Time Series Modelling and Forecasting with Applications in R

Prof. Sudeep Bapat

Shailesh J. Mehta School of Management

Indian Institute of Technology Bombay

Week 07

Lecture 35: Practical Session in R – 7

Hello all, welcome to this course on Time Series Modeling and Forecasting using R. Now, again, just to quickly revise where we are standing this week. So this week, we focused on the broad idea of bivariate time series processes or rather multivariate time series processes. So, just to quickly revise a couple of points before we delve into today's practical session. Similarly, in front of you, you can see that today's session will be entirely based on a few R codes and applications of bivariate and multivariate time series processes.

But just before that, a quick revision: what exactly do you mean by bivariate, trivariate, or multivariate? So, the moment you transition from a univariate time series process—let us say a single stock price of Reliance or TCS—to something like monitoring a couple of stock prices. For example, Reliance against HDFC or Reliance against ICICI, then one cannot model that using univariate time series modeling such as ARIMA. Instead, one has to transition into bivariate time series modeling. For example, vector autoregressive processes (VAR), VRMA processes, or VMA processes, etc., right? Throughout this week, we have studied—in the earlier four sessions—some key time series models that transition a univariate space into either a bivariate, trivariate, or multivariate space.

So, in today's session, we will focus on some of these models discussed this week, such as VAR, VMA, or VARMA, etc. We will see how to model simulated data and later how to apply these models to practical data, right? Now, again, just to give you a quick background on where to deploy these vector models: the moment you transition from a single time series data point to multiple time series processes. For example, as discussed a short while back, if you are focusing on two different stock prices of different companies.

So, let us say HDFC versus Reliance. And then, the idea is that if you want to model them together or you want to try to find out the co-movement between HDFC stock price and Reliance stock price, then one can actually put forward a vector time series model on that, okay. Alright, so again, this is a typical R window that probably almost all of you should be familiar with. So, now that we are into week 7 and particularly, we have covered 6 practical sessions before this, right? So, how do you handle some datasets or how do you import some datasets, or even before that, how do you install some packages, right?

So, at least all those things should be kind of ready now. Now, again, just a very quick refresher as to if you forgot—let us say if you have studied or if you have watched the videos sometime back—then how do you install the packages? Again, you can simply type in install.packages and the corresponding name of the package here, for example, VARS or MTS, etc. Or the other quick way around to doing this is to go to tools and then click on the first option, which is install packages. And again, a couple of things to note here is that you basically ensure that this drop-down menu is selected to repository and in the brackets CRAN, and not the other one, right?

And then, whatever name of the package you want—let us say MTS or VARS, etc. You simply type it here and then. You make sure that you click or simply check this box which says install dependencies, and then hit install, okay. Now, of course, since we have already installed the packages, I will not redo the entire thing again. So, one can again install the package using these commands also, right.

But again, once you install any of the packages, again, do not forget to load them in the R environment using either the library command or the require command, etc. Okay. So, since I have opened a new R window, I have to do that again. So, I will click on library VARS and hit on run. And again, here in the console, you can see that it sort of tells you that the package named VARS is installed successfully and then we have borrowed the package inside the R environment also.

right. And then the other package we require is called as MTS. So, I will make sure that both of them are inside the R environment for us. So, VARS and MTS right. And again as you see that since I have not zoomed in on the console window, but you can drag this upwards and then see a zoomed in version of the console window as to what exactly is going on right.

So, once you hit on any library command and then input a package name, it sort of tells you that which package it is attaching inside the R environment, for example, MTS, right? And a further a few details about the package that you have just installed, right? So, let us say that which other package it has masked or something like that, right? So, a few finer details of which obviously, we need not worry too much about that, but again once you ensure that both the packages have been installed, Then again we will the very first thing we will do is we will try to simulate some data right.

So, we will first try to model the simulated data using some of the bivariate models or multivariate models, etcetera—let us say VAR, VMA, or VARMA—and then, towards the end of today's session, we will delve deeper into a practical dataset also. All right. So, as always, the first command to be run is set.seed, and then again, as discussed in a couple of earlier sessions also. So, the set.seed command kind of ensures that every time you try to simulate the process again and again, it sort of ensures that it will give you a random output. And here, it hardly matters what number you put in.

