Time Series Modelling and Forecasting with Applications in R

Prof. Sudeep Bapat

Shailesh J. Mehta School of Management

Indian Institute of Technology Bombay

Week 01

Lecture 05: Practical Session in R - 1

Thank you. Hello all, so welcome to this last lecture in week 1 of this course on time series forecasting with applications in R. So, the idea for this lecture is just to summarize very quickly as to where we stopped last time, so this entire idea about stationary processes and non-stationary processes etc. And then I will devote the entire time by playing around a few things in R. So, as even mentioned in the outline, this is a kind of practical course. So, many of you might not be very acquainted with the statistical software R.

So, maybe what we can do is we can just run through the entire process one more time. So, how do you install R and then how do you play around with a few numbers here and there in R, all right. So, we will deal with that, okay. But again, just before that just to finish this idea about stationarity and non-stationarity. So, converting a non-stationary process to stationary, and why is it important, ok?

So, again just to summarize very quickly as to what you mean by a non-stationary process from the last lecture, we have seen that, either if you have a trend or seasonality or cyclicality for that matter or changing variance. So, these are some of the key ideas as to when a time series process would be non-stationary. So, again just to summarize this quickly using some graphs. So, let us say if a time series is trending upwards then it will be non-stationary or if you have seasonality. So, seasonality means repetitions.

So, let us say temperature data. or rainfall data. So, all is a seasonal, right? So, you have some specific repetitions here, okay? So, again the time series is non-stationary and lastly changing variance.

So, changing variance means what? So, let us say initially the variance is small, but then it grows, all right? So, this is a problem of changing variance, okay? Now, the idea is why do you have to convert a non-stationary process of stationarity before you move on? As the distributions, moments, and probability laws keep on changing over time for any non-stationary process, and one can see that.

If you have a trend, then it's really hard to put one distribution on top of all the individual time points, as discussed in the previous lecture also, okay? So, hence, we have to convert any non-stationary components of a time series to stationary as much as possible before you try to model or before you try to forecast and so on, okay? And again, one more idea is that this is also done for simplicity because as discussed in the last lecture, if a time series is stationary, then the mean is constant, the variance is constant, the covariance or the correlation does not depend on time. So, the entire process is more like in a statistical equilibrium. So, it is really easy to kind of handle and try to put a distribution on top of that or try to model that and eventually try to forecast.

So, hopefully this very important point is clear that if you have any non-stationary aspects underlying in a time series, the very first idea is to convert those and then transform them into a stationary kind of a series, all right? All right, so now let's shift attention to R. I'll show you how exactly you can install R. So, I have this web page in front of me which gives you R studio for desktop, okay? Now I strongly recommend that rather than installing the plain R, you should actually install RStudio because RStudio is more user friendly and then it has like inbuilt options where you can import some packages and then play around much more freely. Okay. But in order to install RStudio, you actually have to install R first.

So, go to this website. Right. Then first download and install R by clicking this button. And then you have to actually download and install RStudio also. So, through the rest of this course, whenever we have some practical session in between, we'll try to do all those things using RStudio directly.

Hopefully, this is okay. All right. So, now we'll shift and then I'll show that how exactly the RStudio framework looks like. So, if some of you have already seen this, then it's a very good thing. But for those of you who do not have a flavor of either R or RStudio, then this is exactly what a typical window in R or RStudio looks like.

Now, of course, just for simplicity from now on, if I mention R, it means RStudio. So, I will not mention RStudio every time. Okay. So, if let us say we will do this in R or we will work this out in R. So, R means RStudio, which is this framework. Okay.

Alright. So, here just to tell you more about the panes that you see. So, let's say, so initially on top on the left side, this is the coding pane. So, I can write down any code. I can code it in the form of a program and then try to run this.

And then down here you see the console. So, console is where you can actually input some numbers, input some equations and then get the output. So, I'll say that this is more like if you were to do a few things on the go or on the fly. Okay? And whatever you do here, it's stored in this environment or you see history.

So, if you want to see which code you ran earlier, all right? So, you can actually go to history. Okay? And this is more like the directories on the computer that you have. So, let us say if you want to import something directly from here, you can do that.

So, it says files. So, all sort of files that you have here and all the plots that you have here. So, let us say if you try to plot something, then those plots would come here, okay? Alright, so initially again we can pay attention to this console here. So, let us say I can write down a few things here.

