Time Series Modelling and Forecasting with Applications in R

Prof. Sudeep Bapat

Shailesh J. Mehta School of Management

Indian Institute of Technology Bombay

Week 04

Lecture 20: Practical Session in R-4

Thank you. Hello all, welcome to this course on time series modeling and forecasting using R. Now again, until the last lecture, especially in this week, we focused more on, let us say, model identification, then model estimation using MLE, method of moments, OLS, etc. And towards the end of a few lectures this week, probably the last lecture and the one before that, we focused more on diagnostic checking. So, here in this lecture, we will focus more on a practical example. So, this entire lecture will be about doing some simulation-based ideas or simulation-based exercises in R. Now, again, if you vaguely remember from the last R code that we had in the last week, which was code 3.

So, basically today we will start with code 4, as you see here. Right. And then from last week's R code, we focused on a bank data. Right. So again, just to be in line with what is happening with that data, we will again work with the bank data.

Let's say when it comes to diagnostic checking or probably modeling something, and so on and so forth. OK. All right. So, let's start. So, before that, again, we have to import a couple of packages.

So, let's say T-series and then forecast. Now again, just so I'm repeating this again and again, if the packages are not already there in the R environment, you have to go to tools and then install those packages. All right. So let's say click on install package. Then you type in the name here.

Let's say T-series or forecast or whatever other package you want to install. And then, again, make sure that this is from this drop-down menu. It's clicked on repository. Okay. It's clicked on repository, and then click install.

Okay. And once you've installed, then again, make sure that you call the packages in the R environment using the library command. Okay. So, we'll do that now. So, library T-series and then library forecast.

Okay. And again, this is the console window. So, whenever you are bringing any package into the R environment and if you do not have any errors, it will basically give you this sign like that. Okay. For example, here when you outputted library forecast, it did not give me any error, and then R successfully brought in the forecast package.

Okay. Alright, now before we go about dealing with diagnostic checking, we will do a very simple exercise, which is simulating from a Gaussian white noise process. Okay, so simulating some set of data or set of observations or set of time series values from a Gaussian white noise process. Now again, what do you mean by Gaussian? So, probably from one of the earlier lectures, you must remember that Gaussian means nothing but a normal distribution.

So, another name for a normal distribution is Gaussian distribution. And again, white noise probably all of you must know by this time. So, white noise is nothing but a collection of random movements which resembles noise, and probably you can assume a zero mean and some fixed weights. Alright, now again, how do you simulate from a Gaussian white noise, which is nothing but normal white noise? So, in R, you have this R norm function to generate some values from a normal distribution.

So, how many values are you generating? So, let us see if you delete this 0 here, and then initially, we will start with, let us say, just 10. Okay. So, the first argument or the first entry in this R norm function is how many values you want to generate. For example, as you can see here, so n or the sample size, then you basically specify the mean and the standard deviation.

Okay. So, here we are assuming that the mean is 0, and then the standard deviation is simply 1. So, this is more like a standard normal or a standard Gaussian variable. Okay. Alright.

So, let us run this. Okay. And then we are giving it a name, which is x, basically. So, again, if you want to find out what exactly is the data that you have generated, you can click on x and then hit enter. So, pretty much all these are the normal variables where the mean is 0, and the variance or the standard deviation happens to be 1.

And of course, if you keep on repeating this number of times. At every entry or every iteration, you should get different values because you are performing a simulation-based exercise. So, again, what I mean by this is that if you run this again, let us say the same code R norm, and then again get hold of what my x is, then you should get different values, of course. So, since you are simulating at every iteration, it should hopefully give you a slightly different kind of set of observations. So again, remember that we are kind of generating some normal variables with mean 0 and variance 1, and then we are kind of giving that vector the name x. Now, the next thing is to produce some plots.

So, let us see the first is we will draw a histogram, and then let me zoom in on the histogram. So, this is exactly how the histogram looks like. Now again, just by looking at the histogram and then assuming that you have generated the sample from a normal distribution, this is nowhere close to a normal, right? Because again, remember that if the distribution has to be normal, the histogram should be kind of symmetric, right? So, there should be higher bars in the middle and then lower bars at both ends, but this is more like a skewed kind of distribution, so essentially this is left-skewed, right? Because you have a long tail on the left side. Now again, why is this happening? Probably, even though you simulated from a normal distribution, but why is the histogram not looking kind of symmetric in nature?

So again, the answer to that question is that your sample size is very, very small, is it not? So, you are only generating a handful of normally distributed values. So, again, if you are only generating 10 such values. So, obviously, from 10 values, you would not get a completely symmetric kind of density curve. And again, at the same time. So, if you try to plot the data, let us say blindly.

