Group Project Assignment

Time Series and Financial Time Series (TS25)

Nina Deliu and Brunero Liseo

Due Sunday, December 7th 2025 23:59 on Moodle

General Instructions

I expect you to upload your solutions on Moodle as a **single running** `R Markdown` file (`.rmd`) + its `html` or `pdf` output, both named `ProjectWork_xx`, with `xx` the group name of your choice (communicated on Moodle). Assignments are to be performed in groups of maximum 3 students, and individual projects are also welcomed. The report must be submitted by the group leader using the dedicated form on Moodle.

You will give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Your responses must be supported by both *textual explanations and the code* you generate to produce your results. *Just examining your various objects in the "Environment" section of RStudio is insufficient – you must use scripted commands and functions.*

The final grade will be based on the methodological correctness and the arguments you give in the answers. Higher grades will be granted for insightful comments, smart coding and clever data visualization.

The length **must not** exceed 30 pages.

R Markdown Test

To be sure that everything is working fine, start `RStudio` and create an empty project called `ProjectWork_TS25`. Now open a new `R Markdown` file (`File > New File > R Markdown...`); set the output to `HTML mode`, press `OK` and then click on `Knit HTML`. This should produce a web page with the knitting procedure executing the default code blocks. You can now start editing this file to produce your project submission.

Important Info

- For more info on `R Markdown`, check the support webpage that explains the main steps and ingredients: R Markdown from RStudio.
- For more info on how to write math formulas in LaTex: Wikibooks.
- **Policy on collaboration**: *Collaboration on homework assignments with fellow students is **accepted**. However, such collaboration should be clearly acknowledged, by listing the group of the students with whom you have had discussions concerning your solution. You may **not**, however, share written work or code after discussing a problem with others. The solutions are specific to **your group**, and any violation will include a grade penalty.*
- **Policy on AI**: *Any support from AI systems, such as **CheatyGPT**, should be acknowledged and justified. Omission of acknowledgement will result in a penalty proportional to the extent of their use.*

Exercise 1: Stationary, but not Ergodic...

Background

In a strictly stationary or covariance (i.e., weakly) stationary stochastic process, no assumption is made about the strength of dependence between random variables in the sequence. For example, in a weakly stationary stochastic process it is possible that $\rho_1 = \mathbb{C}\mathrm{orr}(X_t, X_{t-1}) = \cdots = \rho_{100} = \mathbb{C}\mathrm{orr}(X_t, X_{t-100}) = 0.5$, say. However, in many contexts it is reasonable to assume that the strength of dependence between random variables in a stochastic process diminishes the farther apart they become. That is, $\rho_1 > \rho_2 > \ldots \rho_{100} > \ldots$ and that eventually $\rho_h = 0$ for h large enough. This diminishing dependence assumption is captured by the concept of **ergodicity**.

The formal definition of ergodicity is highly technical and requires advanced concepts in probability theory. Hayashi (2000) pg. 101 gives the following formal definition of ergodicity: a stationary process $\{X_t\}_t$ is said to be ergodic if for any two bounded functions $f : R^k \to R$ and $g : R^l \to R$,

$$\lim_{n\to\infty} |\mathbb{E}\left[f(x_t, \ldots, x_{t+h})g(x_{t+n}, \ldots, x_{t+n+l})\right]| = |\mathbb{E}\left[f(x_t, \ldots, x_{t+h})\right]| \, |\mathbb{E}\left[g(x_{t+n}, \ldots, x_{t+n+l})\right]| \, .$$

However, the intuitive definition captures the essence of the concept. Heuristically, a stochastic process $\{X_t\}_{t=-\infty}^{\infty}$ is **ergodic** if if it is asymptotically independent: any two collections of random variables X_t and X_{t+h} partitioned far apart in the sequence (that is, for h large enough) are essentially independent.

If the process is stationary and ergodic the observation of a single sufficiently long sample provides information that can be used to infer about the true data generator process and the sample moments converge almost surely to the population moments as the number of observations tend to infinity.

As mentioned in class, all ergodic processes are stationary, but a stationary process is not necessarily ergodic. The simplest example is a constant series: if you draw y_1 from some distribution and then set $y_t = y_1, t = 2, 3, \ldots$, then the process is stationary, but exhibits maximum serial dependence, as its value is frozen at the initial state. Here are other three simple examples.