So, let us say 5 plus 2 or something like that, which is 7. So, I can use this as a calculator also. So, 5 plus 2 or 10 minus 5. So, 10 into 5 or 15 into 2. So, I can do all sorts of things using the calculator.

So, which is one very good thing. And now This is the area where you can actually write down a program and then try to run it line by line. Okay. So, what I've taken here is, just to tell you and just to acquaint all of you with very basic R operations.

So, you see here that the code name is basic R operations in R. So, by the way, any file that you create here would be stored as a dot R file somewhere. Okay. Just before that, ensure that if you are playing around in R and you want to save some codes or save some output somewhere, you have to create a separate working directory for this course. Again, this is a strong suggestion from my side keep a separate working directory, give it a name, let's say NPTEL time series course R files, or something like that. And in order to see what is the current working directory, you can input get WD.

So, get WD and then open brackets. And if you hit enter here, you'll see that the current working directory is nothing but this. So, users and then OneDrive and then documents. And you can actually change this. You can actually change this according to whatever folder you've made.

So, for that, you can go to the session. and then set working directory, right? And then choose some directory. So, let us say if you choose some directory, and then for that matter, let us say if you go to a desktop or anywhere, I created one working directory specifically for this course, which is named as TSM R material, okay? So, time series modeling R material.

So, you can open that, right? And then this tells you that the working directory has been changed to that folder, all right? So, again, just to repeat. So, again, just a suggestion from my side that, if you want to keep all the material that we'll do in R in this course in one particular folder, let's say on your desktop or somewhere else, then changing the working directory is important, okay?

And how do you do it? You go to session again and then set the working directory and then choose whatever directory that you've created on your laptop, all right? Okay, so my current working directory is this one, which is TSM R material, all right? Now we'll walk around briefly over this code here that I have. So, again, as I said, that this is a very basic code just to acquaint all of you with some

specifications as to how you can import some vectors or how you can input some numbers and so on and so forth. So, initially, if you see in this line 7 for that matter, we are basically assigning the value of 12 to some variable which is x right? So, for assigning any value to some variable in R I can actually use an arrow sign. So, x and then arrow and then whatever number I want. So, let us say 12.

So, I can actually do this using an 'equal to' sign also, but then a more professional way is to use an arrow sign. So, arrow tells R that the number 12 has been assigned to the variable which is x. So, by the way, if you want to run any line, you have multiple options. So, you can either go to this run here. So, it runs that command.

And the moment you run any line of this code, you can see that, that has been reflected in the console. So, can you see this? So, in the console, it says that we have successfully run the seventh line here. So, now if you want to see what exactly x is or what exactly the value of x is. So, you have to run the next line which is x. So, let us see if you run this, it will tell you that x is nothing but 12.

So, since we have assigned the number 12 to x, it gets reflected here. And the console is one area where you can see all the outputs either whatever you type in here as a calculator or whatever line you run from the code itself. Alright. So, I will do a slightly different thing now. So, let us say, if I change this to some other number.

So, let us say 24. Now again what do you do? So, let us say, you again run this line. So, x is now 24 and if you want to check it then, you have to run the next line where you have written down x, and then again it will tell you that x happens to be 24 ok. So, let me change this to 12 which was there earlier, and again try to run this line and so again my x is nothing but 12.

Now, in the code itself, I can combine a lot of basic mathematical operations that one can perform on x. So, let us say I can add a constant or subtract a constant, or divide x by some constant, or multiply x by some constant and line 16 is x to the power 2. So, this inverted triangle kind of a thing means you are basically raising x to the power of 2. So, x square basically. So, let us see what each of these commands would give you.

So, x plus 2 should give me 14. So, you see 14 in the console. x minus 2 is 10. x divided by 2 happens to be 6 clearly. x into 2 should give us 24. and then x to the power of 2.

So, x square basically should give me 144. So, in short, once you assign any value to any variable here, I can perform any basic mathematical operation that I want. So, addition, subtraction, multiplication, division, power, etc. So, point 3 is, let us say, if you want to save some output from the previous operation to some other variable, let us say y. So, how would you do it, right?

So, let us say the initial x is, so remember what x is again, so you have to keep this in mind. So, x is 12 in our case, right? So, x plus 2 should be 14, right? And then let us say if you want to save that output to a new variable which is y, okay? So, I can again perform a very easy equation here.