So, plot X and then type equals L. So, type equals L means that you want a line plot. You want to join the dots there. So, let us see what it gives you. So, this is the plot. Now let me add two lines here.

So, the first line is I am adding a horizontal line at the mean of the generated data. So, h equals mean of x. So, whatever the mean of the data is, it will only produce a horizontal line at that value. So, this is my first horizontal line. And then I will add another horizontal line in a different color, let us say blue, at 0. So, essentially, I am adding two horizontal lines.

The first one is at the mean and the second one is at 0. And this is just to tell you that again if I zoom this plot slightly, then we will see what is going on. So, you can actually see a

small difference between the red line and the blue line. Can you see that? So, the red line and the blue line are not superimposed exactly.

So, the actual mean of the data happens to be slightly more than 0, and again, this may not be a very good indication. So, probably what we will do is we will, again, let us say, simulate and then see what happens. So, let us say again, if you simulate the same thing and again, I try to plot it. So, and hopefully, we should get slightly more of a difference this time. So, let us say, yeah.

So, the red line is here, and then the blue line is there. So, now probably this plot is making more sense as to what may be the problem. So, again, I am simulating a slightly newer data set now. So, again, all these observations are still coming from a normal distribution with mean 0, by the way. So, this blue line is at 0.

And this red line is at the mean. And then here, you can clearly see a large difference between the actual mean and 0. So, 0 is the assumed mean. So, since you are generating the observations from a normal distribution with mean 0, you should expect that the mean of the data should also be very close to 0, which is not happening here. So, you can see a large difference here.

And again, it boils down to the same question as to why this is happening. So, both these things are happening because your sample size is small. So, what we will do is we will again, so it says here also. So, repeat the above code with a larger n. So, now what we will do is we will probably increase this to, let us say, 1000 and then see what goes on. So, again generate the normal observations, but this time 1000 in number. Again, we are still assuming the mean is 0, the variance is 1, and then we will see how the histogram looks like.

So, if you look at the new histogram, then now you can clearly see that you can sense a symmetric kind of resemblance. So, the density is kind of looking more symmetric in nature, which is exactly what we want, alright, ok. And then, how about the plot now? So, let us say if you plot the observations, so this is the plot along with the mean line, which is in red, and along with the 0 line, which is in blue. And if you zoom this plot now, then, so here you cannot even make out the difference between the red line and the blue line. So, they are so close, which means that the mean of the data is so close to 0, which is again what we expect, that you cannot even differentiate between the two colors. So, essentially, this is exactly what we want to achieve, right.

So, if you are taking, let us say, a slightly larger sample, which is 1000, then obviously the generated sample has to resemble a normal distribution exactly, ok. Both in terms of, let us say, the histogram and in terms of, let us say, measuring the mean of the data, etc. So, I think this is a small initial part in this code, and then now we will pay attention to diagnostic checking. So, we will basically try to combine all that we have studied in the last lecture and the one prior to that, and so on and so forth. And again, we will take up the same dataset that we worked upon in code 3.

Now, just to give you a brief overview. Now, again, just to give you a brief summary of what that data set was. So, we are basically focusing on this monthly volume of commercial bank real estate loans in billions of dollars. So, you have a particular data set called bank case, and then this data set is stored in a text file. So, firstly, how do you bring it?

So, you have multiple ways, firstly, but AS dot TS again. So, AS dot TS ensures that you are bringing the data as a time series structure. And on top of that, the scan function is basically bringing the data into the R environment. Alright. So, and then we are giving it a name as bank underscore case.

Alright. So, let us run the data. So, bank underscore case, and again, if you want to find out what the observations underlying the data are, you can actually print the name of the data. So, bank underscore case. And you can actually see the entire data set being printed in the console.

So each of these numbers actually gives you the monthly volume of commercial bank real estate loans in billions of dollars. Alright, now again, vaguely, if you remember what we covered in code 3 pertaining to the same data, we are trying to fit some ARIMA model, if you remember, alright. Now, the first thing that we did there is that the actual data is not stationary, if you remember that, right, and then it contains some trend. So, if the original data has some trend, what do you do? So, you difference the data, right, so you apply the differencing operator, alright.

And essentially, how many times did you want to difference the data? Even if you again go back to code 3 and then probably run that again or something like that. Then you can actually remember that we were supposed to difference the data twice. So again, probably, I will show you again as to why that was the case. So let's see if you difference the data once and then run this line and then probably you run this ADF test. So, ADF is augmented Dickey-Fuller, right, which is used to test for stationarity.