So, you can randomly put any number here. So, for example, 1, 2, 3, or for that matter, any other number also. So, this is just a pointer kind of thing. So, the next time you simulate the exact same thing again, it will again start from the same position, but then obviously give you completely random simulated data every time, okay? Okay. So, now, since we are dealing with multivariate time series processes.

So, what we will do is we will simulate two interrelated time series, right? So, the first illustration is to model some simulated data, right? So, for that, we require some interrelated time series to be simulated, right? Now, again, one can mention the sample size. So, the sample size that we want is, let us say, 200, right?

So, in each of the series, we want 200 observations, ok. So, let us say, just for convenience, we will name them as series 1 and series 2. So, let us say series 1 contains 200 observations, and series 2 also contains some 200 observations, alright. So, we will run this. And then here, if you take a note closely as to how we are actually simulating the data set.

So, the first command that we will give to R is that we want two separate vectors which are coming from a normal distribution. So, here, since we are specifying R norm, right. So, R norm stands for if you want to generate some random variables from a normal distribution. And then, if you do not specify any parameters inside the brackets, the default version is mean is 0 and then variance is 1, ok. So, what exactly are we doing here

is that we are simulating two interrelated time series processes, and then of what size again?

So, of size n, and then n in this case is nothing but 200, all right. So, our first sample would be named as E 1. So, we will run it, and then the second sample would be named as E 2, all right. So, E 1 is one of the series, and then E 2 is another series, all right. And now here, this R has a very convenient command called filter.

So, what we will do is we will sort of filter a few values and then here you can specify some coefficients. So, let us say 0.5 and minus 0.3, right, something like that. So, what we are doing is we are slightly changing the E1 and E2 which we just generated into slightly different vectors and then we will give them as name as Y1 and Y2. So, what exactly is y 1? So, y 1 is nothing, but filter command operated on E 1 and then this is my filtering coefficient set.

So, 0.5 and minus 0.3 and then method would be recursive. Similarly kind of similar y 2 is filter on E 2 and my filtering coefficients are 0.4, 0.2 and method is again recursive ok. So, let me run both these commands. So, y 1 and y 2. And now that you have kind of formed Y1 and Y2, so Y1 and Y2 would be our series 1 and series 2 eventually, ok, and not E1 and E2, ok.

Because how it so happens is E1 and E2 are completely independent, is not it? So, how did we generate E1 and E2? So, E1 and E2 while generating, we completely used a independent sort of a command which is RNORM. So, for example, E 1 has nothing to do with the values which are being generated in E 2 and vice versa right. So, to sort of make both the series interrelated we have to throw in some filtering there.

And this is the reason as to why exactly we are using this filter command here, all right. So, again, just to repeat. So, E1 and E2 are generated based on some completely normal 0-1 variables using the rnorm command, and then the sample size is 200 for each of the series. But then, clearly, E1 and E2 are completely independent, isn't it? Because E1 is generated through rnorm independently of E2. But how do we convert E1 and E2 to sort of make them interrelated? It is that we can apply this filter command on E1 and E2. Make sense so far?

So, now, just for convenience, what we will do is we will combine the simulated data in a matrix. So, the name that we are giving to that matrix would be sim_data. And then, cbind is a very useful command in R, which sort of combines multiple vectors into a

matrix kind of structure. So, cbind which vectors do we want? We want y1 and y2, and then not

E1 and E2, all right, OK. So, we will run this, and then we will give both the columns names as series1 and series2, respectively, right. Now that you have simulated some data, also tell R that the simulated data is actually time series data, OK. So, we will slightly change the name here. So, sim_data_TS.

So, TS stands for time series, of course. And then again, this is an inbuilt command, TS, right? So, TS would ensure that the generated data, which is there in SIM_data, has to be read as time series data, okay? So, TS applied on SIM_data. Now, very initially, just to visually check what the two series look like, right.

So, we can have a very preliminary plot, kind of a thing. So, plot.ts and then sim_data_ts, which is nothing but the time series object, right. And then we can give some heading, and then we can give different colors, and so on and so forth, all right. So, let me run this plot command and then show you the plot, okay. Let me zoom in the plot.

And then this is exactly how the simulated data looks like, all right. So, you have series 1 in the first row and then series 2 in the other, okay. So, even though both series might look like they are independent, but then, using the filter command, we are kind of ensuring that there is some co-movement or some interdependency between the two series, okay. So, series 1 and series 2, and here on the x-axis is nothing but the index. So, how many observations have we generated?