So, y and then arrow because you have to assign it and then what do you have assigned? so we assign x plus 2. Okay. So, if you run this and then again try to find out what y is? So, hopefully y should be how much? y should be 14. all right. So, now a sub-point here, so let's say showing the result of an assignment as you do it, so sometimes handy for debugging. So, this point is, let's say, if you have a very lengthy code. First, we'll see what it gives you. So, again, let's say y arrow x plus 2, but then within brackets, within brackets.

So, can you see here that, if you're keeping brackets in, or if you're assigning brackets around the same function, it'll kind of tell you the exact function predominantly, right? So, let's say, if you want to debug, if you want to find out where exactly the problem lay in your entire code, right? then I can actually go back and then backtrack. It is another way that if you're dealing with some lengthy codes, then it's kind of slightly easier to handle brackets rather than open-ended equations. Okay, so now the next point is how do you define a list containing multiple numbers?

So, let's say if you want to assign a vector. So, for assigning vector to any variable in R, you can actually use a C command. So, C and then within brackets, you assign the numbers. So, 1, 2, 3, 4, okay?

So, what this means is that you're actually assigning this entire vector containing four numbers, one, two, three, four, to the variable X using the command which is called as C. So, by the way, C in R is called as concatenate. So, concatenate means combine. So, essentially what you are doing is you are basically combining these four numbers as a collection or as a vector. So, let us say what happens if you run this. So, and if you want to find out what my x is.

So, your x is nothing but a vector now containing these four numbers. So, 1, 2, 3, 4. Now, surprisingly, I can actually do all the mathematical operations we did earlier on the vector itself. And then we will see what results we have. So, let us say x plus 2.

So, x plus 2, what it will do? So, x plus 2 is nothing but it is basically adding 2 to each number. So, 3, 4, 5, 6. Or let us say x minus 2. So, x minus 2, hopefully, it should subtract 2 from each number.

Something like this. So, minus 1, 0, 1 and then 2. And then x divided by 2. So, x divided by 2 should give you 1 half, 1, 1.5 and then 2. And similarly, x into 2.

So, x into 2 multiplies each number with 2. So, 2, 4, 6, 8. And then lastly, x square. So, x square should give you the squares of the corresponding numbers. So, 1 square, 2 square, 3 square and then 4 square.

So, I think this would be very handy if you want to kind of rearrange a few things here and there or if you want to assign some mathematical operation to a vector itself. Okay. So, now R works on lists. So, small comment here that R works on lists and then not vectors. And hence, you must use special forms of mathematical operations to perform matrix operations correctly.

So, I think this is telling us that how do you deal with matrices. So, probably we will see that slightly later. So, we will try to import a matrix and then we will try to manipulate the matrix itself. Okay. So, what do you mean by this point is that we have an example here.

So, let us say we will assign this entire vector containing 8 numbers to y. Alright. So, 1, 2, 3, 4, 5, 6, 7, 8. Now, again how do you assign any variable with a vector? You use C command. Okay.

So, let me run this line and then let us see what y is. So, my y is nothing but 1, 2, 3, 4, 5, 6, 7, 8 as a vector. Okay. Now let's see what happens if you try to multiply your earlier x that you had with this new vector, which is y. Now remember one thing. So, what was x?

So, again, if you're not sure what x is, you can again write down x and then hit enter. So, my x is a vector containing four numbers, one, two, three, four. And what is y? So, my y is a new vector containing eight numbers, one, two, three, four, up to eight. Now again, going back to the important note that R thinks that you have these two lists and they're not vectors.

So, if you try to multiply X into Y, we'll see what happens, okay? So, this is probably not what we want, right? Because what R has given us is slightly strange. So, 1 and then 4 and then 9 and then 16. So, what it has done is it has taken x 1 by 1, the numbers in x 1 by 1 and it has multiplied the numbers in y 1 by 1.

So, 1 into 1, 2 into 2, 3 into 3, and then lastly 4 into 4. So, 1, 4, 9, 16. But again, strangely, since we have exhausted all the numbers in my vector x, it again goes back to the first number that we have. So, let us say, 1 into 5, then 2 into 6, then 3 into 7, and then 4 into 8. And, this is a problem because the lengths of these two vectors are not matching.

So, the length of my y vector is 8, and the length of my x vector is just 4. So, handling this is slightly strange, which is probably not what we want. So, eventually the output is 1, 4, 9, 16, and then 5, 12, 21, 32. So, again my suggestion is that keep on playing on your end also. So, try to take different vectors of different lengths if you want and then try to perform all these operations and then see what you get.