So, if you run this ADF test on the first difference, then what do you get? So, you get that the p-value is larger than alpha. So, if the p-value is larger than alpha, we fail to reject the null, right. Right. So if you fail to reject the null, we have to go with the null hypothesis, and for any ADF test, the null hypothesis is that the model is not stationary.

Okay. So, which kind of concludes that the first difference is still not stationary. So, how do you handle such a situation? You go for the second difference. Okay.

So, you create the second difference, which is nothing but the difference of the first difference. So, something like that, and again, if you implement this ADF test on the second difference now, it should tell you a story which is slightly different. So, now the p-value is less than alpha, which is exactly what we want. So, the p-value is less than alpha, which We should be able to reject the null.

And if we are able to reject the null, we have to go with the alternative. And the alternative hypothesis is that the model is stationary. So, this is just a summary kind of idea as to what happened in code 3. If you do not remember vaguely. So, we had to differentiate the data twice, basically.

And then, if you remember, we fitted this ARIMA model with these orders. So, 0, 2, 1. Okay. And then again, you have to fit this ARIMA model on which data? So, you have to fit this ARIMA model on the actual data, which is bank_underscore_case, and not on the differenced data.

And why is that? Because you are keeping this middle order to be 2. Right. So, any ARIMA model where the middle order is 2 is telling you that you have to actually difference the data twice to achieve stationarity. Okay.

But if you are fitting an ARIMA model where the middle order is 2, that model has to be fitted on the actual data. So, your actual data follows an ARIMA model with these orders. So, 0, 2, 1. Now, again, one small point, just a summary kind of point, is why exactly these orders and not some other orders. So, I will show you exactly why, and I think we discussed this point in the earlier code also, that once you fit this ARIMA, you kind of get hold of the fitted coefficients and so on and so forth.

So here you see that. So, 0 to 1 essentially is nothing but MA, MA1. So you do not have any AR component. So, this kind of gives you the MA coefficient. And here if you see it produces an AIC value.

So I think until last time we were not confident about the AIC as to what it means. But now we are. So AIC is Akaike's information criterion. So just try to remember this value. So the AIC value is 26.1.

Now what we'll do is I'll change the order slightly and then see what happens. So let me change this to probably something like 1, 2, 1. Now again remember that if you're changing the orders, the orders have to be in close neighborhoods of the current order. So suddenly you can't change the order to 10 or 20, something like that, okay. So, if the expected ordering is 0 to 1 and if you want to make some changes and see if your model fit is improving or not, then you have to actually change the orders in close neighborhoods.

So, something like 1 to 1, okay. So now we will see what happens. So, if you try to fit this model, this ARIMA structure, and then again get hold of the fitted coefficients and so on. But now, by the way, remember that this is not an isolated MA structure. So, you have an AR component as well as an MA component.

So, there should be both the coefficients that the model should give you. But now the thing is that the AIC value has increased now. So, earlier it was 26.1, if I am not wrong. But now the current AIC value is slightly more, which is 27.9. So, ideally speaking, if you remember what the goal should be.

So, the goal should always be to minimize the AIC as much as possible. And hence you may try this out and then you may play around with the orders if you want. So, playing around means changing the AR order and MA order very slightly and then seeing what AIC you are getting. So, it turns out that for this particular triplet, 0 to 1, the AIC is in fact the least. Okay.

So, hence we will proceed with this ARIMA structure. So, 0, 2, 1. So, let me refit this again. Okay. And then this is the fit.

So, the AIC is 26.1. So, now once you have fitted the model, we can actually get hold of the residuals using a dollar sign. So, let me show you exactly how. So, if you delete this for a second and if you only have fit, right, and then if you apply a dollar sign afterwards, right, So, can you see that one can actually select out of all these options?

So, if you want to get hold of the coefficients or if you want to get hold of the variance or let's say the log likelihood or the AIC, and then here if you come down the line, you have a residuals option. So, if you want to get hold of all the residuals of the fitted model, you basically select this and then run this code. So, now and then I am storing all the residual

data in the RES name. So, RES contains all the residuals for the above fitted ARIMA model. Now, we will see and we will do some checks.

So, we will see an initial plot of the residual. So, again, type equals L would give you a line plot. So, this is exactly how the residual plot looks. So, let me zoom in slightly. Now again, just by looking at this plot, we are not seeing any major problems as such, right?

So, one can actually assume that the residuals are in fact behaving as if a stationary series behaves, right? And again, you do not see any major changing variance problems either, right? So, because pretty much all the residual fluctuations are within some tight variance bands, right? So, for example, you are not seeing any fanning out or funneling in kind of a structure here. So, in a way, the residual plot is kind of not looking bad at all.