We have generated 200 observations. Does this make sense so far? So, predominantly, this is exactly how the simulated data looks. So, series 1 against series 2. So, we will close this window and then go back to the code. Now, the next important thing is we will try to fit all the models one by one, right?

So, the very first model we will try to fit is a simple VAR model, or rather, a vector autoregressive model, right? Now, again, just a very quick refresher: since we are dealing with multiple series, series 1 and series 2, I cannot fit an autoregressive model on both and then check for interdependency, right? So, a better way of dealing with such a situation, would be to model them jointly using some vector kind of model.

So, either VAR, VMA, or VARMA, etcetera. And then we will deal with all these models one by one. So, the initial model we will try is a VAR model. And this VAR, in caps, is again an inbuilt function in R, right? Now, again, just to show you very quickly: if you

hover over any of the commands, right, it sort of tells you what exactly should go inside the command.

So, can you see that that the yellow box if you see. So, VAR x comma p equal to 1. So, you have to specify the lag. So, since you are dealing with a vector autoregressive model, you would actually specify the value of P also. So, P is nothing but the order of the AR structure, is not it?

So, you have to first specify on what vector you are going to model that using the VAR model and then the second command is you want to specify some order in the AR model. And you can actually tweak couple of other things also which will kind of ignore as of now. So, this is broadly the VAR structure, we will give it a name as VAR underscore model and then VAR applied on simulated underscore data underscore TS. And then here we will kind of fix the order to be 2. So, 2 is nothing but the AR order.

So, P is the lag order. So, let me run this VAR command. Now, here of course, you can change this order if you want right if you want if you are more interested you can actually play around with multiple orders. So, let us say 1, 3, 4 etcetera ok. Now, let us see that what the console throws at us.

So, this is exactly the VAR model fitted on the simulated data, and then it shows you quite a lot of things, right? So, let us say the estimates. So, these are the estimates of the autoregressive structure. Right, and then these are the corresponding standard errors, and by the way, these are the coefficient matrix. So, now, again, just to remember that since we are dealing with multiple series, it is sort of throwing you a coefficient matrix kind of thing, right?

So, it will be phi 1, phi 2 of the first series and similarly phi 1, phi 2 of the other series, right? And then this is the standard error matrix, and then it sort of gives you the variance-covariance matrix, right? If not all, but then some of these things we studied in the theory portion this week, right? So, for example, variance-covariance matrix or correlation matrix, etcetera, right? So, these are some of the summary variables that R throws at us, and let us say here. So, towards the end, it also tells you the AIC value, BIC value, and HQ value.

So, all these are information criteria just to compare across, let us say, multiple models that one can actually fit, right? So, what is the first immediate thing that comes to our mind is that One can actually change the order right away. So, let us say instead of p

equal to 2, if you plug in some other value. So, let us say p equal to 3 or p equal to 1, and then if you want to compare as to which model would be superior.

So, would that be VAR with p equal to 2 or VAR with p equal to 3, let us say, right? So, the best thing to counterargue the statement is to look at some of the information criteria values, and whichever AIC value is least or whichever AIC value is lower, that model is superior. So, the idea behind information criteria and how you conclude based on, let us say, AIC, BIC, or HQ, we have discussed thoroughly in some of the earlier lectures in the prior video series. So, basically, AIC values and BIC values are just for comparison purposes across different models. So, now once you do that, then I can actually get a further summary of this VAR fit also.

So, let me run this, and then this tells you in a concrete manner as to how much the data is. So, it tells you the data length is 400, and why exactly? Because we have two different series containing 200 each. And then how many coefficients you have, how many residual values you have. So, all these values it throws at you.

So, the first thing is to apply a model using a VAR model. Now, once you model using a particular model, the next immediate thing is to forecast. So, again, this VAR pred is an inbuilt function in R, and again we will give the forecast vector a name. So, VAR underscore forecast, and then we have to forecast using which model? So, we have to forecast in the VAR model, of course, and then how far down the line you want to forecast.

So, you can specify h to be, let us say, 30 or whatever window down the line you want to forecast for, right. So, h equal to 30 stands for a 30-step-ahead forecast, simply. So, I can change this number to, let us say, 10, 20, right, etcetera. Makes sense. So, let me run this forecast command and let me print the forecast values or the predicted values. Here, as you see, this ends at 30 because you want 30 down the line, and then this would sort of throw you the 30 predicted values for both series.