For example, I can also do x plus y or probably x minus y or x divided by y. So, try to see yourself as to what you get. So, rather than repeating each and every operation again on both these vectors, so try it out yourself. Okay. All right. And then now, the next thing is how do you import some matrices, right?

So, let us say, so, what exactly is x again? So, x is 1, 2, 3, 4. So, vector containing 4 numbers, 1, 2, 3, 4, okay? So, now if you want to multiply that vector with its transpose. So, first transpose is what?

So, transpose is nothing but you if you have a column vector then convert it to a row vector or if you have a row vector convert it to a column vector. So, transpose of a matrix very basic linear algebra. So, multiplying any two matrices we have to actually use a special function which is percent star percent. So, you have to actually write down the star in between two percentage signs. So, x and then percent star percent is transpose.

So, now we will see what R gives us. So, now it gives us a matrix for that matter. Now, remember one thing when you write down any vector and assign that vector to some variable in R, it kind of considers that vector to be a column vector. So, again remember what x was. So, x was 1, 2, 3, 4.

So, think of it as a column vector. So, you have one column consisting of these four numbers 1, 2, 3, 4 and then you are multiplying that column vector with its transpose. So, if you have a 4 cross 1 vector multiplied by 1 cross 4 vector which is a transpose eventually you should get a 4 cross 4 matrix which you see here. And matrix multiplication is performed as per the rules. So, if you are not very confident about multiplying matrices, I think this would be a very right time to slightly go back and then try to revise the matrix multiplication first.

Or what we can do is we can again invert. So, transpose of x multiplied by x. And then here it should be a scalar because remember that transpose is now a row vector. So, 1 cross 4, and then if you multiply that by a column vector which is 4 cross 1. So, eventually, the answer should be 1 cross 1 which is scalar. And now we can go forward and then do lots of things using matrices themselves.

So, let us say how you create a matrix using some inbuilt function. So, by the way, one very big advantage of R is that, R contains lots of inbuilt functions. For example, matrix. So, matrix of Y comma 2 and we are assigning this matrix to some other variable which is Z1. So, first, let us see what do you mean by or what do you get as Z1.

So, my Z1 is nothing but such a matrix. So, 1, 2, 3, 4, 5, 6, 7, 8. Now, remember what R has done here. So, R has taken the entire vector which is y. Now, remember what y is. So, y is nothing but 1, 2, 3, 4 up to 8, that vector.

And now, if you perform this matrix function on y So, the first number that you specify is how many rows you should have. So, we are telling R that we should have two rows. But if you note that how R has assigned the numbers. So, it has assigned the numbers column wise.

So, it has filled the first column first, 1, 2. Then it has moved on to the second column, 3, 4. Then the third column and then the fourth column. So, if you are not very happy with this structure, you can actually mention by row equal to true. All right.

And then assign it to a slightly different notation so that you don't get confused. So, let's say Z2. So, what is Z2 now? So, Z2 is the same matrix, which is Y comma 2. But here instead we mentioned the extra thing, which is by row equal to true.

Now let's see what R gives us. So, my Z2 is what? So, Z2 again, consists of two rows because you have mentioned two here as the command. But then since you mentioned by row equal true, it kind of fills the values row-wise. So, it fills the first row first.

So, 1, 2, 3, 4, and then the second row 5, 6, 7, 8. So, depending on how you want to place the numbers in the matrix, you can actually write down by row equal to true or just skip that. So, I think the next idea is again important if you want to extract some numbers. So, let us say select a specific element. So, I can actually extract some numbers from a vector using a square bracket.

So, again I will tell you what x is if you do not remember. So, my x is nothing but 1, 2, 3, 4 and then let us say if you specify x and then within square brackets 1. So, what it means is that you want to extract the first element of the vector x. So, what should it be? The answer should be 1 ideally. So, we will see if it gives us 1 or not.

So, you can see that the answer is 1. Now, the second one is if you want to extract the second element from the same vector. So, x and then within square brackets 2. So, it should output the second element which is 2. Now, one can actually select a range of elements also.

So, let us say if you want to select all elements starting from the second position to the fourth position out of this vector x. So, what this should give you is nothing but 2, 3, 4. Or I can actually drop an element also. So, for dropping an element, I can use a minus sign. So, let us say x and then within square brackets, if you write down minus 2. So, this means that you are actually dropping the second element.