Now, what we can do is we can again draw a horizontal line at 0 if you want, all right. So, let me run this and then show you the plot again. So, let us say this is a complete plot right where you see the behavior of the residuals along with the 0 line. So, here again, if you see, you can pretty much see some random movement along the 0 line, which is exactly what we want in terms of the residuals, okay. And now we have come to a point that we will do all the checks now.

So, firstly, normality check. So, are my residuals indeed normal or not? So, for that, we create a histogram. So, let me run the histogram, and if you zoom in on the histogram, you can actually see that the residuals may be symmetric, right? I mean, there might be a slight skew here, probably, but otherwise, you do not see any major problems when it comes to the histogram.

But we indeed won't only rely on the histogram. So, we can get hold of the Shapiro-Wilk test. So, in R, you have a very simple command called Shapiro.test. And then, you apply it on the corresponding variable. So, in this case, it is RES, which is residuals.

So, we will see what happens if you apply the Shapiro test. And then, here, you are getting a p-value, which is 0.2337, which is bigger than alpha. So, anytime we get a p-value bigger than alpha, we fail to reject the null, and then, for Shapiro-Wilk, the null hypothesis is that the distribution is normal. So, here, we can actually safely conclude that the distribution is indeed normal, and for that matter, we can also test using the Jarque-Bera test. So, again, in R, you have this function Jarque.bera.test, and again, you can see that the p-value is indeed bigger than alpha.

So, both the Shapiro test and the Jarque-Bera test are giving the same conclusion that the distribution of the residuals might be normal, ok. And now, the last thing one can do is create this normal Q-Q plot. So, if you create this normal Q-Q plot along with the straight line. So, let me zoom in on the Q-Q plot. So, here you can see that pretty much all the observations, probably apart from these three, are kind of falling on the line, right, which is exactly what we want, ok.

So, in fact, the histogram, Shapiro test, Jarque-Bera test, and the Q-Q plot are kind of concluding the same thing. Now, just in case, I will show you that if the distribution is not normal, then how does the Q-Q plot look like. So, for that, what we will do is we will create some simulated data from a chi-square distribution. So, r chi-square is to generate some simulated values from a chi-square distribution indeed, and chi-square is not normal, right, with these parameters. So, 4 is the degrees of freedom for the chi-square, and then how many values are we generating?

We are generating 100 values, ok. So, let me run this and then again show you the Q-Q plot of the chi-square sample, ok, along with this straight line. So, here, can you spot the difference now? So, if you zoom in, so of course, we can. So, at both ends, you have a lot of departures from this straight line, which is not what we want, right.

So, if you are observing something like this in a normal Q-Q plot, then you can actually safely conclude that the distribution is not normal, which is the case here, right? Because we are simulating the data from a chi-square distribution. Alright, so once you establish normality, then what is the next thing? So, detecting serial autocorrelation. So, or in other words, detecting whether the residuals are independent or dependent. Okay.

And then, for that, we have covered these two very famous tests. So, Box-Pierce and then Ljung-Box. Okay. And again, here you have to specify a couple of things. So, to specify the lag.

So, let us say up to how many lags you want to check. So, in this case, I have to check for up to the 10th lag, something like that, okay. And in both situations, you have a common function in R called as box dot test, alright. And how do you ensure that you are actually testing using a Box-Pierce or Ljung-Box is that you specify that in this type. So, type equal to either Box-Pierce or Ljung-Box.

So, both Box-Pierce and Ljung-Box are inbuilt kinds of function choices when it comes to type, okay? And then both these tests have to be again applied on the residuals, right? So,

RES, we will see what this gives you. So, if you run the Box-Pierce test, the p-value is 0.6278, and if you run the Ljung-Box test, The p-value is again 0.5188. So, in both situations, the p-value is very large, greater than 5 percent or alpha, right? And again, the same conclusion. So, here again, we can conclude that we fail to reject the null, right? Since the p-value is larger than alpha, and whenever you are failing to reject the null, you can actually conclude that there is no serial autocorrelation in the fitted model or among the residuals.

Okay. So, in fact, using both the Box-Pierce test as well as the Ljung-Box test, you can actually conclude that there is no serial autocorrelation among the residuals. Okay. Okay. So, now, once we are done with checking for normality and serial autocorrelation, the last thing is to check for heteroscedasticity or changing variance.

All right. And then for that, as we have covered in the last lecture itself, we have all these possible choices. So, one can actually look for creating some ACF plots and PACF plots of the residuals themselves. So, we will see what these two plots kind of tell you.