Example 1 Let $X_t \sim GWN(0, \sigma^2)$ and $Y \sim G(\mu, \sigma_Y^2)$, where GWN denotes a Gaussian white noise and G the Gaussian distribution, respectively. Note that Y does not change with t. Assume $\{X_t\}_{t=-\infty}^{\infty}$ and Y to be independent. Define now $Z_t = X_t + Y$; then, $\{Z_t\}_{t=-\infty}^{\infty}$ is weakly stationary but not ergodic.

Example 2 Suppose that $\{\epsilon_t\}_{t=-\infty}^{\infty}$ are iid Gaussian random variables with mean 0 and variance σ^2. Consider a Bernoulli random variable $Y \sim Bern(p)$ with outcomes $\{1, 2\}$, and assume the chance of either outcome is half, that is $p = 0.5$. Suppose that Y stays the same for all t. Define you process of interest:

$$Z_t = \begin{cases} \mu_1 + \epsilon_t & Y = 1 \\ \mu_2 + \epsilon_t & Y = 2, \end{cases}$$

with μ_1 and μ_2 two fixed but unknown parameters.

Example 3 Let $\epsilon_t \sim GWN(Y, \sigma^2)$, with $GWN(Y, \sigma^2)$ denoting a Gaussian white noise process with fixed variance σ^2 and random mean $Y \sim U[-5, 5]$, where U is the Uniform distribution. Note that Y is not indexed by t, thus it does not change with t. Assume $\{\epsilon_t\}_{t=-\infty}^{\infty}$ and Y are independent. Define now your process of interest $Z_t = \theta Z_{t-1} + \epsilon_t$. Then, based on the value of θ and the configuration of the initial state of the process determined by an observed $Y = y$, the process can be weakly stationary but not ergodic; sometimes neither stationary or ergodic. To this purpose, we assume $|\theta| < 1$.

Your task

You are required to solve the task for **one** of the above examples, illustrating analytically and with R its properties.

The choice of the specific example will be based on a random sample of the values 1:3 according to a seed specified by the date of birth (format DDMMYYYY) of the group member with the closest date of birthday to August 21. For example, if the group has 3 members with dates of birth 23-07-2000, 14-12-2005, and 09-04-1999, then the one with birthday closest to 21-08 is the first one, and the seed is set to 23072000. You shall also report the name of the student with associated date of birth.

a. We first want to check whether and when the process is stationary. Using the definition of weak stationarity, prove that the process $\{Z_t\}_{t=-\infty}^{\infty}$ has constant mean and variance, with the autocovariance function independent on t. Intuitively, if the process is stationary, then the value of Y determines the equilibrium state of the process. *Note*: As the proof will involve two nested random quantities, you may need to use the Law of Total Expectations and the Law of Total Variance.
b. To assess ergodicity let's now focus on the autocorrelation function. Show that $\rho_h > 0, \forall h$, contrasting with the intuitive idea of an ergodic setting. Intuitively, while stationarity refers to the presence of an equilibrium state, ergodicity tells us whether the equilibrium is unique.
c. Using the `R` functions `rnorm()` and `runif()` or `rbinom()`, setup a suitable simulation study to double-check the previous result (start with $\sigma^2 = \sigma_Y^2 = 1$, $\mu = \mu_1 = 0$ and $\theta = 0.9$). Remember to set the seed before simulating the data, say `set.seed(21081991)`.
   Once you have generated a time series of length $T = 100$, use the `ts()` function to transform it in a time-series object.

   - Plot the simulated series and comment on its stationarity.
   - Compute the empirical mean and variance and compare them with the expected theoretical values obtained in point a.
   - Show in a single plot that by repeating the generation process say 5 times, the process converges toward a different equilibrium state depending on the observed value of Y. (NINA: CHECK)
   - Setup a more extensive Monte Carlo simulation study (with $M = 10000$) to evaluate the theoretical values. Compute again the mean and the variance (for each t, using the function `apply()`) of your process $\{X_t\}_t$. What can you say about the obtained values in relation to the theoretical ones?
   - Let's now have a look at the ACF; you can use the function `acf()` from `stats` package or `auto_corr()` from `simts` package. Set a suitable `lag.max` (e.g., `lag.max` $= 50$) and plot the ACF. What do you notice? Are the sample correlation, say $\hat{\rho}_h$ values, reflecting the theoretical findings? Double-check the theoretical values by computing the correlation (you can use the `cor()` function) between some (Z_t, Z_{X+h}) couples at different lags h of the simulated samples. In light of this, do you think we can check the ergodicity assumption from a *single* sample of one time series?

