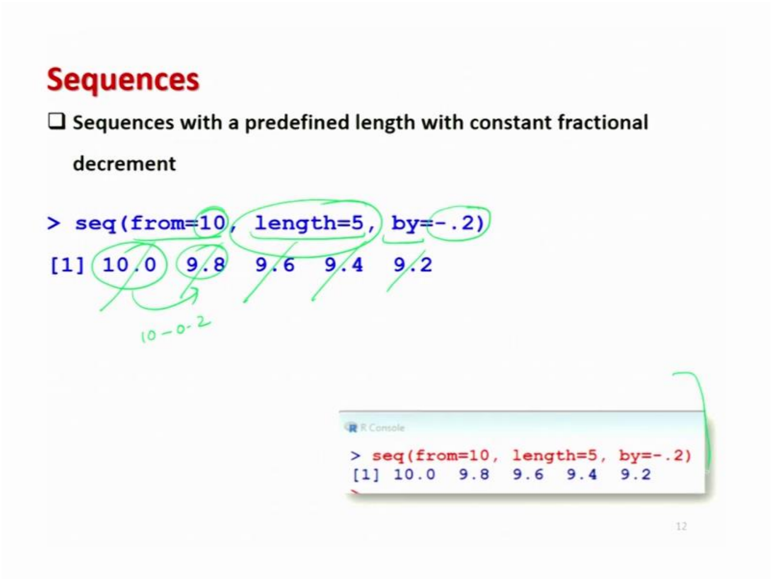
```
> seq(from=10, length=10, by=-2)
 [1] 10  8  6  4  2  0 -2 -4 -6 -8
> |
```

And, similarly if I want to generate a sequence just like what we have done, but I want to have a constant decrement. So, I want that the length of the sequence is predefined and there is a constant decrement. So, I can use either from or to whatever you want. So, suppose I want to generate a sequence here which is beginning from, from is equal to 10 and it is going to have 10 values in the sequence.

So, I can write down here length is equal to 10. And, now what about this by? b y is equal to -2, because I want here a constant decrement of 2 units right. This is here, decrement of 2 units. If you want to give something else, that also you can give. So, this is going to give you a sequence like a 10 8 6 4 2 0 -2 -4 6 -6 -8. And, you can see here these are the 10 elements.

This 10 is corresponding to this from and this number of elements this is corresponding to here this command here; length is equal to 10. And, the difference between the two consecutive terms is say 2 units and they are decreasing. So, this is by is equal to -2.

(Refer Slide Time: 20:35)



Sequences

❑ Sequences with a predefined length with constant fractional decrement

```
> seq(from=10, length=5, by=-.2)
[1] 10.0  9.8  9.6  9.4  9.2
```

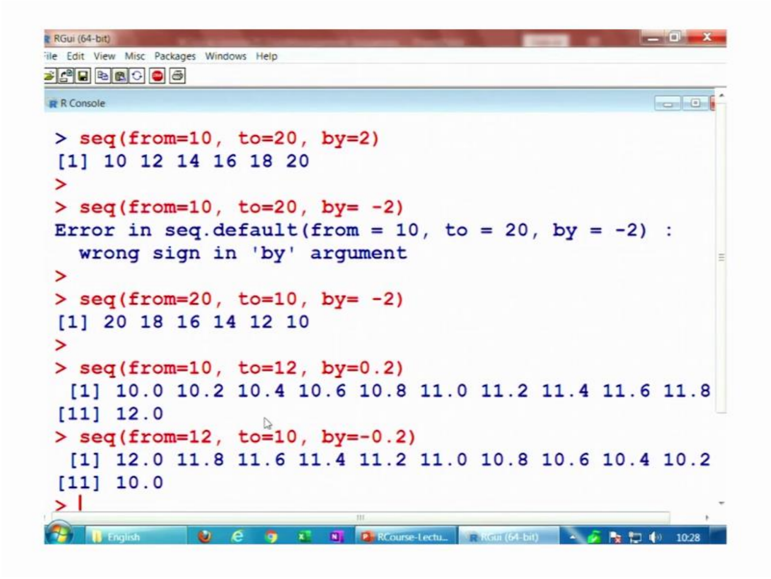
```
> seq(from=10, length=5, by=-.2)
[1] 10.0  9.8  9.6  9.4  9.2
```

So, now you can see here it is not a very difficult thing. Now, in the same example suppose I want to make a constant decrement, but that is not an integer, but it is a fraction. So, how to do it? Now, you know after this you means I am sure that you have understood how the things are working and you can use your this common sense also to make different types of combinations.