So, if you create this ACF plot and the PACF plot. So, let me zoom in on the ACF plot first. So, again, this is the ACF plot of the residual data. And not the squared residuals as of now. But again, here you can see that apart from the first lag, which is nothing but 0, all the further correlations are inside the bounds, which means the residuals themselves are behaving as if a stationary series behaves.

So again, this is exactly what we expect or this is exactly what we want. And possibly the same thing should be seen in the PACF also. So if you create the PACF plot and then zoom on the PACF plot instead, Then again, the same story. So in fact, PACF is better, right? Because all the correlations, regardless of whatever lag they are at, are not significant because all the spikes are in between the confidence limits.

So again, both the ACF plot and the PACF plots are suggesting that the serial autocorrelation or heteroscedastic nature of the residuals is kind of maintained. So residuals are behaving as if they are a stationary kind of a series. But like we discussed, we have to also check for ACF and PACF plots of the squared residuals instead. So, and not just the actual residuals.

So, what do we see here? So, let us see if you create this ACF plot of the squared residuals instead. So, we will see how the plot looks. So, this is exactly the ACF plot of the squared residuals. Let me zoom in on the ACF plot.

And again, we see the same kind of story. So, apart from lag 0, all the further correlations are between the bands, which means that the correlations are not significant. And then, lastly, how about the PACF plot of the squared residuals? So again, it should resemble the same conclusion. So again, if you draw the PACF plot of the squared residuals, then again, you can see the same structure, right?

So again, all the correlations or all the spikes, regardless of what lags they are at, are kind of not significant because all the spikes are between the confidence limits, right? So, in a way, this is more of a visual check. So, one can actually create some ACF plots or PACF plots of the residuals themselves or, on the other hand, one can create some ACF plots and PACF plots of the squared residuals in a way. And now, rather than creating ACF and PACF plots of the residuals or the squared residuals. So, why not just formally test this using a formal hypothesis test?

And here we kind of apply the Weitz test. So, if you apply the Weitz test, again in R you have a simple-looking function called Weitz dot test. So, if you apply this Weitz dot test on the residual data, then what do you get? So, if you apply this Weitz dot test on the residual data, So, the p-value is again very large.

So, 0.8992. So, again the same conclusion that was drawn from the earlier test also that if the p-value is bigger than alpha, you fail to reject the null. And the null hypothesis under either the Weitz test or the Bruch-Pagan test that we covered last lecture in the last lecture is that the variance is constant. So, here since the p-value is bigger than alpha, it is 0.89 close to 0.9, which is very, very large, then we fail to reject the null. So, we can actually safely conclude that the variance is not changing or the variance is constant.

So this is pretty much checking for normality, checking for independence, and then checking for changing variance for the residuals. And I just like to point out this last comment here that in R you have this very interesting function called check residuals. So, what you do is you kind of apply this function on the fit itself. So, if you remember, bank underscore fit is my ARIMA fit of the model. So, if you apply this check residuals function on bank underscore fit, what does that tell you?

It sort of gives you all these plots in a bunch. So, initially, you can see that this is the behavior of the residuals. Probably, this is exactly what we saw just a short while back. So, this is nothing but a plot of the residuals. It says here.

So, residuals from this fitted ARIMA model of these orders. So, 0 to 1, and then here you see the ACF of the residuals, OK? And here, this is a check for normality. So, in a way, this check residual or check residual function in R is an inbuilt function; it sort of gives you all possible angles.

So, let us say checking for independence through this plot, or probably checking for normality through this plot, or probably checking for constant variance through this plot, because even if you simply have a plot of the residuals, here you can clearly observe that the variance is not changing, right? So, I will say that one suggestion is that once you are through individually with all the above sections, for example, normality, serial autocorrelation, or changing variance, right? Then, just to back it off, you can actually use this check residuals function in R, OK? So, I think in today's lecture, we have managed to cover quite a lot in terms of diagnostic checking from a practical point of view. Now, again, let us say tomorrow, if you want to analyze any other practical dataset, the only thing you have to do is you have to change the data right here. So, instead of bank case, you can actually bring in some new data that you want.

But again, remember one thing: you should not jump to diagnostic checking directly. It is always a process. So, first, try to fit some model. So here, we have tried to fit this ARIMA model. Because once you fit a model, only then do you get access to residuals.

Without fitting, how can you check for diagnostic checking, right? So, you have to fit a certain model, which is the best possible model, given all different, let us say, information criteria or probably visual checks, and so on and so forth. And then, once you come to a concluding kind of model structure, let us say, ARIMA with certain orders, then one can actually get hold of the residuals of the model and then perform all these diagnostic checks, let us say, either individually or using this check residuals function in Thank you.