- Comment on the estimator $\hat{\rho}_h$ in relation to the underlying theory for obtaining consistent estimates.
- Play around with different values of the unknown parameters (just a few, well chosen values, will be enough).

Exercise 2: Quantify and Qualify your Forecast...

Before you dive into the real-world, **a brief word of motivation**. Time-series forecasting is a bit like predicting whether your cat will knock something off the table: the past contains clues, the future contains chaos, and models such as ARIMA stands proudly in the middle pretending it can handle both.

In this exercise, you will use real financial data (which, like cats, often behaves unpredictably and ignores your assumptions) to explore how you can quantify the uncertainty in your forecasts. Along the way, you'll discover that analytic intervals may be too optimistic, while other non-parametric solutions such as bootstrap are needy (they require *many* resamples/simulations). More sensible and statistically honest friends exist and we will cover them during the last weeks of our course. Stay tuned!

Your task (for now...)

In this exercise, you will fit ARIMA-type models to a financial time series (of your choice) and compare alternative predictive uncertainty evaluation methods.

a. Load in R your working time series, motivating your choice. You may use any publicly-available financial series (stock, ETF, index), e.g., from Yahoo Finance via the `quantmod()` package. You can decide to work with daily or monthly data, adjusting the `frequency` accordingly.

b. Perform any pre-processing and transformation step, as well as any relevant inspections for identifying and choosing the most appropriate model within the ARIMA class. You are expected to follow the Box & Jenkins procedure covered in class. Comment on your output, in terms of both the selected ARIMA order (and its goodness of fit or appropriateness) and parameter estimates. Compare your result with the output of an automatic model selection with `auto.arima()`. A useful guidance is provided here.

c. Using the fitted model from the previous point–either your selection or the automatic suggestion–perform a forecast for the next $h = 30$ time-points. An example R script is provided below. Clearly, what we are trying to forecast is unknown (or we would not be forecasting it), and so we think of it as a *random variable*. When we obtain a forecast, we are estimating the middle of the range of possible values the random variable could take. Discuss the predictor you used and the trade-offs between choosing a low and a high value of h.

```r
library(quantmod)
library(forecast)

# Specify start and end dates
start_date = as.Date("2024-01-01")
end_date = as.Date("2025-11-01")

# Load S&P 500 data (Attention to '^GSPC' symbol when copy-pasting!)
invisible(getSymbols("^GSPC", src = "yahoo", from = start_date, to = end_date))
```

```
SP500 = na.omit(GSPC$GSPC.Close)

# Fit ARIMA (auto selection)
fit = forecast::auto.arima(SP500)
print(fit)
```

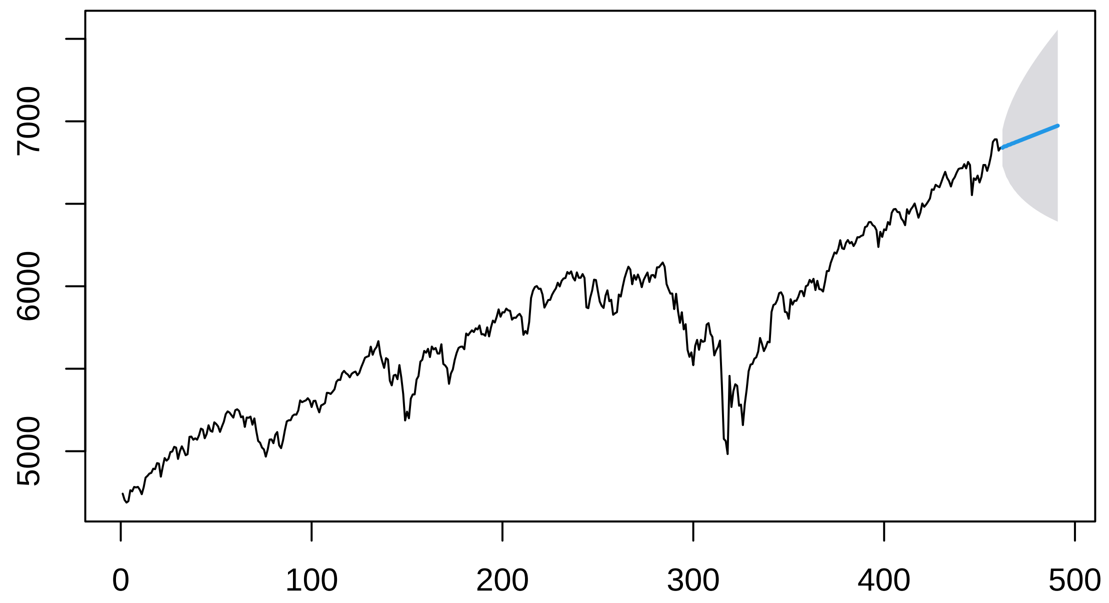
```
## Series: SP500
## ARIMA(1,1,1) with drift
##
## Coefficients:
##           ar1     ma1   drift
##       -0.7698  0.6983  4.5523
## s.e.   0.2096  0.2350  2.5171
##
## sigma^2 = 3186:  log likelihood = -2506.5
## AIC=5020.99   AICc=5021.08   BIC=5037.52
```

```
# Forecasting
h = 30
fcast = forecast::forecast(fit, h = h)
fcast$mean
```

```
## Time Series:
## Start = 462
## End = 491
## Frequency = 1
##  [1] 6839.279 6848.045 6849.353 6856.403 6859.033 6865.065 6868.478 6873.908
##  [9] 6877.785 6882.857 6887.009 6891.869 6896.185 6900.919 6905.331 6909.992
## [17] 6914.461 6919.077 6923.580 6928.170 6932.694 6937.268 6941.803 6946.369
## [25] 6950.911 6955.471 6960.018 6964.575 6969.123 6973.678
```