So, the first column is series 1, and the second column is series 2, basically. So, 30 timestamps down the line, it sort of throws you the predicted values using a VAR model structure. So, now the next thing is to visualize the forecast. So, visually, are the forecasts making sense or not? So, for that, I will run the plot command initially and then specify the x-axis from 1 to 200, and then which data do you want to output first in the plot.

So, do you want to plot the actual data first or the simulated data first and then the forecast? So, here I will draw the plots in layers. So, the first layer would be the simulated data, and then, using this lines command that we see in line 37, we will add the forecast on that using a different color, of course. So, you can see the red color, okay. So, I will show you how exactly the code runs.

So, the first is the initial plot. So, by the way this is the initial plot of the series right. This is nothing but the simulated data from 1 to 200 and here if you see I have left some empty space for the forecast to fit in right. Now, on top of this plot, I want to plot the forecast using a red color and this is exactly how the forecast look like. So, let me zoom in for a better picture.

Now, can you understand that how the forecast look like using a VAR model? So, the black line that you see is entirely the simulated data and then the red line that you see is nothing but the forecast. Make sense? Alright. So, this is exactly how once you forecast something or once you model and then forecast something, visually how do you check if the forecast is making sense or not? Okay. Now, the next model we will try is a VMA model.

So, vector moving average model. So, the first model was VAR. So, vector autoregressive model and now the second model we will try out is a VMA model or vector moving average model. Now, again exactly similar story the only difference is instead of mentioning P here we will mention Q because Q stands for the order in the MA portion ok. And then probably we will give it a name as VMA underscore model and then we want to put forward this VMA model on which data again SIM underscore data underscore TS and again we can mention Q to be 2 all right.

So, let me run this VMA model command, similar to how the VAR command throws a number of things like coefficients, variance-covariance matrix, AIC, BIC, etcetera. Similarly, the VMA command also provides all these values. All right? So, estimates in matrix form—what exactly are the constants, if any, right? And down the line, it gives you the AIC value and the BIC value. Does that make sense? OK. And then, a similar summary that we saw earlier for the VAR command. So, let me run the summary.

So, I guess the summary just tells you how many data points there are, how many residuals there are, how many coefficients there are, how many standard errors of the coefficients there are, how many constants there are—all these things, basically. OK. So, for example, here you can see how many theta values you have. So, I have 8 theta values

and then 2 mu values. So, mu's are nothing but the constants, right? And here, if you add 8 plus 2— So, 8 plus 2 is 10, which is nothing but the number of coefficients you have.

So, again, if you go up— So, it tells you that you have 10 coefficients overall. Does that make sense? So, this is my VMA model, and now the last model we'll try out is a VARMA model, right? And then we'll see if the VARMA model performs better than, let's say, a VAR model, for example, etcetera. OK? Now, again, here the only change is that rather than VAR or VMA, I will apply the VARMA command, which is again an inbuilt command. Now, of course, all these commands—they are not inbuilt in the initial R. So, all these commands are part of the MTS package, right?

So, if you remember, if you scroll up, we installed this MTS package initially, right? So, the MTS package is required for fitting all these individual models. So, be it VAR, VMA, or VARMA, etc., right? All right. So, now, the next thing is to fit an appropriate VARMA model on SIM_data_TS, which is the data for us, and here again, we will specify the order.

So, P is 1, and then Q is 1. Now, again, make sure that since we are dealing with a VARMA structure, we have to specify both orders. So, one for the AR lag and one for the MA lag. So, P would be 1 in our case, and then Q would be 1. So, let me run this VARMA model structure, or rather the model fit, and again, similar to VAR or VMA commands, it throws us all these things, right?

So, let us say the residual covariance matrix, AIC value, BIC value, coefficient matrix, etc., all right. And then I can actually run the summary command on the VARMA model fit, and again, it tells us how many coefficients are there, how many residuals are there, AIC, BIC, theta values, etc., okay. So, now, once you have fitted the VARMA model, the next thing is to forecast, right? So, forecast using the VARMA model, and for that, we have this built-in command called VARMAPRED, which stands for PRED, right? Now, again, on what dataset do we want to apply the forecast?