Or eventually I can drop a range of elements also. So, let us say if you want to drop all elements from the second position to the third position, then I can use a minus sign and 2 colon 3. So, this colon actually means a range. So, from the second element to the third element within some rounded brackets all right. So, since you are dropping the ones in between now my x is only left with these two which is one four. All right. Or I can specify which elements to drop or which elements to select right for example, something like this so let's say initially I assign a slightly new vector containing only two numbers 1 3 to y. Ok. Let me run this.

So, now if you want to see what y is? right so, my y contains only two numbers now which is 1 3. And now this is a very interesting way of extracting the elements in y from the vector X, okay? So, X and then within square brackets, if you specify Y, we'll see what happens. So, this will give you one three. So, essentially what I'm doing is since Y is one three, right? I'm telling R that I only want to extract the first element and the third element from my X vector.

And the answer is in front of you, which is one three, because X was one, two, three, four, right? And then if you're extracting the first element and the third element, I have 1 and 3 with me, right? Or if you include a minus sign here, it tells R that you want to drop those elements. So, let us say I want to drop the first element and the third element from my vector x and the answer should be 2 4. So, again obviously it is not very easy to cover each and every function in R or each and every inbuilt equation in R. So, you have to try it out on yourself.

So, by the way all these codes would be shared to you do not worry. So, maybe install R and then try playing around with some functions or not. So, for example, the next thing is to combine two lists. So, how do you combine two vectors, right? So, let us say if you want to combine x and y. So, again use a standard C command.

So, and then again by the way I am assigning the combination of my x vector and y vector to x again. So, let us see what this would give you. So, if you run this concatenate x comma y and then see what x is. So, now, my x is 1, 2, 3, 4 and then 1, 3 because remember what y was as defined earlier probably here. So, now, my new y is nothing but 1, 3.

So, if you want to combine x and y, it should be 1, 2, 3, 4 and then 1, 3. Now, how about reversing a list? So, reversing a list means you want to output the numbers in a reverse order. So, you have a function called as REV stand for reverse of course. So, if you apply REV on X, it should give you 314321.

Now, the next thing is one can actually test for some logical operations. So, whether 1 equal 5 or not. Obviously, the answer is no. So, it tells you false or whether 1 equal 1 or not. So, answer should be true or test some inequality.

So, by the way, inequality can be specified using exclamation mark and then equal to. So, 1 and then exclamation equal to 5 means not equal to. So, is 1 not equal to 5? Answer is yes. So, the output should be true, and is 1 not equal to 1?

The answer should be false because 1 is equal to 1. So, answer is false ok. Now, how do you assign or how do you tell that I have to specify some vectors in a range directly. So, you have multiple ways again. So, let us say colon as discussed, a short while back.

So, I can write down 1 colon 3. So, 1 colon 3 means 1 2 3 by the way ok or I can tell R to input a sequence instead. Now, sequence the short form is SEQ right. So, within a sequence, you have to specify from which number to which number and then what should be the jump size in the sequence. So, by how much jump should you keep on changing the number.

So, here if you let us say keep 0.12, then what R would give you is this vector. So, the first number is 0, the next number is 0.12, the next number is 0.24, and remember the last number should be 1, right? So, R has stopped at 0.96 because after 0.96, it would exceed. Or there is one more another way of assigning numbers in a sequence. So, first number is 0, second number is 1 and length.

So, you can tell R as to how many numbers you want. So, something like this. And the down the line you can actually have other inbuilt functions such as mean, variance, standard deviation. So, let's say if you input x vector, I can find out the mean of x, which is 2.5, or the variance of x, which is 1.66, or the standard deviation, which is 1.29, right? Or you can sum all the numbers in x. So, summing the list.

So, let's say sum of one, two, three, four is 10, or take the product of all the numbers. So, product is 24. Or find out the length of x. So, length of x should be 4, right? Which is what you see here. So, I guess, these are some very basic commands.

I mean, again, just to sum this up, you should try to install R on your own PCs or laptops, right? And then try to open this worksheet if you can. Try to play around with some functions. I mean, input some vectors using C. Try to manipulate some numbers here and there, okay? So, I guess this is a very, very basic worksheet on R. So, maybe down the line, of course, on a weekly basis, We will cover some time series aspects and then try to blend all the theory using some practical ideas in R. So, since time crunch, we did not talk about lot of different things. Let us say plotting, or how do you assign some distribution to random variables in R. But of course, down the line in subsequent weeks we can kind of cover that and then try to link all the things with what has been taught in that particular week. Thank you.