d. Usually, the forecast is accompanied by a **prediction interval** giving a range of values the random variable could take with relatively high probability. For example, a 95% prediction interval contains a range of values which should include the actual future value with probability 95%. **The value of prediction intervals is that they express the uncertainty in the forecasts. If we only produce point forecasts, there is no way of telling how accurate the forecasts are. However, if we also produce prediction intervals, then it is clear how much uncertainty is associated with each forecast. For this reason, point forecasts can be of almost no value without the accompanying prediction intervals.** In R, a `forecast` object provides you an idea of this uncertainty. Report the assumptions and the underlying computation behind the *intervals* generated from the `forecast()` command in R. Discuss benefits and harms.

```
alpha = 0.05
fcast = forecast::forecast(fit, h = h, level = 1 - alpha)
plot(fcast, main = " ")
```



Bootstrap Prediction intervals give us an idea of the uncertanty or accuracy of our predictions. Unfortunately, almost all prediction intervals from time series models are too narrow. Authors have found that the actual coverage in certain cases may provide coverage as low as 71% rather than the nominal 95%. This phenomenon arises because they do not account for all sources of uncertainty, including, 1. The random error term; 2. The parameter estimates; 3. The choice of model for the historical data; 4. The continuation of the historical data generating process into the future. We generally only take into account the first one or two of these sources of uncertainty. Even if we ignore the model uncertainty and the uncertainty due to changing data generating processes (sources 3 and 4), and we just try to allow for parameter uncertainty as well as the random error term (sources 1 and 2), there are no algebraic solutions apart from some simple special cases. To overcome this problem, we can use Bootstrap (Recall from **Advanced Statistics** course?!)

e. When a certain (e.g., Gaussian) distribution for the forecast errors is an unreasonable assumption, residual bootstrapping is a simple alternative; this only assumes that the forecast errors are independent. Guided by Section 3.5 here, formally describe the residual bootstrap procedure and implement it in the context of your chosen time series. Do not forget to set your seed! Plot the resulting intervals and comment on differences with point d. Any issues to report?

f. In the previous point, we bootstrapped the residuals of our time series in order to simulate future values of a series by mimicking the same error patterns. However, because there may be autocorrelation present in the remainder of a series, a more ideal alternative for time series is the Moving Block Bootstrap (MBB). Simple slides here illustrate the intuition behind it, while a more advanced reading is this paper. The advantage of MBB is to preserve the (local) dependency structure of your data. Write your R code to implement the MBB; in R, the built in functions `tsboot()` and `boot.ci()` that together implement this strategy. Comment on your output!

g. *Bonus (+2 points on your project grade).* Another use for these bootstrapped time series is to improve forecast accuracy. If we produce forecasts from each of the additional time series, and average the resulting forecasts, we get better forecasts than if we simply forecast the original time series directly. This is called **bagging** which stands for **bootstrap aggregating**. Following the guidance from this page and/or this attached paper, try it out with your data and model!