We want to apply the forecast on the VARMA fitted model, right? So, VARMA underscore model, and for how long down the line? So, h to be 30 again, right? So, let me run the forecast and let me print both the values. So, again, as before, these are the printed forecasted values for 30 time steps down the line, and then initially the plot.

So, initially, the plot of the simulated data. Now, again, obviously, the simulated data would not be different, right? I mean. So, once you simulate data, we fix the simulated

data. And on that fixed data, they are kind of changing the models. So, from VAR to VMA to VARMA, right?

So, even if you plot the simulated data, it will not change from before, right? So, again, this is the simulated data, but what would change is the forecast. I mean, since you are applying a VARMA model now instead of a VAR model, the forecasted line would change, right? Something like that. Now, again, let me zoom in and then show you the forecasts.

So, this is exactly how the forecast looks like for a VARMA model. So, broadly so far we have covered that how do you fit different multivariate time series processes on let us say a simulated data. So, be it VAR or VMA or VARMA etcetera right. Now, the last thing that is left here in the first section is diagnostic checking right. So, again we can perform some preliminary diagnostic checking.

So, let me run this MTSDIAG. So, DIAG stands for diagnostics command. And it sort of throws at us this plot containing four subplots. Let me zoom in the plots. So, by the way, all these are residual plots, right?

And then here, nowhere you can see that you have some significant residuals, which is a good sign, right? Because we want that the residual should be independent and there should not be any underlying pattern among the residuals, right? So, since almost all the residuals are between the bands, you can sort of say that the residuals act as a stationary series, which we want. Alright and here again if you go to the console it tells you that you have to actually keep on hitting enter so it says that hit enter for getting the p value plot of the individual So, if you hit enter here now can you see that it sort of throws at us a p value plot and then almost all the p values are above the threshold right which means that you cannot reject the null right.

So, and then since you cannot reject the null we can actually go with the fact that the underlying residuals are stationary right and not just the p value plot if you can actually get multiple other things also. So, keep on hitting enter. So, this is one more plot right. And lastly, these are the residual plots right. So, broadly speaking we can actually cover the whole breadth of diagnostic checking once you fit let us say one of the multivariate models right.

Now, the very last thing we will do is we will apply the exact same thing on a real data. So, this data is named as Canada right. So, let me import the data and then I will show

you that what exactly the data set contains So, by the way, the data set contains all these things. So, employment rate, production, utilities, right, etc.

So, right, and then this is a practical data coming from a Canada scenario. Now, exactly how we did for the simulated data, we will try to fit all the models. So, the first model is VAR, right. Now, again I will not specify the details because all these things have been covered in the initial first half of the lecture video. But again the first plot we will try to plot is that VAR, right.

And again, I can get the summary, etc. And then, I can get the forecasts. Now, how far down the line do you want the forecast? You want the forecast for 10 time steps ahead. So, if you see, H is 10, right? And again, since we are focusing on multiple series here.

So, here in general, we are focusing on 4 series, right? That you saw in the actual data also. So, there will be 4 different columns of forecasts, right? And then these are the printed values. And then, this is the actual data, and this is the forecast line. By the way, this is the forecast line for the forecast values for 10 time steps ahead, basically. Make sense? All right.

And then, this is fitting a VARMA model to the data. So, let us say, instead of a VAR model, I will fit a VARMA model now. And then, I can get a summary as before. And then, eventually, the VARMA forecasts, right. So, VARMA forecast, again the print, then initial plot, and then the lines, right.

So, here you can see that the VAR model was making slightly more sense because, if you see, the forecasts are shooting up, which does not resemble the actual pattern of the data, right? So, at least visually, you can make sense of which model could be better. So, whether VAR or VARMA, etc., and eventually, you can actually print out the AIC values for both models. So, the VAR model as well as the VARMA model. So, here, as you see, the AIC for the VAR model is -6.30, which is much lower than the AIC for the VARMA model, which is 19.76, right?

So, again, just to summarize, between these two models, we can actually pick the VAR model both visually and statistically. As per the AIC values that are provided to us. So, again, just to summarize in one or two sentences. So, broadly, we have covered the whole idea about multivariate time series processes. So, we started with different models.

So, VAR, VMA, VARMA. And especially in today's session, we have combined all the things. And then we explained how you can implement some of the theory that we have studied this week in a practical scenario.

Thank you.