So, you can see here now I want to have a sequence which starts at the value 10 and it has 5 elements. So, the command for l e n g t h length is equal to 5 and I want here a constant fractional decrement of 0.2 right. So, I can write down here b y is equal to minus 0.2. So, what will happen? This sequence will begin at 10 and then there is a decrement of 0.2. So, this will become a 10 minus 0.2 is equal to 9.8 and this will continue.

So, this will be a 1 2 3 4 and here 5 values which is controlled here by the command length is equal to 5. So, you can see here it is not a very difficult operation and you can see here, if you try to do it in the R console you will get here this type of outcome. So, now let me try to show you these things on the R console. And, I will try to inform you here that here I have taken the example using the from, but you can also take it here to and there can be many-many combinations which actually you can do here.

(Refer Slide Time: 22:01)



```
> seq(from=10, to=20, by=2)
[1] 10 12 14 16 18 20
>
> seq(from=10, to=20, by= -2)
Error in seq.default(from = 10, to = 20, by = -2) :
  wrong sign in 'by' argument
>
> seq(from=20, to=10, by= -2)
[1] 20 18 16 14 12 10
>
> seq(from=10, to=12, by=0.2)
 [1] 10.0 10.2 10.4 10.6 10.8 11.0 11.2 11.4 11.6 11.8
[11] 12.0
> seq(from=12, to=10, by=-0.2)
 [1] 12.0 11.8 11.6 11.4 11.2 11.0 10.8 10.6 10.4 10.2
[11] 10.0
> |
```

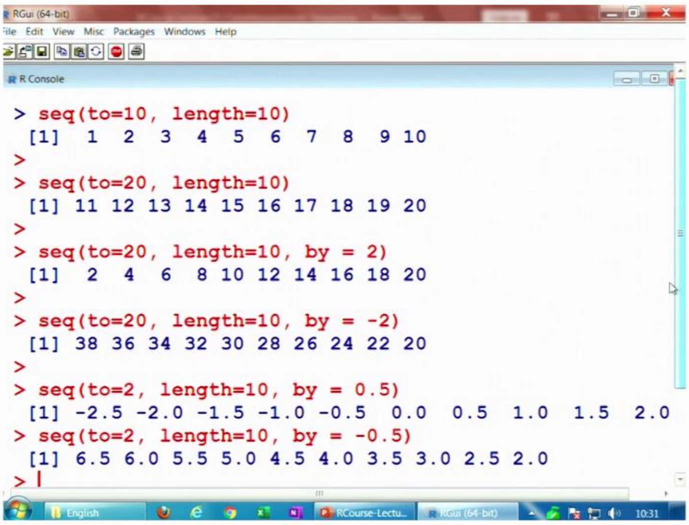
So, for example, if I want to generate here this type of sequence which is beginning from 10 and then going up to 20 and there is a constant increment of 2 units. So, this will here look here like this. Now, in case if you want to have here the constant decrement. So, you simply have to write here by is equal to -2. So, this will decrease every term by minus 2 that is like this, you can see here.

This is wrong why? That now you have to understand, because if you try to see mathematically how can we start a sequence from 10 going to 20 and this is going to be decrement. This is a mathematical contradiction. So, I wanted to show you just that you have to think before you create a sequence. But, now in this case if you try to write down here, you want to begin the sequence at 20 and you are going up to 10.

Then, it is going to be a decreasing sequence, right and you can see here this is going to give you this value. There is no issue right. Now, similarly if you want to have a sequence say starting from here 10 and going to suppose here up to 12 and you want a constant increment of say fraction say 0.2. So, you can see here this is here like this, 10.0 10.2 up to here 12, right.

And, similarly if you want to have here a decreasing sequence. So, decreasing sequence is starting from 12 and it is going up to 10 and the decrement here is by 0.2 units. So, if you try to see here, this will give you here this type of command, right. And, now after this we try to take care these commands here, where we have some defined here length.

(Refer Slide Time: 23:41)



```
> seq(to=10, length=10)
 [1]  1  2  3  4  5  6  7  8  9 10
>
> seq(to=20, length=10)
 [1] 11 12 13 14 15 16 17 18 19 20
>
> seq(to=20, length=10, by = 2)
 [1]  2  4  6  8 10 12 14 16 18 20
>
> seq(to=20, length=10, by = -2)
 [1] 38 36 34 32 30 28 26 24 22 20
>
> seq(to=2, length=10, by = 0.5)
 [1] -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0
> seq(to=2, length=10, by = -0.5)
 [1] 6.5 6.0 5.5 5.0 4.5 4.0 3.5 3.0 2.5 2.0
> |
```

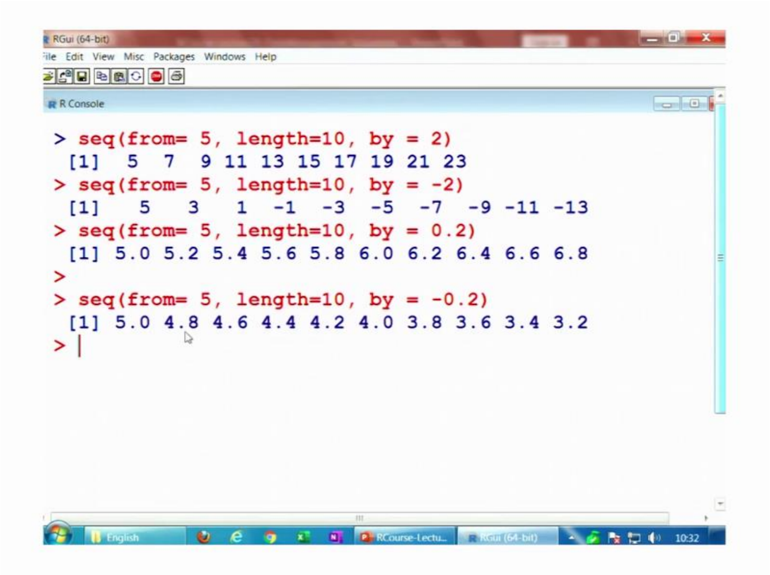
So, if you try to see here, suppose I want to have a sequence of length 10 which is beginning at 10. Then, I have to give here t 0 is equal to 10 and this you can see here, these are my 10 elements. Similarly, if you want to start here a sequence in such a way that ends at 20, then you can give here 2 is equal to 20 and then the length is going to be here 10. And, you can see here that will start at some number, but the end point is given here by 2.

And, similarly if you want to have here some constant increment or decrement; so, you can see here you can give here say constant increment is suppose by 2. So, you can see here this is ending at 20, but now this is giving you a 20 18 16 up to here 2, because this is the length of the data vector which is giving you here the 10 elements in the data.

Now, suppose if I try to use here by is equal to -2, let us try to see what happened. Yeah, that works why? Because, your this 2 here is 20. So, this is fixed and you are simply trying to ask here that you want 10 elements and which are such that they are in the decreasing and the number of total elements in the vector should be 10. So, this starts at 20, then 20 22 24 and then it ends here 38.

Now, in the same command, in case if I try to change this value up to here to some fraction here say 0.5. So, I want to begin my sequence at some point so, that it is ending at 2 and I need here 10 number of elements in my sequence which are at a constant increment of 0.5. So, we can see here that with the final value here is 2.0 and after that the values are coming here in this direction and the total number of values are here 10. And, in case if you want to have a constant decrements and you can give here the option here by is equal to -0.5 and you can see here these values are here, right.

(Refer Slide Time: 25:40)



```
> seq(from= 5, length=10, by = 2)
 [1]  5  7  9 11 13 15 17 19 21 23
> seq(from= 5, length=10, by = -2)
 [1]   5   3   1  -1  -3  -5  -7  -9 -11 -13
> seq(from= 5, length=10, by = 0.2)
 [1] 5.0 5.2 5.4 5.6 5.8 6.0 6.2 6.4 6.6 6.8
>
> seq(from= 5, length=10, by = -0.2)
 [1] 5.0 4.8 4.6 4.4 4.2 4.0 3.8 3.6 3.4 3.2
> |
```

And, now similarly as you have used here to, I can also use here from say sequence from is equal to suppose 5 and I want to give here length is equal to 10 and I want to increase it by 2. You can see here you can also use here from and this increment that can also be decrement. So, if you try to give here the by equal to here -2. This will give you here this type of decrement and similarly that can also give you here constant increment or decrement.

So, if you try to take a constant increment in the same sequence, you can see here it is here like this. And, if you want to have a the constant decrement here which is here like this. So, you can see here all sorts of combinations are possible in this R software to generate different types of here lengths, right. So, there should not be any problem in now generating a particular type of sequence that you want.

So, now, we come to an end to this lecture and you can see here my objective was very simple here. I have used only here one command s e q, but this sequence command has many applications. But, how are you going to control the parameter so, that you get the outcome of your requirement, that is the judgment which you have to take, right. And that is actually according to the need what you have.

But, according to that you have to choose the correct value of the parameters from, to, by etcetera. And, beside these things there are many more possibilities and options. So, you can choose any two or three combinations and that will give you a different types of sequence. So, I would say why do not you go in the other way, that you try to understand what these parameters, these options are doing, how they are working. And, then you try to take a particular sequence and see how you can generate it, that will be a very interesting exercise.

So, you try to practice it and I will see you in the next lecture with more options on the sequence command, till then goodbye.